Integration of Publishing and Document Management into Windows

Technical Report #CS04-21-00
University of Cape Town
Computer Science Department

Lefa Ramike
lramike@cs.uct.ac.za

Kumoyo Mukunyandela
kumoyo@cs.uct.ac.za

Dr Hussein Suleman
hussein@cs.uct.ac.za

ABSTRACT

Document Management is the initial stage of most electronic publishing endeavours. Individuals need to organise documents in such a way that they can be retrieved and accessed on demand, shared selectively, integrated with the work of others and preserved forever.

Publishing of documents is another very important aspect of document management. However, tools that exist for publishing documents are limited in that they only allow a user to send documents to local devices such as disk drives. No tools exist in the user workspace to publish documents in a wide area network.

The report outlines the development of a document management tool that is aimed at addressing the issues of integrating document management and the publishing of documents in the user workspace. The components of the system are the metadata extractor, the publishing tool, the user interface and the email gateway.

The system was tested and evaluated, yielding the following results that show that metadata can be extracted from research documents by defining simple heuristics that define properties of the document and using a statistical learning model. Publishing of documents to a repository in the wide area network and version control of documents was also integrated into the user workspace.

Keywords
Document Management, document version control, version document dissemination.

1. INTRODUCTION

Document Management is the initial stage of most electronic publishing endeavours. Individuals need to organise documents in such a way that they can be retrieved and accessed on demand, shared selectively, integrated with the work of others and preserved forever. Tools in existing operating systems (e.g., Explorer in the Microsoft Windows OS) as well as standalone software tools for document management can be used to achieve this. However, in most cases these tools have to be used collectively to achieve a desired goal, such as listing several versions of a document in a single view. In this case, a document is first accessed using the existing tools in the operating system and then an external document management tool is used to sort documents according to a particular attribute, such as the last date of modification. This presents a change in the workspace which is undesirable and inefficient.

Publishing of documents is another very important aspect of document management and some desktop operating systems have tools to facilitate publishing (e.g., Microsoft Windows has a ‘send to’ function). However, these tools are limited in that they only allow a user to send documents to local devices such as disk drives. To publish a document to a remote location or repository, one has to use application software tools such as FTP or WINSCP, which again presents a change in the

workspace and is also not sufficiently automated.

The first aim of the project is to integrate document management functionality into the existing operating system in a seamless way that is transparent to the user while providing all the document management functions as well as maintaining different versions of documents. This integration should be achieved in both the operating system and the email system, a common mechanism for exchanging documents.

The second aim of the project is to seamlessly integrate the functionality that enables dissemination of documents into the operating system so that management and dissemination can be done in a single user workspace. Web standards such as SOAP will be used where available and the system will conform to the OAIS model for digital preservation.

2. BACKGROUND RESEARCH

Related work is divided into three major sections - document management, dissemination of documents and metadata extraction from documents.

2.1 Document Management

Many document management software tools exist. These provide functionality to organise documents in such a way that they can be easily accessed and retrieved when required, shared selectively and be preserved forever. One such example is Kepler [2, 3] which provides a framework for self-archiving. It is an archivelet publishing tool that creates an OAI-PMH-compliant data provider for individual users as publishers.

Another example of a document management software tool is peerDoc [1]. This tool is mainly designed for institutional use and enables researchers to organise documents in repositories in a local area network.

A functionality that isn't incorporated into most document management packages is that of version control. CVS [4] is a system that implements this by storing copies of the differences between the latest version and other versions. It uses an efficient algorithm to determine how to construct other versions of a given document (i.e., the last version).

The attribute browser is a document manager that provides improved visualisation and usability of the file system to help users search for files easily [6]. It uses the existing file system as well as database queries to access a database so as to link hierarchically stored files by common attributes such as the date of their creation. This work can also be extended to solve the problem of version control.

2.2 Dissemination of documents

Mechanisms for dissemination of documents have been implemented in peer-to-peer as well as other networks.

In a peer-to-peer network, a group of computers can communicate directly with one another rather than through a central server. An example of a document manager that uses a peer to peer network is peerDoc [1], which provides tools for users to search and download one another's documents in a Local Area Network. Dissemination of documents in a network is a limitation as researchers wish to self-archive and distribute their work internationally.

In wide area networks, Web standards such as SOAP can be used to form the basis for an overlying generic dissemination format. SOAP defines a protocol for sending messages with file attachments using an underlying application transport protocol such as HTTP.

2.3 Metadata extraction

Metadata generation is fundamental in many document management software tools. Metadata is used in browsing, searching and managing documents. In many of these tools metadata has to be 'plugged in' manually by the user, which is undesirable. An example of this is peerDoc [1], discussed earlier. Other tools, like Greenstone [8], provide ways of automatically extracting metadata from different document types (e.g., portable document format (pdf), hypertext mark-up language (html), word and rich text format (rtf) documents).

Several methods have been used for automatic metadata extraction. These include regular expressions, rule based parsers and machine learning. Another approach that is used is to treat metadata extraction as a classification problem and use support vector machines. [7]

Machine learning methods are preferred as they are robust and adaptable and in theory they can be applied to any document set. Machine learning techniques for information

extraction include symbolic learning, grammar induction, support vector machines, hidden Markov models and statistical models.

3. APPROACH

This section gives an outline of the general architecture of the system as well as a breakdown of all the components that make up the overall system.

The diagram for the system architecture is shown below in Figure 1.

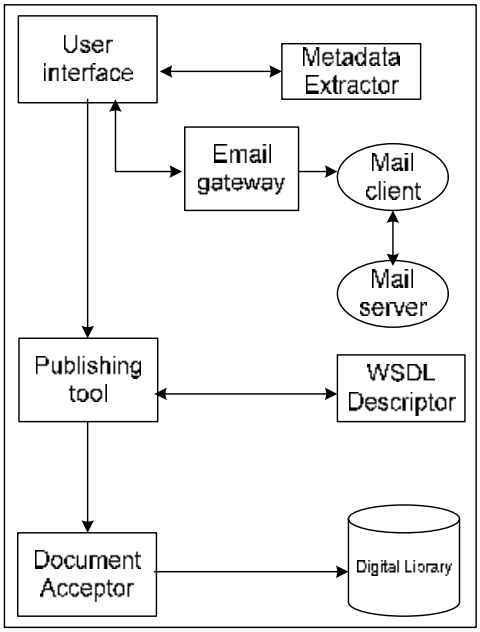

Figure 1: Diagram showing the architecture of the system

The following subsections discuss the design and the implementations of each of the components of the system.

3.1 User Interface

This component is seamlessly integrated with the operating system to allow users to customise their workspace for enhanced document management. It also incorporates asynchronous version control of documents.

This User Interface encapsulates the implementations of the other units and presents a single workspace to the user. It uses the Metadata Extractor to generate tags to describe a particular document. It is linked to the email system via an email protocol gateway that automatically extracts email attachments from the mail client. The system performs automatic organisation of documents into system folders named after the email source or author of the document.

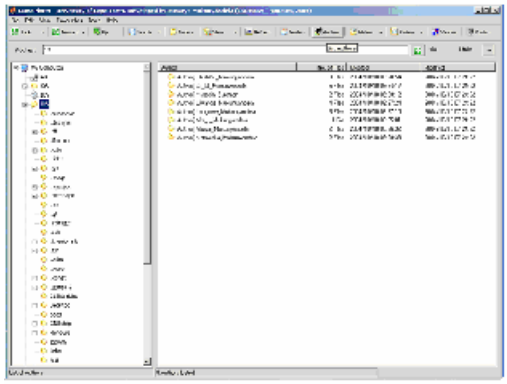

Figure 2: User Interface presents all authors on the system as folders that a user can access

These folders only contain shortcuts to aid the mapping of a selected document to its actual location in the file system. The system maintains transparency as the user is unaware of this particular implementation and the interface closely resembles that of Explorer.

A user can select to view versions of a particular document and will be presented with all versions of that document in a single view. The system only keeps the full original version while subsequent versions are stored as encoded differential files. Upon a user selecting a particular version, the system uses decoding and reconstruction algorithms to integrate all the differences between that particular version and the original version.

3.2 Metadata Extractor

This module is used to generate metadata from an incoming document. It is used by both the User Interface and the Publishing Tool module. Its primary function is given a document it should generate metadata for the purpose of version control and/or the dissemination of the document. Figure 3 below shows the system diagram for this module.

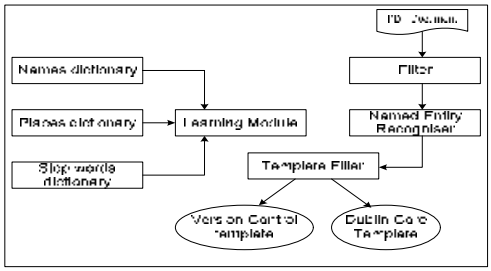

Figure 3: The metadata extractor module

The document is first converted from PDF to ASCII format and then the first page is then retrieved by the Filter module. It is then passed to the Named Entity Recogniser. This module uses simple heuristics that define the properties of the elements extracted in the document. An example of a rule used is:

```
Title Name Address Abstract
```

The learning module can also be used by the Named Entity Recogniser for the extraction of title and the author elements. This module uses a statistical machine learning method to analyse properties of a documents from a training set of metadata records.

Templates of metadata are then created which contain the elements for the asynchronous version control and the publishing tool.

3.4 The Publishing Tool

This module allows a user to automatically post documents to a repository. Figure 4 below shows the system diagram for this module.

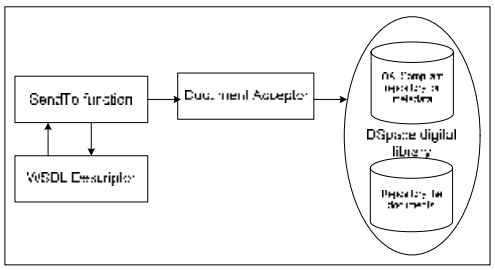

Figure 2: The publishing tool module

The following steps are taken by the publishing tool to publishing a document to the digital library:

Firstly, a SOAP request is made to the WSDL Web Service defined by the metadata format descriptor module for a WSDL description of the document acceptor module Web Service. This Web Service defines a service for accepting a document together with its metadata and then importing it into the digital library.

Secondly, the WSDL Web Service returns a reply to the publishing tool with the WSDL description of the document acceptor module Web Service.

Lastly, A SOAP request with a document and its associated metadata is made to the document acceptor Web Service with the document metadata (from the metadata extractor module) and the collection identifier which then imports the document and the metadata into the specified collection in the DSpace digital library.

3.5 The Email Protocol Gateway

The Gateway provides an interface between the email client and the system to allow the system to filter incoming email so that all documents associated with a particular email source or conversation can be presented together.

The system uses a "poll on demand" scheme whereby the gateway checks the mail client once every 24-hour period for new incoming mail with attachments. Thereafter when a user wishes to list the names of individuals who have sent attachments, the gateway polls the mail client to check if any new emails with attachments have been received since the last time polling was done. If there are any emails, these are added to the system after which the system presents an updated list of all available email sources.

4 RESULTS

In this section we discuss the results from the evaluation of the system. We discuss the results in four sections that relate to the system modules earlier discussed.

4.1 User Interface

We developed an interface that closely resembled Windows Explorer and implemented a subset of its functionality.

Our results showed that documents could be organised in terms of attributes such as the author of the document.

In addition, the interface allowed users to search for documents by the name of the author as well as by the email source name. Users could view all versions of a particular document in a single view and the reconstruction algorithm used was found to be efficient. Our system seamlessly integrated this document management functionality into the user workspace.

4.2 Email Attachment Management

The performance of the "poll on demand" scheme discussed previously was evaluated. Our results showed that our system saved a significant number of CPU cycles by taking

this approach and there was no significant delay in presenting information to the user. Overall our results showed that our scheme was both efficient and sufficient for our purposes.

We also showed that different versions of documents were correctly received among several email sources and therefore showed that asynchronous version control could be incorporated into both the operating and email systems.

4.3 Metadata Extractor

Three approaches we investigated in the evaluation of the metadata. The first used only the defined heuristics to extract metadata from document. The second used heuristics and the learning module to learn the patterns for both the title and the author elements of the metadata. The third used the learning module to learn the patterns for the author element only. The results are shown in Table 5 with the accuracy of the element defined as the number of documents from which the element was extracted correctly.

Element name	**Acc1**	**Acc2**	**Acc3**
Author	50%	85%	95%
Title	80%	80%	95%
Place of publication	25%	65%	90%
Date of publication	100%	100%	100%
Description	35%	65%	95%
Email	65%	60%	95%
Phone number	55%	95%	100%

Table 5: Accuracy of elements using heuristics, learning and heuristics with names in the dictionary defined as Accuracy1, Accuracy2 and Accuracy3 respectively.

4.4 Publishing Tool

This is a proof of concept so a test was conducted to verify that the publishing tool works as required.

The experiment was conducted using 20 test case documents. A set of documents was imported into each of the 5 collections in the DSpace digital library. The importation of each document and its associated metadata into the DSpace digital library was then verified using the browsing utility in the DSpace user interface. All documents were imported correctly.

5 CONCLUSIONS

We developed a system that transparently integrated automatic document management into the user workspace. The system automatically organised documents according to author name as well as the name of the email source who sent the document. The system also facilitated improved access to these documents via these attributes. Users could also select to display all versions of a particular document in a single view.

The results show that metadata can be extracted from a document by defining simple heuristics. Results for the extraction can be improved by either inserting the authors listed in the paper in the dictionary of names or using the learning module to classify the author element.

The metadata extractor generator is also applicable to other areas of document management. It can be used to classify documents using the metadata extracted.

With the publishing integrated into the use workspace, research documents can be self-archived to conference repositories.

6 REFERENCES

[1] Mhlongo., S., Tshivengwa.,P. and Mafike.,S. Peer Group Document and Citation Management. Available online at: *http://pubs.cs.uct.ac.za/archive/00000077/01/Paper.pdf*

[2] Zubair., M., Lui., X. and Maly., K. Enhanced Kepler Framework for Self-Archiving. in *IEEE Computer Society, Proceedings of the 2002 International Conference on Parallel Processing Workshops*, 455.

[3] Zubair., M., Lui., X. and Maly., K., Kepler-An OAI Data/Service Provider for the individual. in *Lib Magazine* April 2001. 7,4 .

[4] Cederqvist, P. *The CVS Manual - Version Management with CVS*. Network Theory Limited, 2002.

[5] Nutt, G. *Operating Systems (3rd edition)*. Addison-Wesley, 2003.

[6] Marsden, G. Improving the Usability of the

Hierarchical File System. *South African Computer Journal*. 32, 69-78.

[7] Fox., E.A., Zhang., Z., Han., H., Zha., H., Manavoglu., E. and Giles., C. Automatic Document Metadata Extraction using Support Vector Machines. *ACM/IEEE Joint Conference on Digital Libraries (JCDL), 2003*, 37-48.

[8] Witten., I., Payntor., G., Boddie., S. and Bainbridge,. D. The Greenstone Plugin Architecture. *in Proceedings of the second ACM/IEEE-CS joint conference on Digital libraries*, 2002, 285-286.