Integrating PDF interface into Java application

Abstract

Purpose – The purpose of this paper is to propose a novel approach to integrate PDF interface into Java-based digital library application. It bridges the gap between conducting content operation and viewing on PDF document asynchronously.
Design/methodology/approach – In this paper, we firstly review some related research and discuss PDF and its drawbacks. Next, we propose the design steps and implementation of three modes of displaying PDF document: PDF display, image display and XML display. A comparison of these three modes has been carried out.
Findings – We find that the PDF display is able to completely present the original PDF document contents and thus obviously superior to the other two displays. In addition, the format specification of PDF-based e-book does not perform well; lack of standardization and complex structure is exposed to the publication.
Practical implications – The proposed approach makes viewing the PDF documents more convenient and effective, and can be used to retrieve and visualize the PDF documents and to support the personalized function customization of PDF in the digital library applications.
Originality/value – This paper proposes a novel approach to solve the problem between content operation and the view of PDF synchronously, providing users a new tool to retrieve and reuse the PDF documents. It contributes to improve the service specification and policy of viewing the PDF for digital library. Besides, the personalized interface and public index make further development and application more feasible.

Keywords Portable document format (PDF), Java, User experience, Integrated interface, Index mechanism, Digital library
Article Classification Technical paper

Introduction

Nowadays, there is a large number of PDF documents in digital libraries and full-text databases, such as e-books, e-journal and other relative files (Adobe Systems Inc., 2012). They are increasingly popular and significant to libraries (Nelson, 2008). The PDF documents carry a combination of information in various medium formats, such as text, image, font, color, symbol and shapes, etc., which bring the readers an unprecedented reading experience. Compared with the media formats such as TXT, HTML and XML, PDF has the advantage of making describing and printing a document easier (Adobe Systems Inc., 2012).

However, the PDF has its inherent defects. Most PDF documents do not have basic high-level document logical structural information, which makes the retrieve and reuse of the documents difficult. Moreover, Wang pointed out the drawbacks of PDF (Shaofeng, 2004), including:

- Lacking of a powerful search engine to deal with the structure and content of the document.
- Being difficult for users to have different access privileges on different parts of the document.
- Being difficult to provide personalized interface for users.

All of these defects will greatly limit the access to PDF and its circulation in the digital library. Though there are many readers and analysis tools about PDF in reality, they are always either displaying the PDF document alone or analysis of the PDF text content. For example, Adobe Reader, Foxit Reader, etc., can display the PDF document completely but don't have the function of analyzing the text content on PDF. Like the concordance (Watt, 2012), TextArc (Paley, 2002), FeatureLens (Don *et al*., 2012), ProfileSkim (Harper *et al*., 2006) and iSee (Sun *et al*., 2008), though they have a strong ability of text analysis, they are not good document readers. At present, if we want to view and analyze the content of PDF documents, we will have to abstract the PDF text content in a document reader and put it into a analysis tool. It not only decreases the effectiveness of reading and learning, but also affects the efficiency of search and retrieve. Veronica *et al* once compared searching in realistic books (Liesaputra and Witten, 2012) with searching in PDF files, the result demonstrated that the searching in PDF files couldn't satisfy the users (Liesaputra *et al.*, 2009). The users found it hard to understand the structure of the PDF document, not knowing where they were since they are easily disoriented and finding it difficult to return to a specific location (Liesaputra *et al*., 2009).

Therefore, we propose a novel approach to solve this disjunction problem between content operation and the view of PDF synchronously, namely integrating PDF interface into Java application. First, we analyze the current relevant research and applications about the PDF. Based on current research results, we design three modes to integrate the PDF document content into Java panel. After comparing the three experimental results, we demonstrate that integrating the PDF interface into Java application can completely display the PDF content and improve the view and search experience on PDF documents.

Related research

The portable document format (PDF) is widely accepted as a digital archiving format and PDF documents are supported in virtually every repository (Seadle, 2009). Its platform-independent and openly accessible feature makes it the ideal file format to release and disseminate electronic documents in the digital library (Adobe Systems Inc., 2012). Though some discussions on whether PDF formats are appropriate for long-term digital archiving (Seadle, 2009; Zhao, 2011; Uneson, 2005), there is no doubt that the digital library has become the main aggregation of the PDF (Vasileiou *et al.*, 2009). Several major publishers, like Elsevier, Emerald, Wiley and Springer, provide large numbers of academic e-books and e-journal based on the PDF format. Their target market mainly focuses on the libraries - academic, public and special (Vasileiou *et al*., 2009). Therefore, the electronic books have a more significant potential to impact users' reading experience (Liesaputra *et al*., 2009), with important consequences for the future role and existence of libraries (Vasileiou *et al*., 2009).

At present there are many types of e-book format in the market. Vasileiou *et al*. analyzed the main websites of nine e-book publishers and eleven e-book aggregators and found that the most common format of e-books appears to be PDF. While the majority of vendors use PDF, a couple of companies provide their content in HTML, such as Knovel and Gale. However, Ebrary uses its own EDF format and Questia uses XML (Vasileiou *et al*., 2009). In addition, Thomas *et al*. built the JSTOR system to scan the documents and convert them to the images. Although this method can create the content identical to the original document, its generated images lose the textual context in the original document and fail to search the full text in the generated image files (Thomas *et al.*, 1999). Patrick van *et al*. developed i*Doc that is based on the Extensible markup language (XML) which can serve as integration format and deliver the personalized contents (Patrick van *et al.*, 2000).

In order to explore the differences among the commonly used electronic formats, many researchers try to distinguish these formats from different perspective. Shaofeng indicated that though HTML was designed to describe the structure of the document and thus concerned the appearance more than the content of the document, it has weakness in retrieving and printing the electronic document as well as other problems (Shaofeng, 2004). He also made a comparison between PDF and XML, the result shows that though the PDF has the advantage of describing and printing what a document looks like, the non-structure and access privilege control make it hard to be searched and reuse (Shaofeng, 2004). Furthermore, users criticized that all the clients have the same interface. By contrast, the XML can achieve efficient retrieval and personalized customization by the third-party software (Shaofeng, 2004). Zhao and Uneson then compared the PDF with XML from a perspective of long-term preservation (Zhao, 2011; Uneson, 2005). The results demonstrate that the PDF is not the best choice for retrieving and reusing, and the XML does not have the absolute advantage of fully preserving the appearance of an electronic document.

All in all, the researchers found the main drawbacks of PDF are about supporting search and extraction function, and began to look for new ideas of extracting PDF's

content and structure to achieve the goal of retrieving and reusing the PDF. Numerous studies have shown that the structured content is more effective to obtain information accurately and quickly. There is need for extracting objects in a structured form and saving the document in XML format in order to allow indexing, more accurately searching, and dealing with versioning and Meta data. Initially Hadjar *et al*. proposed a new tool to extract text, images, and graphics from a PDF document, but did not consider the hidden layout and logical structures of documents (Hadjar *et al.*, 2004). Consequentially, Chao and Fan made a further improvement, developing techniques that identified logical components on a PDF document page. The outlines, style attributes and the contents of the logical components were extracted and expressed in an XML format (Chao and Fan, 2004). Still someone try to convert the PDF to other formats. For example, Rahman and Alam converted the PDF into HTML (Rahman and Alam, 2003), Déjean and Meunier converted PDF document into structured XML format (Déjean and Meunier, 2006), and Zhang converted PDF files to XML files (Zhang, 2008) and so on. Unfortunately, these conversion and extraction lost the original appearance of PDF files, such as font, image and paragraph information or others. But these techniques do facilitate the retrieve and reuse of the layout and the content of a PDF document page.

Research shows that the complete display of original document information contributes to faster learning and retention, and thus leads to more effectively searching and higher satisfaction (Ahmed *et al.*, 2006). However, one of the significant challenges with PDF accessibility is the lacking of effective ways to unify content operation and the view on PDF synchronously. Thereby, it is difficult to design the user interface that embeds the PDF document into applications. Though several applications are now available, they are just online service, such as *Scribd* for viewing and storing, *Pdfvue* for online editing, and *Zamzar* for PDF Conversion.

The preparation for integration

To enable the user to interact with the PDF document friendly, we should consider the user's demands before designing the interface:

- The contents of PDF document should be displayed completely, including the picture, graph, font and others;
- Operation and view should be in the same interface synchronously, avoiding the shifting among multiple interfaces;
- User could integrate suitable application into the interface to aid them analyze the contents of PDF document;
- The application could be run on every platform and suitable for e-book based on the PDF format.

Based on these requirements, we develop an application that provides users with the following functions:

- Allowing the user to view multiple PDF documents via the interface;
- Achieving the synchrony between content operation and the view of PDF ;

- Providing the index file for the developers to enable the further development and then more applications could be integrated into the interface.

Therefore, we propose a new tool to integrate the PDF interface into Java application. The framework of its design is presented in Figure 1. We design our tool in the following steps.

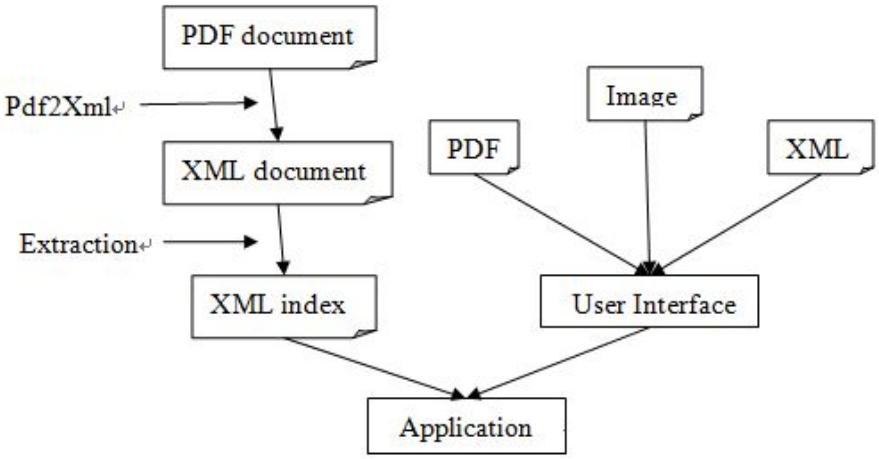

Figure 1.The framework of design

1).The PDF converting. We find it difficult to display the complete original information of PDF document through calling the APIs directly provided by the Adobe, which may lost the paragraph information. Therefore, we should first convert the PDF format to other formats. Considering the fact that XML (W3C, 2012), a very flexible file format, is widely used in information storage, information interchange and other aspects (Moghrabi *et al.*, 2004), and can do what Java has done for programs. There are also a number of research achievements on XML application (Luk *et al.*, 2002; Chu *et al.*, 2000; Zhang *et al.*, 2008; Geroimenko and Geroimenko, 2001; Lu *et al.*, 2008). These advantages prompt us to choose it as the converted object. The specific process can be found in a paper by Zhang (Zhang,2008).

2).The index construction. That what kind of index should be constructed will affect the performance of retrieve. Moskovitch *et al*. made a comparative evaluation of full-text, concept-based and context-sensitive search; they demonstrated usefulness of concept-based and context-sensitive queries for enhancing the precision of retrieval from a digital library of semi-structured clinical guideline documents (Moskovitch *et al.*, 2007). Jimmy's results suggest that the highest overall effectiveness may be achieved by combining evidence from spans and full articles (Lin, 2009). Then we constructed a XML index mechanism from sentence to page. The index is mainly used for the synchronous mechanism. We could make a further application development with the aid of it. Through the converted XML file, we could build the index. The specific process will be revealed in a follow-up paper. The specific index mechanism will be illustrated in next section.

3).Interface design. In this step, we develop a specific Java application to contain the PDF document. We choose Java as our development language for it is platform-independent and can run on any computers that have the Java's runtime environment, or virtual machine. The combining XML with Java keeps its

cross-platform features effectively. Another reason is that Java provides the needed technologies for XML. For example, JAXP allows integration of any XML parser with a Java application in order to read, manipulate and generate XML documents. The specific interface layout is as follow:

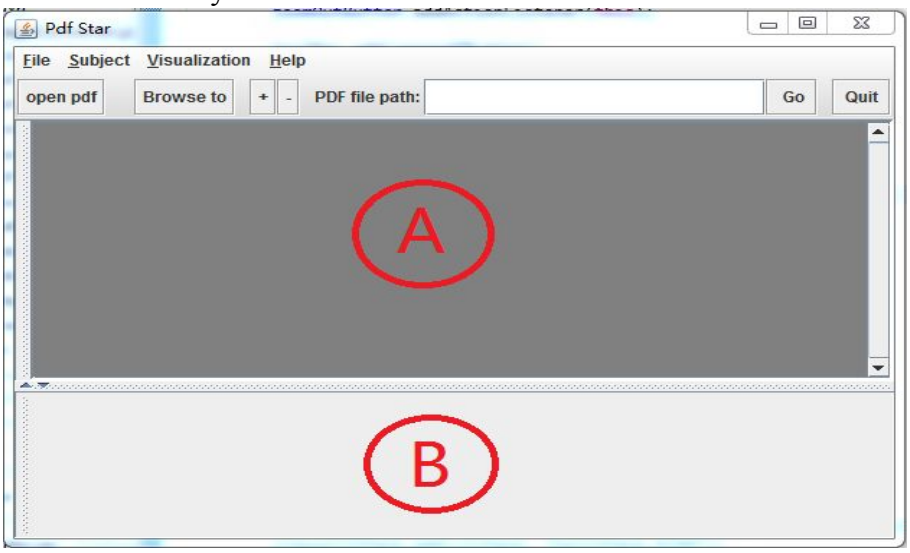

Figure 2.user interface (the area of A contain the PDF document content, the area of B could be integrated application)

4). The PDF integration. In the above relative researches, we mention some different ways to view an electronic document. In order to clearly identify what kind of interface display is more suitable for users, we design three modes. The first one contains the PDF format, the second one contains the image format and the third one contains the XML format. Through the mutual comparison, we can easily find what interface display satisfies user better.

The implementation procedure

We choose a general e-book, containing the table of contents, body and pagination, based on the PDF format as a sample, and then elaborate the implementation of three modes from the following aspects: how to embed and how to synchronize

How to embed the PDF document content into the interface

After conversion and extraction of the e-book's information, the designed tool has the ability to integrate the e-book into the Java panel. The designed tool provides a friendly and reliable way to browse the e-book's information. User could browse the e-book by the following three ways.

1) Embedding the XML into the interface.
   - First, we need the XML file derived from the converted e-book through the conversion method (Zhang, 2008).
   - Second, we exhibit the XML information on the Textfield component added into the area of A through the Java API, namely *dom4j.jar*. This library could read the XML information and print it on the screen.

- Third, in order to visit the XML information conveniently, we design a content tree that can jump to different position in this development. You can design other functions to visit the XML information.

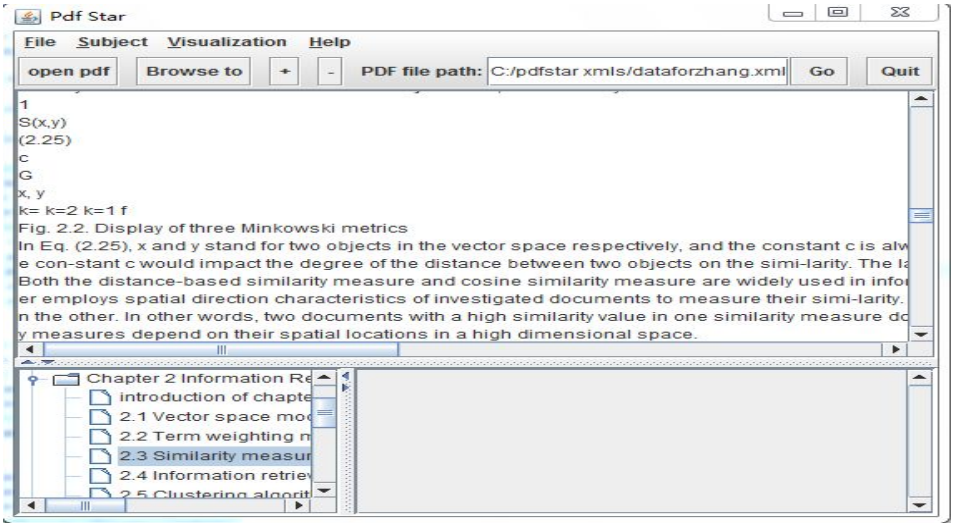

Figure 3.XML display

2) Embedding the image into the interface.

- First, we need to convert the e-book to images through the *pdfview.jar* provided by the sun company. We mainly use the class of *PDFFile.java* and *PDFPage.java* drawing the e-book page.
- Second, we add a Label component into the area of A and use the Label component to exhibit the converted image. We can scan every image according to the selected page number or content information.

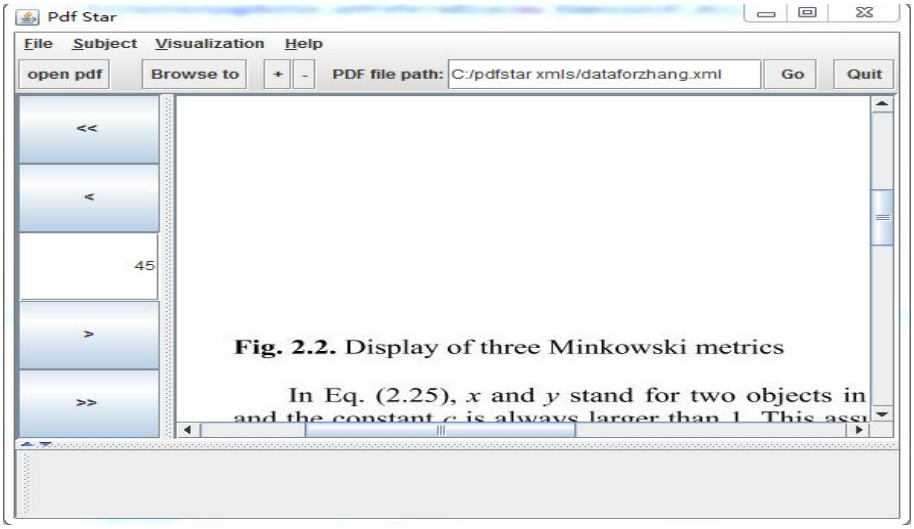

Figure 4.Image display

3) Embedding the PDF into the interface.

- This approach mainly uses the *ICEpdf.jar* to exhibit the e-book on the area of A. We add a panel component into the area of A and put the e-book into it. Through this way, we can easily scan the original PDF format. Besides, this java package also provides us many useful tools. The mainly embedded code are as follow:

```
SwingController pdfcontroller = new SwingController();
SwingViewBuilder factory = new SwingViewBuilder(pdfcontroller);
Jpanel viewConponentPanel = factory.buildViewerPanel();
pdfcontroller.openDocument(pdfUrl);
```

Among them, *pdfcontroller*, *factory* and *viewConponentPanel* represent the Java classes, *pdfUrl* represent the e-book save path. The interface is as follow:

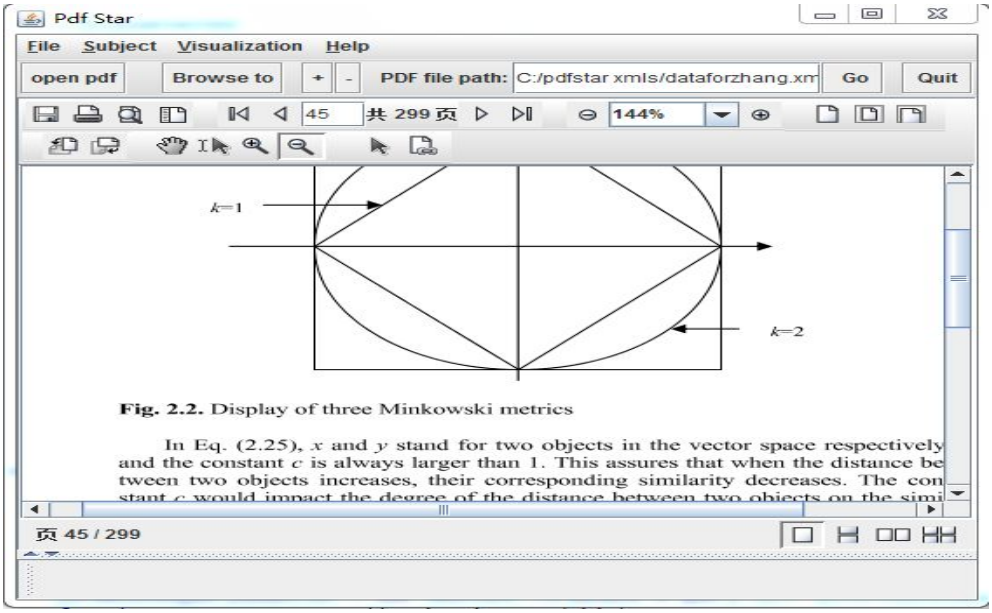

Figure 5.PDF display

How to accomplish the synchronous mechanism

Our research goal is more than browse the e-book content information. More importantly, we provide the synchronous mechanism to operate the e-book. We can not only exhibit the e-book information, but also support the user API to integrate more applications in this interface. User operates the XML index instead of the PDF. Of course, all of these benefits are from the establishment of the index.

The index mechanism: we construct three-level index in the XML file. First, the structure of index is like this, a root node as the catalog and its sub-note is the chapter; similarly, the sub-note is the section and the lowest level is the sentence. We could obtain the node information from the converted e-book, namely the XML file. According to the contents included in the XML file, we could establish tree-based XML index from chapters, sections to logical page number. However, it needs further processing to obtain the node of sentence information. Second, through scanning the XML file, we segment the e-book on the basis of sentence and add the sentence nodes into the section's child note. Third, in order to exactly locate where the sentence occurs, we add some attribute into the node. The main goal is to achieve the correspondence between the physical page and the logical page. The final index is as follow:

```
<?xml version=”1.0” encoding=”utf-8”?>
<catalog>
  <chapter chaptername=”chapter0” startp=”1” chaptercount=”1” endp=”12”>
     <section sectionname=”section1” startp=”2” fullseccount=”2” endp=”4”>
          <sentence startp=”2” fullcount=”60”> this is an information age. </sentence>
                        …
      </section>
         …
  </chapter>
     …
</catalog>
```

In above index, we label the start logical page and end logical page in the chapter and section, and also set their chapter and section name and count the number of chapters and sections in the entire e-book. In addition, we use a node to store the sentences in every section. After these settings being completed, we could display and find the information synchronously.

1) The synchronous mechanism of XML.

Because the index derived from the XML file, we only find the node that you select and print the information on the screen.

2) The synchronous mechanism of image.

Because many e-books' contents do not start from the page number one, there is an inconsistency between the logical page and the physical page. To solve this problem, we should record the physical page numbers before the article body starts if the article body starts from the logical page one. We call this page numbers "distance". This phenomenon is also suitable for the synchronous mechanism of PDF.

When finding a sentence, we first find the logical page number from the index. But this is not the eventual pagination, we need to use this numerical value to add the "distance", then the eventual value is the physical page numbers where the sentence occurs. Next, we find the image by the sequences and return it to the Label.

3) The synchronous mechanism of PDF.

Because of the "distance", we can't direct to the page according to the logical page number derived from the index. We should add the logical page number to the "distance", and then obtain the eventual pagination. Next, we could call the function that *ICEpdf.jar* has to direct to the page.

Results analysis and discussion

Through three modes' implementation, we could present the content of PDF document by three different formats. However, their performance differs. We can find something among them, as shown in Table 1:

Through table 1, we can clearly find that the PDF display is obviously superior to the other two in the integrity display of the PDF document content. It can almost view the PDF document content completely. For example, a graph can be presented completely in PDF display, but it is none in the other two, as shown in Figure 3, Figure 4 and Figure 5. In terms of efficiency, despite the poor performance of XML on display, but it runs the fastest. When meeting the PDF documents that contain almost entirely of simple text, the XML display may be a good choice, like the document of literature, history and philosophy and so on. But if you view an album of art, the image display may be a better choice. We could choose different modes to view a PDF document according to the real problems. Of course, we can also identify their performance from the aspects of aesthetic feeling, usability and legibility. In short, embedding the PDF is an effective and suitable way to display the original PDF document content, and can improve the user experience.

The significant contribution of this study lies in integrating PDF interface into the Java application to achieve the goal of viewing and operating the PDF document synchronously. Based on this study, practical implications are discussed. It can improve the digital library's usability from four aspects.

- First, it can be used to retrieve and visualize the PDF document. Digital library provides a comprehensive collection of digital resources and services that are accessible through the Web. Every day there are large amounts of search behaviors. The search engine always provides a set of keywords for users, but the major search engines typically do not index the content of PDF documents at all (Lawrence *et al.*, 1999). Therefore, users can retrieve the XML index to achieve the goal of retrieving the PDF document. If possible, it will make for the digital library to construct an index library of PDF document content based on XML. In addition, we can visualize the PDF document information by the XML advantage of storage content, as shown in figure 6. This will greatly enhance user retrieval and reading efficiency. It could also be used in other areas such as the anti-plagiarism services (Patel *et al.*, 2011), e-commerce (Seng and Lai, 2010) and teaching (Carlock and Perry, 2008), etc.
- Second, it can be used to support the personalized function customization of PDF. Due to the limitation of security and privacy in PDF (Castiglione *et al.*, 2010), the available tools or software are very few and their functions are also limited. Users can not customize the required functionality according to their own preferences. Kani-Zabihi *et al*. have surveyed the digital library about what do users want and found that based on users' previous experiences with digital libraries, their requirements with respect to specific features may change (Kani-Zabihi *et al.*, 2006). Many researches emphasized the importance of user-centered design. Our study provides the interface and index for the researcher to make a further development. For example, users can integrate the Lucene into the interface to help them view and analyze the PDF full text content synchronously. Besides, users can also put the visual information of PDF full text content into the interface to aid them browse and view. As shown in Figure 6, in a related research, we use a series of concentric circles to visualize the full text, a certain

chapter, a certain section and a certain paragraph in one book. When a paragraph object in section 2.3 is clicked in the visualization area, the corresponding text and its context can be shown synchronously.

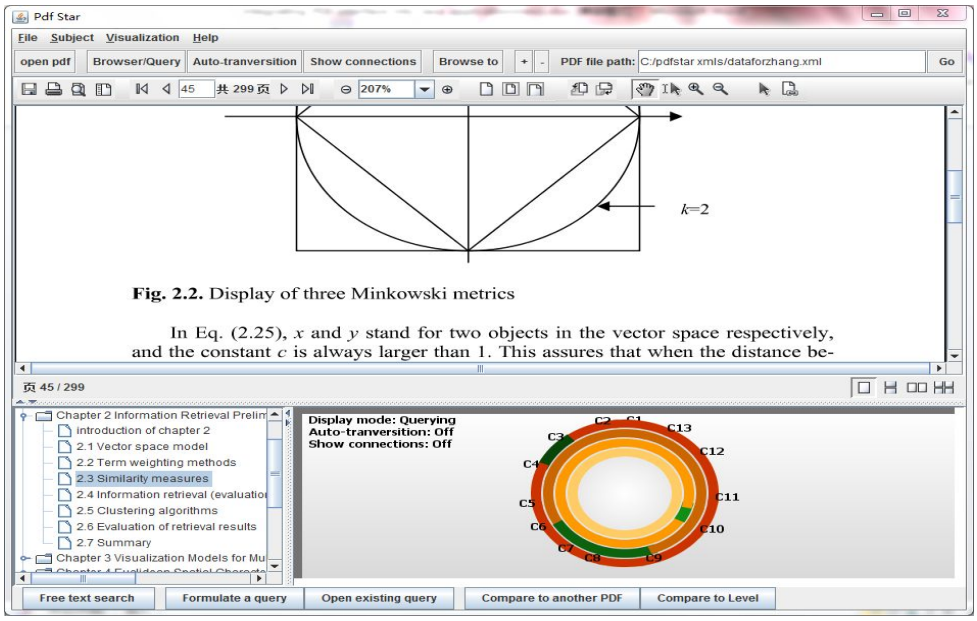

Figure 6.The visualization of PDF content

- Third, it enhances the user experience in digital library. Whether the field of computer science or information science, one of the focuses is always on the user experience and yet one of the most obvious factors is user interface (Marchionini and Komlodi, 1998; Bates, 2002; Ahmed et al., 2006; Davis and Price, 2006). User interface is the way through which a user can communicate with particular software. Berg *et al*. also indicated that the usability of a given interface should be a primary consideration when investigating e-book platforms (Berg *et al.*, 2010). And their experiments showed that the information-seeking and content-viewing that are not on the same page would greatly affect the user experience. Our study integrates the PDF interface into the java application. Users can synchronously view and operate the PDF document on the same page. The PDF display can completely exhibit the PDF document content and provide the navigation through browse, search, and indexes for user (Browne and Cue, 2012). It helps to decrease the user cognitive load of reading and learning.
- Fourth, it improves the service specification and policy of viewing the PDF in digital library. Most digital libraries always provide the way of metadata and downloading for viewing the PDF. And sometimes when viewing the PDF files in the browser, an additional browser plug-in is needed. This dependence on the software or plug-in makes it inconvenient to use the PDF because it is not easy for clients to download and install the additional software or plug-in. Generally, our study will help to solve the dilemma and develop a new standard of browsing the PDF without downloading. The aided analysis and synchronous display the PDF online will change the traditional way of viewing PDF, rather than requiring users to download and then view and analysis. Furthermore, the personalized

interface and public index make further development and application more feasible.

This study has some limitations. First, the sample might not be able to represent the general PDF document. Our survey finds that different publishers have different standard specifications. The different e-journals or e-books based on PDF in the same publisher still have different standard specifications. Even the same journals in the same publisher still have the different styles in different time periods. Especially, a special phenomenon occurs in publication, which is that the table of content sometimes cannot match the actual page number, which influences the structure of automatic extraction and the synchronic mechanism. Second, index of the PDF document mainly reflects the relationship between sentences and paginations. Its precision cannot reach the index of words. Third, we have not integrated useful tools into the interface. Further researches are discussed in the conclusion section to respond to the limitations of this study.

Conclusion and future work

There is a large amount of PDF documents existing in the digital library. Many technologies have been proposed and used on viewing or seeking PDF documents, such as conversion and extraction. This study proposes a novel approach to view and operate the PDF document synchronously in the same interface. Meanwhile, we demonstrate that the PDF display is apparently superior to the image display and the XML display in terms of the original exhibition and user experience. And it has potential advantages compared with other PDF readers in digital library. The main contributions of this study can be summarized as follows: (1) a novel approach to deal with the PDF document is recommended. (2) Integrating PDF interface into Java application to achieve the goal of viewing and operating the PDF document synchronously. (3) A new indexing mechanism has been designed and implemented, which is conducive to the further development. (4) Contributing to improve the service specification and police of viewing the PDF in digital library. Therefore, it could be regarded as a new way to bridge the gap between content extraction and original exhibition on the PDF document.

However, there are still lots of work to do in the future. Different ways of extraction and index should be provided for different styles of sample. In addition, it is worth to make the further application development in this integrated interface. Of course, it is still a key point to enhance the interface and its interaction with users.

References

Adobe Systems Inc. (2012), "Adobe Portable Document Format", available at: http://www.adobe.com/products/acrobat/adobepdf.html(accessed 5 March 2013).

Ahmed, S.Z., McKnight, C. and Oppenheim, C. (2006), "A user-centred design and evaluation of IR interfaces", *Journal of Librarianship and Information Science,* Vol. 38 No. 3, pp. 157-72.

Bates, M.J. (2002), "The cascade of interactions in the digital library interface", *Information Processing & Management,* Vol. 38 No. 3, pp. 381-400.

Berg, S.A., Hoffmann, K. and Dawson, D. (2010), "Not on the Same Page: Undergraduates' Information Retrieval in Electronic and Print Books", *Journal of Academic Librarianship,* Vol. 36 No. 6, pp. 518-25.

Browne, G. and Cue, M. (2012), "Ebook Navigation: Browse, Search and Index", *Australian Library Journal,* Vol. 61 No. 4, pp. 288-97.

Carlock, D.M. and Perry, A.M. (2008), "Exploring faculty experiences with e-books: A focus group", *Library Hi Tech,* Vol. 26 No. 2, pp. 244-54.

Castiglione, A., De Santis, A. and Soriente, C. (2010), "Security and privacy issues in the Portable Document Format", *Journal of Systems and Software,* Vol. 83 No. 10, pp. 1813-22.

Chao, H. and Fan, J. (2004), "Layout and content extraction for PDF documents", in Marinai, S. and Dengel, A. (Eds.), *Document Analysis Systems Vi, Proceedings*, Springer-Verlag Berlin, Berlin, pp. 213-24.

Chu, C.-H., Huang, C.-H. and Lee, M. (2000), "Building an XML-based unified user interface system under J2EE architecture", in *Multimedia Software Engineering, 2000. Proceedings. International Symposium on*, IEEE, pp. 208-14.

Déjean, H. and Meunier, J.-L. (2006), "A system for converting PDF documents into structured XML format", in *Document Analysis Systems VII*, Springer, pp. 129-40.

Davis, P.M. and Price, J.S. (2006), "eJournal interface can influence usage statistics: implications for libraries, publishers, and Project COUNTER", *Journal of the American Society for Information Science and Technology,* Vol. 57 No. 9, pp. 1243-48.

Don, A., Zheleva, E., Gregory, M., Tarkan, S., Auvil, L., Clement, T., Shneiderman, B. and Plaisant. (2012), "Exploring and Visualizing Frequent Patterns in Text Collections with FeatureLens", available at: http://www.cs.umd.edu/hcil/textvis/featurelens/(accessed 21 November 2012).

Geroimenko, V. and Geroimenko, L. (2001), "Visual interaction with XML metadata", in *Information Visualisation, 2001. Proceedings. Fifth International Conference on*, IEEE, pp. 539-45.

Hadjar, K., Rigamonti, M., Lalanne, D. and Ingold, R. (2004), "Xed: a new tool for extracting hidden structures from electronic documents", in *Document Image Analysis for Libraries, 2004. Proceedings. First International Workshop on*, IEEE, pp. 212-24.

Harper, D.J., Koychev, I., Sun, Y., Muresan, G. (2006), "ProfileSkim - An Intelligent Document Browser", available at: http://www.comp.rgu.ac.uk/staff/sy/msProf.htm(accessed 5 July 2006).

Kani-Zabihi, E., Ghinea, G. and Chen, S.Y. (2006), "Digital libraries: what do users want?", *Online Information Review,* Vol. 30 No. 4, pp. 395-412.

Lawrence, S., Bollacker, K. and Giles, C.L. (1999), "Indexing and retrieval of scientific literature", in *Proceedings of the Eighth International Conference on Information Knowledge Management, Cikm'99,* Assoc Computing Machinery, New York.

Liesaputra, V. and Witten, I.H. (2012), "Realistic electronic books", *International Journal of Human-Computer Studies,* Vol. 70 No. 9, pp. 588-610.

Liesaputra, V., Witten, I.H. and Bainbridge, D. (2009), "Searching in a Book", in *Research and Advanced Technology for Digital Libraries*, Springer, pp. 442-46.

Lin, J. (2009), "Is searching full text more effective than searching abstracts?", *Bmc Bioinformatics,* Vol. 10.

Lu, W., Liu, D., Fang, F., Long, Q., Yuan, Z. and Zhang, M. (2008), "WHU-XML: An XML based digital library system", in *IT in Medicine and Education, 2008. ITME 2008. IEEE International Symposium on*, IEEE, pp. 380-84.

Luk, R.W., Leong, H.V., Dillon, T.S., Chan, A.T., Croft, W.B. and Allan, J. (2002), "A survey in indexing and searching XML documents", *Journal of the American Society for Information Science and Technology,* Vol. 53 No. 6, pp. 415-37.

Marchionini, G. and Komlodi, A. (1998), "Design of interfaces for information seeking", *Annual Review of Information Science and Technology,* Vol. 33, pp. 89-130.

Moghrabi, C., Le, T.H., Roy, J. and Hachey, J. (2004), "Digital library resources description", in *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, IEEE, pp. 632-37.

Moskovitch, R., Martins, S.B., Behiri, E., Weiss, A. and Shahar, Y. (2007), "A comparative evaluation of full-text, concept-based, and context-sensitive search", *Journal of the American Medical Informatics Association,* Vol. 14 No. 2, pp. 164-74.

Nelson, M.R. (2008), "E-books in higher education: nearing the end of the era of hype?", *Educause Review,* Vol. 43 No. 2, pp. 40.

Paley, W. B. (2002), "TextArc: An alternate way to view a text", available at: http://www.textarc.org/(accessed 21 August 2002).

Patel, A., Bakhtiyari, K. and Taghavi, M. (2011), "Evaluation of cheating detection methods in academic writings", *Library Hi Tech,* Vol. 29 No. 4, pp. 623-40.

Patrick van, A., Pim van der, E., Evert, H. and David, K. (2000), "An interchange format for cross-media personalized publishing", *Computer Networks,* Vol. 33 No. 1–6, pp. 179 - 95.

Rahman, F. and Alam, H. (2003), "Conversion of PDF documents into HTML: A case study of document image analysis", in *Signals, Systems and Computers, 2003. Conference Record of the Thirty-Seventh Asilomar Conference on*, IEEE, pp. 87-91.

Seadle, M. (2009), "COLUMN: ARCHIVING IN THE NETWORKED WORLD PDF in 2109?", *Library Hi Tech,* Vol. 27 No. 4, pp. 639-44.

Seng, J.L. and Lai, J.T. (2010), "An Intelligent information segmentation approach to extract financial data for business valuation", *Expert Systems with Applications,* Vol. 37 No. 9, pp. 6515-30.

Shaofeng, W. (2004), "A method of Java-based electronic document publishing system", *Electronic Library, The,* Vol. 22 No. 4, pp. 351-56.

Sun, Y., Harper, D. J., Watt, S. N. K. (2008), "iSee: Using the Organizational and Narrative Threads Structures in an e-Book to Support Comprehension", available at: http://www.comp.rgu.ac.uk/staff/sy/msPhD.htm(accessed 31 March 2008).

Thomas, S.W., Alexander, K. and Guthrie, K. (1999), "Technology choices for the JSTOR online archive", *Computer,* Vol. 32 No. 2, pp. 60-65.

Uneson, M. (2005), "Tomorrow is File Endings: On Archiving Principles and Archiving Formats", *ScieCom Info,* Vol. 2 No. 2.

Vasileiou, M., Hartley, R. and Rowley, J. (2009), "An overview of the e-book marketplace", *Online Information Review,* Vol. 33 No. 1, pp. 173-92.

W3C. (2012), "e-Extensible Markup Language", available at: http://www.w3.org/XML/(accessed 24 January 2012).

Watt, R.J.C. (2012), "Concordance", available at: http://www.concordancesoftware.co.uk/(accessed 30 April 2013).

Zhang, B., Geng, Z. and Zhou, A. (2008), "SIMP: efficient XML structural index for multiple query processing", in *Web-Age Information Management, 2008. WAIM'08. The Ninth International Conference on*, IEEE, pp. 113-18.

Zhang, W. (2008), "Converting PDF files to XML files", *Electronic Library,* Vol. 26 No. 1, pp. 68-74.

Zhao, F. (2011), "On choosing the digital document's file format for long-term preservation", in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*, IEEE, pp. 370-72.