DL2Go: Editable Digital Libraries in the Pocket

Hyunyoung Kil, Wonhong Nam, and Dongwon Lee*

The Pennsylvania State University
University Park, PA 16802, USA
{hkil,wnam,dongwon}@psu.edu

Abstract. A preliminary framework, termed as DL2Go, that enables *editable* and *portable* personal digital libraries is presented. For mobile offline users of digital libraries, DL2Go can: (1) package digital libraries into mobile storage devices such as flash drives, along with needed application softwares (e.g., wiki and DBMS), (2) (de-)compress contents of digital libraries to address storage constraints of mobile users when needed, (3) enables users to add, delete, and update entities of digital libraries using wiki framework, and (4) share/sync edited contents with other DL2Go users and the server using web services and RSS framework.

1 Introduction

Due to the significant improvement in underlying technologies, strong support from governments (e.g., NSF NSDL, DELOS), and rapid increase in its user base, *Digital Libraries* (DLs hereafter) have became a norm, reaching out a large number of audiences in cyberspace. For instance, there are currently about 930 DLs at NSDL[1] for diverse audiences, ranging from kids to K-12 educators to academic scholars. Although abundant, however, in this paper, we claim that the current support of DLs is not without a problem. In particular, we believe that current DLs are not well designed for *nomadic users* who want to use DLs in *offline environment*.

Example 1. Consider the following motivating scenarios: (1) Together with students, a K-12 teacher "John" leads a bird watching trip to a remote park. He prepares his laptop computer with a CD-ROM encyclopedia including full of photos and descriptions of various birds. However, when they want to add their comments about birds spotted, they have to write on papers because the encyclopedia does not allow them to write comments. Upon returning from the trip, "John" realizes that it is hard to gather students' comments and share them with other K-12 educators; (2) A computer science professor "Sue" needs to finish out the draft of her paper while she flies from Los Angeles to New York. However, in the airplane, she realizes that she does not have access to literatures that she needs to read and cite; (3) An officer "Kim" at a non-profit NGO travels through backcountries in Africa to distribute a new version of the

* Partially supported by IBM and Microsoft gifts.

[1] http://crs.nsdl.org/collection/

G. Buchanan, M. Masoodian, S.J. Cunningham (Eds.): ICADL 2008, LNCS 5362, pp. 1–11, 2008.

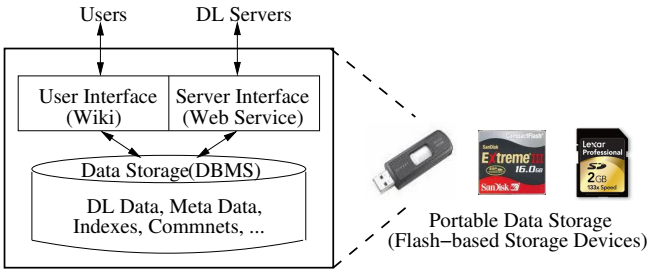


Fig. 1. The overview of DL2Go framework

children's book of Encyclopedia to children who use laptops from "One Laptop per Child"[2]. However, she cannot carry her laptop computer nor external devices because of too many baggages. Moreover, due to the weak network infrastructure and lack of CD-ROM drives, the only option is to store the online book in a USB flash drive. □

Tasks sketched in these examples cannot be resolved easily in conventional DL frameworks where users are assumed to have stable network access to DLs and majority of users' operations is only "reading" the contents of DLs. To address these new needs, therefore, we have developed a novel framework and prototypes, termed as DL2Go, with the following desiderata in mind: (1) DL2Go is easy to carry around for nomadic users, (2) DL2Go is cheap to produce and distribute, (3) DL2Go supports direct update on the contents of DLs in both personal and collaborative environments, (4) DL2Go supports the sharing of contents of DLs with other users, and (5) (in addition) DL2Go supports all regular features of DLs such as archiving, indexing, searching, and browsing of contents.

2 The DL2Go Framework

Figure 1 illustrates the preliminary design of DL2Go. Consider a DL X that users may have access to (e.g., arXiv [6]). Initially, users download the current snapshot of X and package it into a *portable data storage*. Once building a personal copy of X, users can browse, search, read, and even add, delete, and update the contents of X freely via the *editable user interface* of DL2Go. Users can *sync* the contents of X in a portable storage with ones in the server. Next, we present various issues that we have considered in implementing the prototypes of DL2Go framework.

2.1 Portable Data Storage

We envision that users of DL2Go have some forms of electronic devices (e.g., laptop, PDA, smartphone) with limited space in the primary storage area. Since

[2] `http://laptop.org/`

DL2Go aims to be a portable DL in a mostly offline environment, we consider portable devices to host contents of DLs. As a secondary data storage, popular choices include external HDDs, Solid-State Drives (SSDs), CDs/DVDs, and flash-based storage devices. External HDDs or more recently-developed SDDs can store a large amount of data for relatively low price, but are bulkier to carry around for DL2Go users. Although CDs/DVDs are affordable and a popular device in many settings, they are not appropriate for data with frequent updates. Even if there are writable CDs/DVDs, their usage is still limited. Furthermore, not all machines that we have in mind come with a reading device for CDs/DVDs. Finally, flash-based storage devices are small and supported in many electronic devices. Furthermore, with the rapid advancement in flash technologies, more and more affordable flash-based devices with an ample space become available. These days, it is not difficult to find a USB flash drive with 32GB space for less than $100. Among all the candidates, therefore, we decide to use flash-based storage devices. Figure 1 shows three popular commercial examples – USB Flash Drive, CompactFlash, and Secure Digital. Even though the I/O speed of current flash-based storage devices tends to be slower than that of HDDs (e.g., 30MB/sec vs. 100MB/sec for read/write), we believe that running DLs in flash-based devices is still viable.

2.2 Data Management

DLs can contain various types of data as their contents – text/multimedia data objects, metadata, index, comments, etc. Depending on the characteristics of the DL data, different data management system can be selected. For DLs with a relatively small amount of data (e.g., DLs with textual data only), one may use the file system with the support of OS to manage the contents of DLs – i.e., data are stored as files in the hierarchy of folders. Although simple and appropriate for small DLs, however, this approach does not work well for DLs since a large amount of storage space is wasted. For instance, MS Windows file system allocates at least 4KB for a file even if it includes only 1 character. If we were to store all 1 million paper metadata information in DBLP as files, then it would require roughly 4GB just for the metadata information without the PDF copies of papers (see Table 2), making it inapplicable for DLs with a large amount of (multimedia) data. Furthermore, file systems support only limited capability for indexing and searching data. Therefore, in DL2Go, we decide to use a RDBMS as the underlying data management system. Most commercial RDBMS can optimize space and fragmentation issues by using sophisticated schemes, provide a standardized way to query, and is equipped with efficient query optimization techniques. In particular, in building prototypes, we use the MySQL open-source RDBMS, modified to run on flash devices.

In addition, to address the issue of large-size DLs further, one can also use various *compression* techniques, including database compression [5], PDF file compression [18], and general-purpose file compression [25,19]. To select one among the compression techniques above, we carry out preliminary experiment. We have tested 1,000 PDF files (247 MB) randomly selected from arXiv

Table 1. Experimental result for PDF file compression

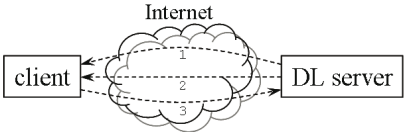
	PDF compression			General purpose compression		
Tool	Verypdf	CVista	Magic	WinZip	WinRK	7Zip
Compression ratio	0.99	1.01	0.98	1.18	1.23	1.24
Avg cmp time (in sec.)	0.48	4.77	1.25	0.39	0.67	0.37

`gr-qc` with three PDF file compression tools (Verypdf Advanced PDF Tools 2.0 [22], CVista PdfCompressor 4.0 [7] and Magic PDF compressor 2.2 [17]) and three general purpose compression tools (WinZip Pro 11.1 [25], WinRK 3.0.3b Normal compression [16] and 7-Zip 4.57 [19]). Table 1 presents the compression ratio and the average compression time in second for each tool, where $\textit{the compression ratio} = \frac{\textit{the uncompressed size}}{\textit{the compressed size}}$. In our experiment, PDF compression tools did not achieved better results than general purpose compression tools (even two of them increased the total size). With this result, we believe that the PDF compression tools are not appropriate for research papers. 7-Zip finally showed the best performance (the compression ratio is 1.24—it has saved 19.54% for space). Therefore, in our prototypes, we used 7-ZIP as the compression method. We leave tighter and more efficient integration of the compression scheme in the data management layer of DL2Go to the future work.

2.3 Editable User Interface

The role of users in DLs has changed from the passive read-only users to active participatory ones of Web 2.0 era. These days, many DLs not only allow users to customize the look and feel of interfaces, but also encourage users' input such as commenting on contents, correcting errors, and adding missing information. Moreover, due to its pervasive usage, most DLs are designed to have web interface of its contents. That is, using hyperlinks and images on web pages, users can search and browse the contents of DLs. To accommodate such needs, via the *wiki* system, DL2Go is designed to have *editable* web interface to all contents of DLs. As stated[3] as: "A wiki is software that allows users to create, edit, and link web pages easily. Wikis are often used to create collaborative websites and to power community websites.", the wiki system is primarily used to create and edit web pages for multi-user web environment. In DL2Go, however, we adapt this wiki system as the user interface of DL2Go running on flash devices for mainly personal DL users. However, when needs arise, DL2Go can still handle multi-user collaborative tasks. For instance, when multiple computers share one server machine or storage device in a underdeveloped country, one can plug DL2Go into a USB port of the server and do collaborative editing on some topics. In addition, we take advantage of built-in index and search capability of the modern wiki system (although we allow direct manipulation of stored data objects in the underlying RDBMS).

[3] `http://en.wikipedia.org/wiki/Wiki`




```
1. file downloadCurrentVersion()
2. file checkout(date lastUpdate)
3. boolean checkin(data newInfo)
```

Fig. 2. Interface with a DL server by web services

2.4 Interface with Server

Our DL2Go requires two kinds of interactions with the DL server; (1) *Downward interface*: Ideally, the server provides standard-based interfaces for users to download data – e.g., OAI-PMH[4] for metadata harvesting, OAI-ORE[5] for exchanging information about digital objects in DLs, RSS-based update syndication, and FTP-based direct download; and (2) *Upward interface*: When users make updates on contents in DL2Go and need to upload them to the server (for archiving or sharing), updated information can be uploaded to the server and merged with the rest of contents uniformly. Such a case can occur when an archeologist comes back from the field trip with full of comments recorded in her DL2Go based DL, and wants to share her updated DL contents with other archeologists. Although the current support of DLs for downward interface is abundant, we believe that one for upward interface is rather limited. To address this issue, in the design of DL2Go, we propose to use the framework of web services [23] to implement the upward channel.

Web services are software systems designed to support machine-to-machine interoperation over the Internet. By using the standard XML-based web services framework, we can have a programmable interface to a DL server so that not only human users but also software agents can build DL2Go easily. Figure 2 illustrates an example for the interface with a DL server by web services, which we propose. For the interface, three web service operations are provided as follows: `downloadCurrentVersion` allows clients to download the current snapshot of the DL, and by `checkout` operation the client is able to obtain new data since the last update. Moreover, `checkin` operation provides the upward interface by which users can upload the information they want to share.

3 Case Studies

To demonstrate the soundness of our proposed idea, we have built two prototypes of DL2Go based DLs in the domain of bibliographic DLs—"DL2Go for arXiv" and "DL2Go for DBLP." Table 2 shows the detailed statistics of two DLs that we used. The subset of arXiv [6] data set we used includes the information about 78,146 papers and 35,803 authors in Physics discipline. Often, the venue information of papers is missing in the arXiv data set. It contains PDF files for

[4] `http://www.openarchives.org/pmh/`
[5] `http://www.openarchives.org/ore/`

Table 2. Statistics of two data sets

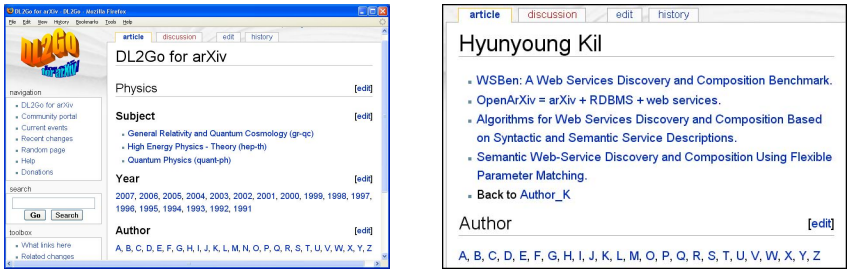
DL	papers	authors	conf./jour.	tuples in RDBMS	total size (in MB)
arXiv	78,146	35,803	N/A	114,729	234
DBLP	925,494	569,227	9,331/41,808	1,558,625	2007

most papers. On the other hand, the DBLP [15] data set includes the information about 925,494 papers (580,509 conference papers and 344,985 journal papers) and 569,227 authors in mainly Computer Science discipline. Among venues, there are 9,331 conferences and 41,808 journals. We have separately counted each year of conferences and each volume(number) of journals. Note that DBLP data set contains only bibliographic metadata without actual PDF or PS files. As the software and hardware systems, we have used the WOS Portable[4] package (that includes Apache 2.2.4, MySQL 5.0.41, and PHP 5.2.3), MediaWiki 1.10.1, and a USB flash drive with 4GB space. All features, except the upward channel using web services, are currently implemented in the prototypes.

3.1 DL2Go for arXiv

The arXiv [6] is one of the most popular scientific DLs in the fields of physics, mathematics, computer science, and so on. Although the number of archived publications in the arXiv is smaller than that of other scientific DLs, since it is self-archived by authors, the quality of associated metadata in the arXiv is excellent. In addition, since it often contains publication files in various formats (e.g., PS, PDF, DOC, TeX), the amount of the required storage for data and metadata is substantial. Since the arXiv does not provide the snapshot of their collection for download, we use the subset of arXiv data downloaded from a repository by Karczmarek [11]. This subset is created to be useful for anyone who wants to have a local copy of the arXiv, and contains the collections of the PDF-formatted papers and index files in three fields of Physics (i.e., General Relativity and Quantum Cosmology (`gr-qc`), Quantum Physics (`quant-ph`) and High Energy Physics - Theory (`hep-th`)) from 1991 to 2007.

By these index files, we reorganized the paper metadata, and created the main wiki page of DL2Go for arXiv (see Figure 3(a)) where users could browse through papers based on fields, years, and authors, mimicking the interface of the original arXiv web site. Although stored as tuples of various tables in the underlying RDBMS, conceptually, DL2Go for arXiv is comprised of a hierarchy of wiki pages. There is a wiki page for each paper at the bottom of the hierarchy while various kinds of intermediate internal paths in the hierarchy are also mapped to wiki pages. By following different internal paths, for instance, users can browse papers based on fields (i.e., `gr-qc`, `quant-ph` and `hep-th`), year (i.e., 1991-2007) or author names. An example of a wiki page for an internal path on author names is shown in Figure 3(b) whose role is to guide users to the correct wiki page. When a field is selected on the main wiki page, users need to choose a specific year



(a) Main page (b) Internal path page on author

Fig. 3. Screen-shots of "DL2Go for arXiv"

and month. Then, the wiki page containing the paper list of the field submitted in the period shows up. Through the path by year, DL2Go displays the paper list submitted in the month of the year selected, regardless of fields. Through the path by author, users can find out author names in alphabetic order, and finally we can obtain the list of papers written by the author.

A wiki page for each paper at the bottom of the hierarchy includes metadata of the paper (e.g., title, authors, subject, abstract, etc.), and links to the paper file and to user's comment (see Figure 4). Through this page, users can read the paper using a PDF viewer, edit the metadata of the paper information (e.g., correcting a typo in author names), add comments, and search other papers by keywords. Note that searching by keywords is available in any wiki pages. In addition to (or in stead of) the PDF versions of paper files, DL2Go may also provide text files, which contain the contents extracted from the original PDF paper files, for space and user flexibility. In general, the size of text files is much smaller than that of the original PDF files—e.g., 79,673 text and PDF files consume about 2.8GB and 17.5GB, respectively. In DL2Go, users can choose which versions of files to package, according to their preference.

One of unique features of DL2Go is that users can leave their comments for any papers (or any wiki pages). While reading a paper, for instance, users can add their own review for the paper in the corresponding wiki page. In addition to user's comments, every data including paper context and metadata in our DL2Go are editable by users. While we read papers or books, it is not unusual to find out wrong or missing data as well as a simple typo, but it is not easy for readers to correct data and share it with other users in conventional DLs. Since our DL2Go adopts the wiki system as a user interface, users can edit all pages easily. In addition, through the server interface, the updated information can be reported to DL servers easily.

3.2 DL2Go for DBLP

The DBLP (Digital Bibliography & Library Project) [15] provides bibliographic information on major Computer Science journals and proceedings. Our DL2Go

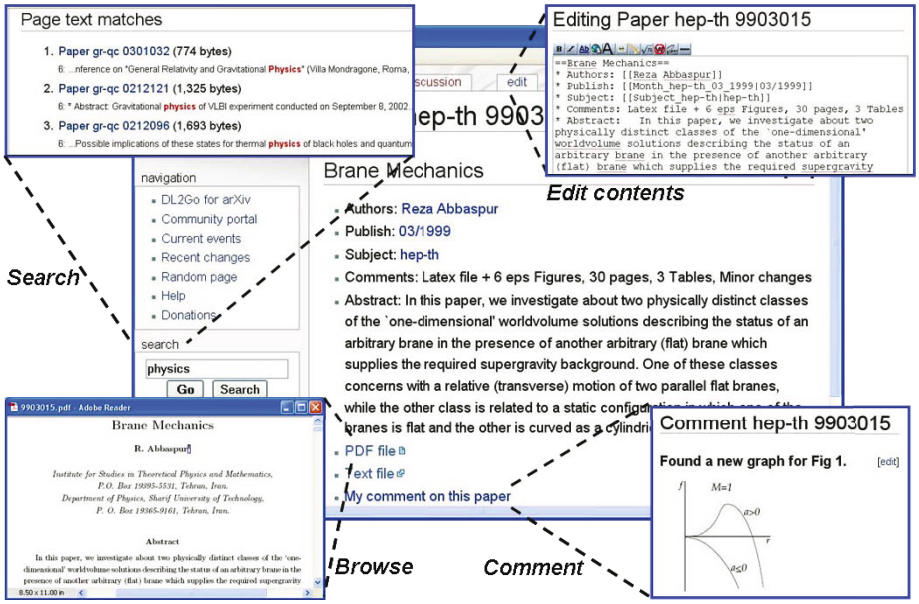
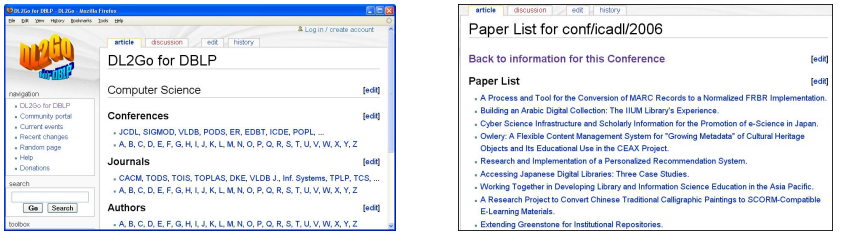


Fig. 4. Functions on paper page of "DL2Go for arXiv"

for DBLP is built with all metadata for conferences, conference papers, journal papers in XML files downloaded from the DBLP web site [15]. Similar to the implementation of DL2Go for arXiv, we reorganize the parsed data and generate wiki pages for the main, three internal paths (i.e., by conferences, journals, and authors), and papers. Figure 5 illustrates screen-shots for the main page and an internal path on conference. On the main page (Figure 5(a)), a paper can be easily searched by conference, journal, and author names. When users choose a conference/journal name and a specific year/volume(number), a wiki page for the selected conference/journal in that year/volume is displayed. The page includes the detailed conference/journal information (e.g., its full name, the location and date, the homepage URL) and a link to the list of the papers published in the conference/journal (see Figure 5(b)). Then, users can browse a page containing the paper list via the link, and finally obtain a paper page from the list. As DL2Go for arXiv, users are able to find author names in alphabetic order, and obtain the list of papers written by the author. A wiki page for each paper includes its title, authors, conference/journal information, and links to a electronic file for the paper and to user's comment. Since DBLP does not provide PDF nor PS files for papers, DL2Go for DBLP does not support them either. However, DL2Go has a link to a paper file as a blank so that users can add the files once they obtain files from other resources. Every data in DL2Go for DBLP is editable as DL2Go for arXiv.



(a) Main page (b) Internal path page on conference

Fig. 5. Screen-shots of "DL2Go for DBLP"

4 Related Work

Several personal DL systems are related on our DL2Go framework. Both of [26] and [13] are open-source projects for extensible digital library platforms. The former project provides a new way to build digital library collections and to publish them on the Internet or CD-ROMs, and the latter aims to create flexible, collaborative digital libraries based on the previous Fedora project [14]. In the Personal Libraries system of Berkeley Digital Library [24] which is largely premised on a distinction between collection (a specification of some set of documents) and repository (a means of storing and retrieving individual documents), users can create their own collections which are built from materials extracted from a very large document collection. The UpLib system [10] is specially designed for secure use by a single individual and is extended for collaborative operation of multiple UpLib repositories [9]. In the Salticus system [3], each user can build a personal digital library by collecting documents and generalizing user's choice based on a user's interest. They, however, do not support editable user interfaces and/or consider portability issues. [12] discusses user interface issues to browse, edit, store, and annotate DL resources but do not consider the wiki system.

A few recent studies [2,8,20,21,1] consider portability issue for DLs. Some [2,8] of them present DL systems where users can access through wireless network and/or mobile phones. [2] explores how users can be given access to digital information while they are mobile. Mobile G-Portal study [21] shows that a group of mobile devices can be used as learning assistant tools to support collaborative sharing and learning for geography fieldwork. [20] has developed indexing technology to search Wikipedia on a conventional CD. Bainbridge et al. have built a self-contained digital library on an iPod [1].

A number of efforts [18,19] are made to save storage space for (PDF document) data files and database compression [5]. MaximumCompression [18] has studied PDF file compression, and compressed 1 file (4,526,946 bytes) with a number of different compressors/archivers (195). The result is that WinRK [16] performs best with 21.60% saving for space. In addition, we have experimented on a huge number of PDF files (1,000 files, 247 MB) using several compression tools. 7-Zip [19] shows the best performance with 19.54% saving.

5 Conclusion and Future Work

The novel framework, termed as DL2Go, that supports *editable* and *portable* DLs for nomadic users on the road is presented. Since the whole DL2Go system and the contents of a DL are stored in a tiny "in-the-pocket" flash-based storage device, our proposal can be useful for users who have to work in an offline environment where the network access is scarce and storage space is limited (e.g., scientists in the remote fields).

Ample directions are ahead for future work. We are working on implementing an efficient upward interface to the DL server using Web services. We believe that this interface deserves further exploration for interactive DLs. For data (de)compression, we need to build complete integration of the compression scheme in the data management layer of DL2Go.

References

1. Bainbridge, D., Jones, S., McIntosh, S., Jones, M., Witten, I.H.: Portable digital libraries on an iPod. In: ACM/IEEE Joint Conference on Digital Libraries (JCDL), pp. 333–336 (2008)
2. Bhargava, B., Annamalai, M., Pitoura, E.: Digital library services in mobile computing. ACM SIGMOD Record 24(4), 34–39 (1995)
3. Burke, R.D.: Salticus: guided crawling for personal digital libraries. In: ACM/IEEE Joint Conference on Digital Libraries (JCDL), pp. 88–89 (2001)
4. Software, C.H.: Webserver On Stick, `http://www.chsoftware.net/`
5. Chen, S., Nucci, A.: Nonuniform compression in databases with haar wavelet. In: Data Compression Conference (DCC), pp. 223–232 (2007)
6. Cornell. arXiv.org, `http://www.arxiv.org/`
7. CVISION Technologies, Inc. Cvista pdfcompressor 4.0., `http://www.cvisiontech.com/`
8. Imai, S., Kanamori, Y., Shuto, N.: Retrieving tsunami digital library by use of mobile phones. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 525–528. Springer, Heidelberg (2007)
9. Janssen, W.C.: Collaborative extensions for the uplib system. In: JCDL, pp. 239–240 (2004)
10. Janssen, W.C., Popat, K.: UpLib: A universal personal digital library system. In: ACM symposium on Document Engineering, pp. 234–242 (2003)
11. Karczmarek, J.: The arXiv on Your Harddrive, `http://www.theory.physics.ubc.ca/arxiv/`
12. Kikuchi, H., Mishina, Y., Ashizawa, M., Yamazaki, N., Fujisawa, H.: User interface for a digital library to support construction of a virtual personal library. In: International Conference on Multimedia Computing and Systems (ICMCS), pp. 429–432 (1996)
13. Krafft, D.B., Birkland, A., Cramer, E.J.: Ncore: Architecture and implementation of a flexible, collaborative digital library. In: JCDL, pp. 313–322 (2008)
14. Lagoze, C., Payette, S., Wilper, C.: Fedora: An architecture for complex objects and their relationships. International Journal of Digital Libraries 6(2), 124–138 (2006)

15. Ley, M.: DBLP: Digital Bibliography & Library Project, http://dblp.uni-trier.de/
16. Software Ltd, M.: WinRK., http://www.msoftware.co.nz/WinRK_about.php
17. Magicteck Software. Magic pdf compressor 2.2, http://www.magicteck.com/
18. MaximumCompression. Adobe Acrobat document PDF file compression test, http://www.maximumcompression.com/data/pdf.php
19. Pavlov, I.: 7-Zip, http://www.7-zip.org/
20. Potthast, M.: Wikipedia in the pocket - indexing technology for near-duplicate detection and high similarity search. In: ACM SIGIR, p. 909 (2007)
21. Theng, Y., Tan, K., Lim, E., Zhang, J., Goh, D., Chatterjea, K., Chang, C., Sun, A., Yu, H., Dang, N., Li, Y., Vo, M.: Mobile g-portal supporting collaborative sharing and learning in geography fieldwork: an empirical study. In: JCDL, pp. 462–471 (2007)
22. VeryPDF.com, Inc. Advanced pdf tools v2.0, http://www.verypdf.com/
23. W3C. Web Services, http://www.w3.org/2002/ws/
24. Wilensky, R.: Personal libraries: Collection management as a tool for lightweight personal and group document management. Technical Report SDSC TR-2001-9, San Diego Supercomputer Center (2001)
25. WinZip International, LLC: WinZip, http://www.winzip.com/
26. Witten, I.H., Boddie, S.J., Bainbridge, D., McNab, R.J.: Greenstone: A comprehensive open-source digital library software system. In: ACM International Conference on Digital Libraries, pp. 113–121 (2000)