DOL: An Interoperable Document Server

Sergey Melnik*, Erhard Rahm, Dieter Sosna

University of Leipzig, Germany
http://dbs.uni-leipzig.de

Abstract

We describe the design and experiences gained with the database- and web-based document server DOL, which we developed at the University of Leipzig (http://dol.uni-leipzig.de). The server provides a central repository for a variety of fulltext documents. In Leipzig, it has been used since 1998 as a university-wide digital library for documents by local authors, in particular Ph.D. theses, master theses, research papers, lecture notes etc., offering a central access point to the university's research results and educational material. Decentralized administration and different workflows are supported to meet organizational and legal requirements of specific document types (e.g., Ph.D. theses). All documents are converted into several formats, and can be downloaded or viewed online in a page-wise fashion. The documents are searchable in a flexible way using fulltext and bibliographic queries. Moreover, a multi-level navigation interface is provided, supporting browsing along several dimensions. DOL is interoperable with global digital libraries such as NCSTRL and can be ported to the needs of different organizations. It is also in use at Stanford University.

1. Introduction

In the last decade, we have observed a proliferation of digital libraries accessible over the web [5]. A large variety of such libraries has emerged, ranging from centralized document collections to truly global libraries that interconnect collections from many sites. Companies, universities, and other institutions offer their own digital libraries containing documents of their employees, authors etc., which are accessible either within an intranet or globally over the Internet. Many collections focus on specific document types, such as research papers or Ph.D. theses. Such a focus makes it easier to interconnect collections from different sites.

NCSTRL (Networked Computer Science Technical Reference Library) is a well-known example of such a global library of research papers. It is based on a distributed architecture where participating institutions can maintain their own collections and provide bibliographic metadata to achieve interoperability and accessibility over a central interface [8]. Many national and international efforts are under way to provide either centralized or distributed collections of Ph.D. theses from different universities, e.g., NDLTD (Networked Digital Library of Theses and Dissertations) or the German DissOnline [9,4].

We describe the design of and experiences with the database- and web-based document server DOL, which we developed at the University of Leipzig (http://dol.uni-leipzig.de). Its development started in 1998 and was motivated by the fact that commercially available solutions and other document servers at the time did not provide sufficient ease of use and flexibility to cope with the diverse requirements of a university environment. In particular, there is a strong need to support different workflows for bringing in different types of documents such as Ph.D. theses, master theses, research papers, educational material etc. These workflows may differ from department to department, e.g., in the case of master theses, and contribute to the need of a decentralized administration where subcollections are controlled by different persons.

Another distinctive feature of DOL is support for a high degree of interoperability by providing open interfaces for search engines, taking part in the NCSTRL network, and exporting metadata in Dublin Core, a widely accepted standard for bibliographic metadata [2]. Furthermore, DOL offers multi-lingual web interfaces and is controlled by an XML configuration file permitting easy adaptation to different organizational structures, document types, etc. At the university of Leipzig, DOL is in operation as a university-wide digital library since July 1998. It is part of the NCSTRL network since May 1999. At Stanford university, a DOL installation has been used since March 2000 for managing the publications of the database research group (http://dbpubs.stanford.edu).

In the following, we briefly outline the major requirements that influenced the design of the DOL document server (Section 2) and highlight the capabili-

* Current address: Stanford University, Stanford CA 94305 USA

ties of DOL with respect to its end-users (information users, authors, and administrators). In Section 4 we describe the architecture and implementation of DOL. Section 5 discusses the architectural choices we made and the lessons learned so far.

2. Requirements and objectives

The design of DOL was influenced by general requirements for digital libraries as well as specific considerations from a university environment. Of course, the system had to be internet-based to permit access to all documents in the collection at any time from any machine in the world running a web browser, independently of the hardware or operating system in use, and without having to install special software. All documents should be accessible from a central interface despite the fact that they are from different departments and covering different subjects.

To easily determine the documents relevant to a particular user, flexible navigation and search facilities are to be provided including bibliographic queries and fulltext retrieval with good precision and recall. To attract local and international users, multi-lingual user interfaces are needed (in our case, at least German and English). Also important is the provision of widely accepted document formats permitting a high quality for online viewing and printing.

The system should make it easy for authors to submit their documents and to make them accessible worldwide. Authors should provide the bibliographic data including optional keywords themselves to avoid the expensive and time-consuming involvement of librarians to classify and index documents. The document server has to guarantee the persistence and stability of a document and its citation data, so that it can be referenced in other publications and can be electronically accessed at a stable Web URL.

Administration of the document collection is necessary in order to provide a high quality service. In particular, it must be ensured that only authorized users can store documents in the system and that the authors agree with the online publication of their documents. Furthermore, copyright violations should be prevented. The system should also support specific workflows for documents such as master or Ph.D. theses to permit their electronic dissemination while guaranteeing conformance to local regulations. Finally, the system should be easily adaptable to changes in the organizational structures, supported document types, etc. Moreover, it should be interoperable with other digital libraries.

3. Overview of Capabilities of DOL

In this section we briefly summarize the functionality and the capabilities of DOL from the perspective of its end-users. The exposition shows how we addressed the requirements discussed above and motivates the implementation details presented in the next section. The end-users of the server can be divided into the following four groups: authors, readers, collection administrators, and system administrators.

Readers, or information consumers, are users who access the document collection to satisfy their information needs. DOL offers them multi-lingual navigational browsing, bibliographic field search and fulltext search. Browsing can be performed along five dimensions in arbitrary order: organizational unit, subject area, publication language, document type and publication year (see Section 4.2). Each document can be downloaded and viewed online in one of several formats, currently PDF, Postscript, or ASCII. Additionally, documents can be viewed page-wise either in ASCII/HTML or in a "graphical mode". In the latter case a GIF image per document page is displayed guaranteeing a high presentation quality for images, formulas, etc. A page-wise preview allows to check the contents of a document before downloading which may be especially valuable for users with low-bandwidth connections.

DOL provides information about the total number of documents in the system and its sub-categories, about the most recently added documents and the most frequently accessed documents ("top hits"). Moreover, DOL supports a query interface that can be used in other web pages to dynamically determine commonly needed collections, e.g., all Ph.D. theses of a specific department or all documents of a given author[*]. Individual documents have a stable URL so that information users can use them in their reference lists, etc.

Authors, or information providers, want to make their (typically scientific) publications available on the Web with little effort. They want to reach broad audience and have their work preserved for a long period of time. DOL offers a simple form-based procedure for providing complete bibliographic information and for uploading the publications by the authors to the server either in PDF or Postscript. DOL copies the input file and automatically generates and stores the additional formats mentioned above to offer for all documents the same presentation and download formats. Moreover the document is indexed to support fulltext retrieval, and is assigned a stable and simple URL containing the publi-

[*] For instance, the query / URL "`http://dol.uni-leipzig.de/pub/search?AUTHOR=Rahm&lang=en`" returns an English list of all DOL documents at the Leipzig server authored by Rahm.

cation year and a relative number per year (e.g., http://dol.uni-leipzig.de/pub/1998-43).

There are different workflows for bringing in a new document depending on the document type and organizational unit, e.g. department. A workflow involves the author, a collection administrator and possibly other persons. For instance, in the case of a Ph.D. thesis it must be checked that the author has already passed certain steps in her procedure, that the electronic version is identical to the final printed version, etc. To help ensure the quality of the documents, collection administrators verify the content, metadata, and the authorship of documents. Authors have to provide a signed *permission and copyright statement* to ensure that they agree that their document is (non-exclusively) kept in the document server and to confirm that there are no conflicting copyrights, e.g. by publishers. DOL offers a standard copyright notice to be displayed with a document which may be changed by the author when specifying the bibliographic metadata. DOL automatically generates e-mail notifications to inform authors and administrators about the current document state, or to request further action.

To make new documents known worldwide, DOL (a) provides a specialized interface that allows the crawlers like Google, AltaVista etc. to index of the collection; (b) supports a Dienst interface that allows the documents to be found in the NCSTRL system, and (c) exports document metadata in DublinCore. DOL facilitates backup of the entire collection to ensure the persistence of the documents.

Collection administrators are responsible for maintaining certain kinds of publications, for example, those of a Civil Engineering Department. Often, the collection administrators are committed to document maintenance by organizational structures and guidelines. For instance, the Libraries of Leipzig University are committed to managing all Ph.D. and professorial (habilitation) theses available both electronically and on paper. As another example, the rules of procedure of the Math & CS Dept. in Leipzig require all Master's theses to be received and archived in a digital format before the M.S. degree is awarded.

The server supports the needs of collection administrators by offering a capability of decentralized online administration that supports automatic format conversion, basic workflows, keeping the history of document processing etc. Moreover, DOL provides explicit support for gathering permission and copyright forms from the authors, and gives the administrator an instrument for enforcing copyright statements, if required.

The *server administrator* deals with the technical maintenance and physical preservation of the collection. Furthermore, he or she determines the privileges of collection administrators depending on their roles.

4. Architecture and Implementation

This section highlights the implementation details of the DOL document server. Its architecture is based on the layered model for cellular document repositories proposed by Crespo et al [1]. The key idea of this model is to organize the components of a document repository into layers that offer well-defined interfaces to the layers above. The implementation of each layer can be exchanged without affecting the other layers.

As shown in Figure 1, a Document Server site comprises several loosely coupled components and systems. The central component of the server site is a Java servlet that implements most management and presentation functions. The servlet is executed by a Web server, e.g. Apache. The documents are stored in a Data Store in several presentation and download formats. Along with the fulltext, the bibliographic and administrative metadata of the documents is stored in the Data Store. The metadata is indexed using an SQL database to support field search, workflow, and access authorization. Fulltext indexing is done using a stand-alone fulltext-indexing engine, currently htDig[1]. The arrows in Figure 1 indicate major data flows.

Document fulltext and metadata are stored persistently using a so-called *Data Store* component. The Data Store provides basic methods for managing opaque data objects associated with a document identifier. For instance, the Data Store is capable of opening input/output streams for a given type of data object like "PDF", or it can return the list of all document identifiers in use. Currently, the Data Store is backed by a file system, but can be easily replaced by an alternative implementation that, for instance, stores BLOBs in a database (this flexibility is the primary benefit of using a layered approach mentioned above). Currently supported fulltext formats are uncompressed and compressed PostScript, PDF, and ASCII, and GIF images supporting page-wise preview of documents. There are two metadata files per document for bibliographic and administrative information, respectively (see below).

The *Document* component is used to access the document metadata. For minimizing Data Store access, which typically results in read/write operations on the file system, it caches document metadata in memory using the LRU (Least Recently Used) caching policy. For efficient querying of document metadata, the *Metadata Indexing* component extracts metadata relevant for queries, and replicates it in a SQL database.

The *Workflow Component* captures the logic of document submission and admininistration. It uses the Metadata Indexing component for answering queries

[1] htDig is a moderate-scale search engine, which is available without charge from www.htdig.org.

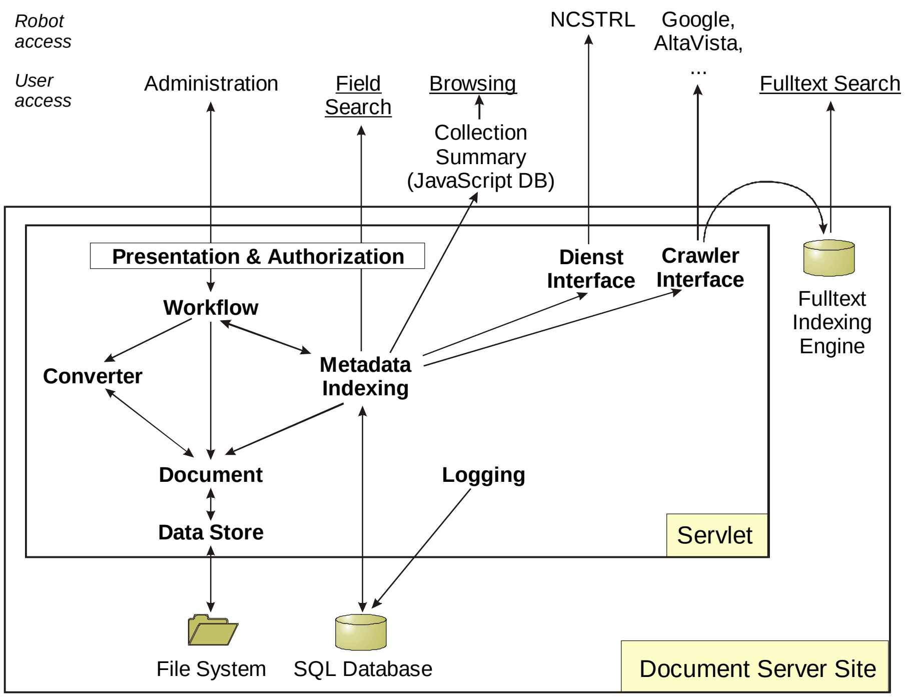


Figure 1: Functional diagram and components of Document Server

that involve more than one document like listing of all documents to be processed by a given collection administrator. The workflow component relies on *Converter* for automatic conversion between document formats.

The Document Server provides several interfaces that are tailored for readers, administrators, search engines and services such as NCSTRL. These interfaces are backed by the aforementioned servlet components. The *Presentation* component takes care of generating HTML pages shown to readers and administrators. The crawler interface is used by the on-site search engine for building a fulltext index of the collection.

The following sections describe the various components of the server introduced above in more detail.

4.1 XML-based Configuration

To facilitate the configuration and maintenance of the Document Server, most of configuration options of the server have been made explicit and are kept in an XML file. This file contains the following information:

- Definitions of bibliographic fields
- Multilingual textual elements
- Authorization information (Section 4.4)
- Workflow and conversion parameters (Section 4.5)
- Miscellaneous system parameters such as database configuration, maximal number of search results shown, NCSTRL parameters, locations of log files etc.

The example below shows a portion of the definition of the bibliographic field “ORGANIZATION” used in the Leipzig server:

```
<field bib=”ORGANIZATION”
          type=”enumeration”
          presence=”required”
          card=”multiple”>
  <name xml:lang=”de”>Organisation</name>
  <name xml:lang=”en”>Organization</name>
  <values>
    <value id=”LEI.12”>
      <name xml:lang=”de”>Fakultät für
       Physik und Geowissenschaften</name>
      <name xml:lang=”en”>Faculty of
            Physics and Geosciences</name>
      <desc href=”http://www.uni-
leipzig.de/physik/”/>
     </value>
     ...
   </values>
</field>
```

In the above example, the bibliographic field "ORGANIZATION" is defined as a required field with an enumerated range of values that may occur multiple times. The identifier "LEI.12" denotes one of the valid values for the field that corresponds to the Faculty of Physics and Geosciences. Controlled vocabularies or classification schemes used for enumerated bibliographic fields are specified in a similar fashion. For instance, the server in Leipzig uses the subject list of the German National Library (that includes entries like "Fine Arts", "Pollution control" etc.), whereas the server in Stanford uses the subject list defined by the members of the Stanford Database Group ("Databases and the Web", "Data Integration and Mediation" etc.). Furthermore, instead of categorizing the publications by organization, the server in Stanford classifies the documents according to projects like TSIMMIS, Lore etc. The definitions of bibliographic fields may include JavaScript code necessary to validate the syntactic constraints for the fields that are checked during document submission.

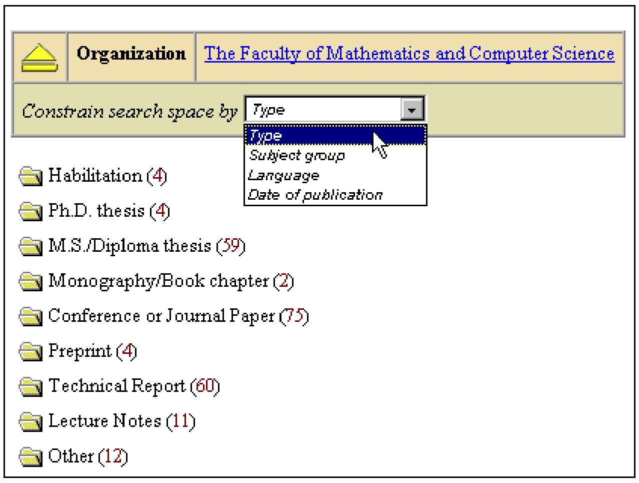
The names of all textual elements that appear on the HTML pages generated by the server are contained in the configuration file. This approach allows porting the server for any international installation easily. Currently supported are interfaces for English and German.

4.2 Browsing and Navigation

Browsing is aimed at giving the users a quick overview of the content and extent of the document collection. The browsing in the Document Server is realized as a client-based JavaScript application. The JavaScript application supports drilling-down of the collection according to the dimensions defined by the enumerated fields. As an additional dimension, the server uses the date of publication. An example of the browsing user interface is shown in Figure 2. In the example, the first dimension that the user selected was organization. At the second navigation level, which is shown in the figure, the user is presented with further options of restricting the scope of browsing along the remaining dimensions like document type, subject group etc.

Notice that along any navigation path the user is presented with the number of the documents in the corresponding document subspace. As the figure suggests, there are 75 conference or journal papers that belong to the Math & CS Dept. A click on the corresponding folder symbol lists these 75 documents.

The statistical information about the collection is gathered by the server periodically, and is used for generating a compact JavaScript database that is shipped to the client when browsing is invoked. This small database contains exactly one record for any permutation of attributes at a given navigation depth. For instance, the number of documents (like 75 in above example) is recorded only once for the given document type and organization. If the user selected "Conference of Journal Paper" at the first level of navigation, she would be presented with 75 hits for Math & CS Dept that originate from the same single row in the JavaScript database.



Organization | The Faculty of Mathematics and Computer Science

Constrain search space by: Type

Type
Subject group
Language
Date of publication

Habilitation (4)
Ph.D. thesis (4)
M.S./Diploma thesis (59)
Monography/Book chapter (2)
Conference or Journal Paper (75)
Preprint (4)
Technical Report (60)
Lecture Notes (11)
Other (12)

Figure 2: Client-based navigation gives an overview of the document collection

The size of the JavaScript database grows combinatorially with the navigation depth and the number of possible values for a given field. For example, for maximal navigation depth 2, the size of the JavaScript database generated by the server in Leipzig is 12KB. For depth 3, the database size amounts to 35KB.

4.3 Document Metadata Files and Metadata Indexing

Each document stored on the Document Server, comprises several fulltext versions and two metadata files. One of the metadata files is a so-called BIB file [6] that contains the bibliographic metadata of the document. The other metadata file contains the administrative and workflow metadata stored in XML.

As an example, consider a publication with the title "S-layer reconstitution at phospholipid monolayers" stored on the Document Server in Leipzig with identifier 1998-43. Compressed PostScript and PDF are saved in the DataStore as files 43.ps.gz and 43.pdf.gz, respectively. The bibliographic data is contained in the file 43.bib, which has the following content:

```
BIB-VERSION:: CS-TR-v2.1
ID:: 1998-43
AUTHOR:: Wetzer, Barbara;
   Pfadler, Alexander; Györvary, Erika;
   Pum, Dietmar;  Lösche, Mathias;
   Sleytr, Uwe B.
```

```
TITLE:: S-layer reconstitution at phos-
pholipid monolayers
CITATION:: Langmuir, 14 (1998),
        6899-6906
PAGES: 21
LANGUAGE:: en
ORGANIZATION:: LEI.12
TYPE:: LEI-Preprint
DISCIPLINE:: DNB-5-29;DNB-5-30
...
```

The organization, type and discipline codes shown above are defined in the server configuration file as described above. A portion of the workflow data of the document stored as an XML file is shown below:

```
<state version="1.0"
     contact-email="loesche@physik.uni-
leipzig.de"
     submission-date="1998-10-23">
    <converted PDF="yes" PS="original"
             ASCII="yes" graphics="yes"
             reverse="no" color="yes"/>
    <workflow state="legitimate"/>
    <copyright
         submission-date="1998-11-09"/>
    <history>
       ...
</state>
```

The above metadata indicates that the document was submitted on October 23, 1998 in PostScript format and is in a "legitimate" state, i.e. the copyright statement for the document has been received by the administrator. The PostScript file has been converted to PDF, ASCII and graphics, as stated by the XML tag `<converted>`. The Document Server records the complete document processing history, which includes conversion information, correspondence with the author(s) and descriptions of other management operations performed on documents. An example of the administration interface that shows an excerpt of the document history is displayed in Figure **4**. Decentralized administration is addressed below in Section 4.4.

The metadata-indexing component extracts a portion of the bibliographic and administrative metadata of each document and stores it in a relational database. The amount of replicated metadata determines the query capabilities of the server. The indexer runs as a separate thread, which periodically calls the DataStore to determine which documents have been changed. Besides the "pull" mode, the indexer supports "push", i.e. can be invoked explicitly. The changes of the document metadata are typically due to document submissions and administrator activities. From time to time, the indexer rescans the whole collection to make sure that the metadata replicated in the SQL database is consistent with that in the DataStore.

The administrative metadata collected by the indexer is primarily used for deciding which documents are public, i.e. can be found during search and browsing. For efficiency purposes, the metadata-indexing component is heavily used by all components that access metadata of more than one document at a time. Such components include user search, Dienst and crawler interface etc. (compare Figure 1). The indexer communicates with the SQL database using JDBC.

4.4 Administration and Authorization

Decentralized administration is one of the key features of DOL. The tasks of an administrator include verification of the authorship of publications, checking success of the format conversion, and copyright management. The assignment of an administrator to a document is based on the metadata of the document. The document collection can be viewed as a multidimensional space, whose individual dimensions are associated with the enumerated metadata fields like document type, organization, or subject. For example, all documents with the ORGANIZATION field carrying the value "LEI.10" form a document subspace that corresponds to the publications of the Math & CS Department of Leipzig University.

Using a combination of field values across different dimensions allows dividing the collection in a granular fashion and tailoring it to the administrative structures of the university. For example, the Libraries of Leipzig University exclusively maintain and preserve all Ph.D. and professorial (habilitation) theses available both on paper and electronically. Therefore, the privileges of the administrator of say the Math & CS Dept. are defined in the way that allows her to manage all documents published by the department except those that have a value "LEI-Diss" or "LEI-Habil" in the TYPE field (see Figure 3).

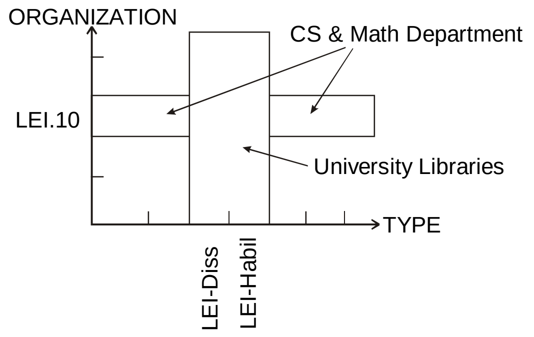
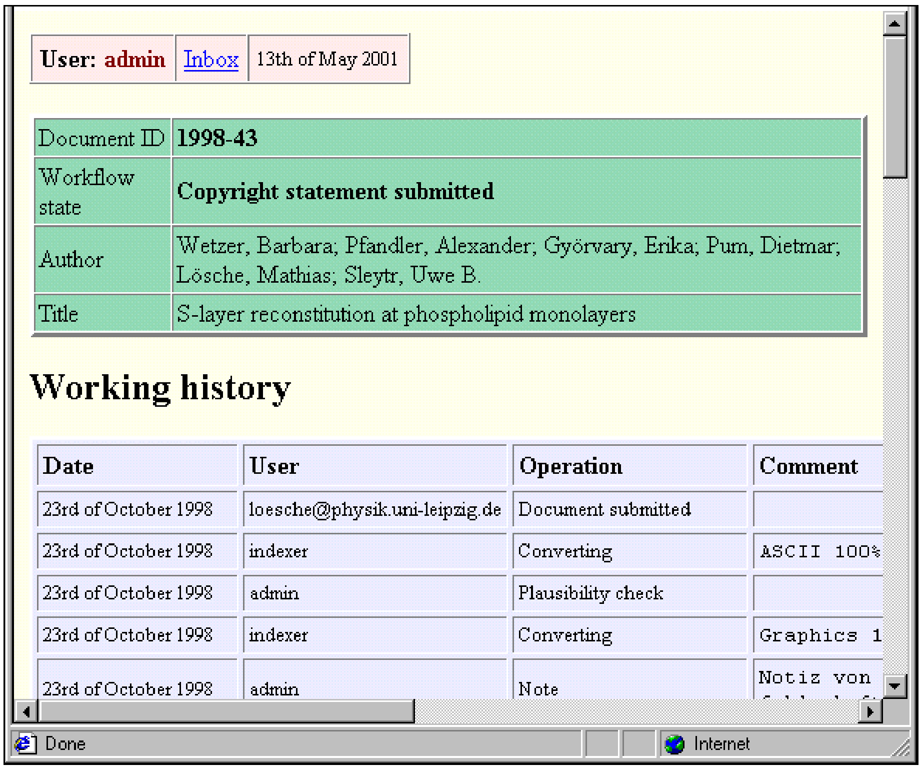


Figure 3: Example of division of administrative privileges



User: admin	Inbox	13th of May 2001

Document ID	**1998-43**
Workflow state	**Copyright statement submitted**
Author	Wetzer, Barbara; Pfandler, Alexander; Györvary, Erika; Pum, Dietmar; Lösche, Mathias; Sleytr, Uwe B.
Title	S-layer reconstitution at phospholipid monolayers

Working history

Date	User	Operation	Comment
23rd of October 1998	loesche@physik.uni-leipzig.de	Document submitted	
23rd of October 1998	indexer	Converting	ASCII 100%
23rd of October 1998	admin	Plausibility check	
23rd of October 1998	indexer	Converting	Graphics 1
23rd of October 1998	admin	Note	Notiz von

Figure 4: Administrator has access to the history recorded during the lifetime of a document

Using the XML configuration described above, the administrative privileges of the Math & CS Dept. administrator are encoded in the following way:

```
   <user id="ifi" role="legitimator"
email="doladmin@informatik.uni-
leipzig.de">
      <subspace type="include">
        <dim field="ORGANIZATION"
             value="LEI.10"/>
      </subspace>
      <subspace type="exclude">
         <dim field="TYPE"
              value="LEI-Diss,LEI-Habil"/>
      </subspace>
   </user>
```

Upon submission, every document whose ORGANIZATION field contains "LEI.10" and whose field TYPE does not contain any of "LEI-Diss" or "LEI-Habil" appears in the Inbox of the Math & CS Dept. administrator. The author and the administrator are notified by email about the successful submission.

4.5 Workflow and Format Conversion

During its lifetime, a document changes several workflow states. The main workflow states of a document are summarized in Figure 5. The workflow information is kept in the XML metadata file. A newly submitted document is marked as "new". Immediately, an automatic conversion process is initiated in the background. The converter attempts to generate the missing fulltext formats of the document. For example, if the document arrives in PostScript format, the converter tries to create a PDF version, and the other way around. It is noteworthy that since PostScript is a programming language, the conversion tools that we are using sometimes run into endless loops. In such cases, the conversion processes are forcefully terminated after a timeout interval.

Once the authorship of the document and the conversion status have been verified by the administrator, the document changes its workflow state to "plausible". After the permission and copyright statement has been received from the author by regular mail, the document becomes "legitimate". In any state except "legitimate" the document may expire either due to negligence of the administrator or a missing copyright statement from the author. Expired documents are deleted, but are not physically erased for a long period of time to enable a possible recovery of the document.

Further workflow operations used in the server include disabling/enabling individual documents and activating modification privileges for the authors when, for example, the fulltext version of the document turns out to be corrupt and needs to be resubmitted (see states marked as "resubm." in Figure 5). The ability to resubmit just the fulltext saves the authors from tedious reinput of the bibliographic metadata.

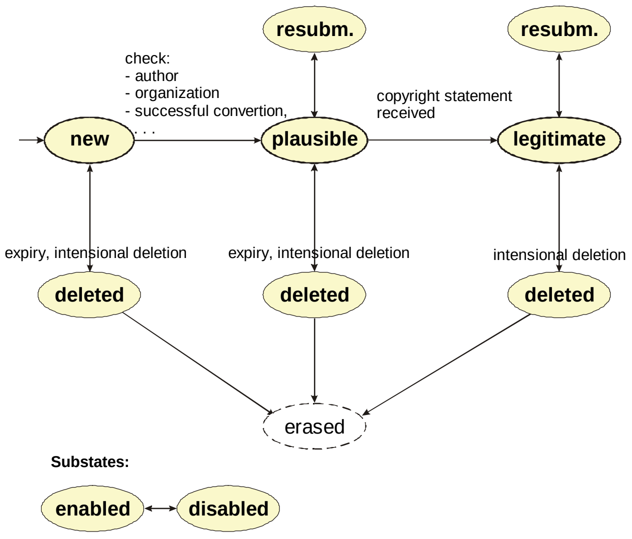


Figure 5: Summary of workflow states of a document (conversion omitted for brevity)

The sketched document workflow is part of larger workflows that can be defined for specific document types and organizational units. These workflows are defined and enforced outside the document server involving the author and collection administrator. For instance, the collection administrator can decide when a new document is made public, e.g. in the document state "plausible", in state "legitimate" or when additional real-world constraints are satisfied, e.g. a passed examination needed for a master's degree.

4.6 Fulltext Indexing and Export Interfaces

The fulltext indexing of the document collection is done using a loosely coupled external fulltext search engine. In both Leipzig and Stanford sites, we use htDig. The crawler interface of the server (see Figure 1) is designed to export the bibliographic metadata and the fulltext of each document in ASCII format to the crawler engines. Whenever a document is accessed, the "Agent" field of the HTTP request is evaluated to determine whether the request is coming from a browser of a search robot. If a search robot like htDig, Googlebot (Google), or Scooter (AltaVista) is detected, the crawler is presented the combined metadata and fulltext content instead of the Web page shown to human users.

The same approach is used both for driving the on-site search engine as well as the external ones. The on-site engine is invoked periodically, typically several times a day, as a separate process that reindexes the whole collection. That is, a newly submitted document usually becomes available for fulltext search after a delay of several hours.

Besides the crawling interface, the Document Server offers two other ways of accessing the document metadata. First, the HTML pages presenting the document descriptions as human-readable HTML pages contain embedded DublinCore metadata that is encoded in RDF/XML. This feature is provided to anticipate metadata-aware crawlers.

A more interactive interface for accessing document metadata is offered via the Dienst protocol, which provides metadata-querying capabilities [3]. The Dienst protocol is used in the NCSTRL system. Dienst supports incremental indexing of documents residing at multiple participating sites. The Dienst component used in the server deploys the metadata-indexing component for reporting the incremental changes in the collection.

5. Discussion and Lessons Learned

In this section we summarize the technical decisions made in the Document Server and the lessons that we learned by deploying two sites in Leipzig and in Stanford.

One of the non-traditional decisions that we made was to store all document metadata in the file system (strictly speaking, in a Data Store with voluntarily limited access capabilities), and use the SQL database only for indexing purposes. That is, the file system is the primary source of document metadata, whereas the SQL database holds a partial replica. The necessity of this architecture was partly caused by the desire to use format conversion and presentation tools developed at Karlsruhe University [10]. These tools rely on the availability of fulltext and BIB metadata in the file system. The desire to use these components also prompted us to keep bibliographic metadata in BIB files instead of encoding everything in XML.

The upside of relying on a SQL database just for indexing is the reduction of database maintenance effort and the dependency on the database as a point of failure. In fact, the server offers limited functionality even if the SQL database is down (fulltext search still works fine, individual documents referenced directly can be displayed). Writing tools for migrating the SQL database to a new format is unnecessary, since its content can be scrapped and restored from the file system. Furthermore, backup of fulltexts and metadata is simplified. The main downside of replicating a portion of the metadata is due to the required synchronization of the database content and the metadata in the file system.

For fulltext indexing, we are using a loosely coupled fulltext search engine. While this approach guarantees maximal vendor independence and flexibility, its major disadvantage is the decreased freshness of the fulltext index. However, due to the incremental nature of our document collections, typically only newly submitted documents are affected by the indexing latency, which is not dramatic.

Another experience that we gained is that using a presentation format like PostScript for ASCII text extraction is a painful and error-prone task. Often, the extraction tools fail due to the fact that some PostScript generating programs produce graphics instead of keeping the text characters. Furthermore, special care was required for designing a robust conversion subsystem that does not get stuck when a PostScript interpreter runs into an endless loop. We anticipate that using comprehensive OCR tools may be a better alternative. OCR tools have another serious advantage in that they facilitate processing of documents for which electronic versions are not available.

Recall that the Document Server supports a page-wise preview feature that allows the users to skim through a document using GIF images generated for each page. This feature is primarily targeted at low-bandwidth users. By analyzing the server logs, we found that despite increased bandwidth available to the users, page-wise preview is still helpful. For example, in Leipzig for each download of a fulltext version, three document pages are viewed as images (averaged across all documents). In Stanford, this figure is substantially lower; only one page is viewed per download.

An apparent reason for this difference is that on average, the documents in Leipzig are five times larger than the ones in Stanford. The Stanford server mostly contains conference publications, which were originally written in LaTeX. In contrast, a significant portion of the collection in Leipzig are theses, lecture notes, and book contributions, many of which were authored using WYSIWYG editors like MS Word. The average sizes of compressed fulltext files in Leipzig are 1 MB for gzipped PostScript, and 700KB for PDF. In Stanford, the respective figures are 150KB and 180KB. These differences in the content of the collections may also partially account for the skewed download figures; in Leipzig, 38% of accesses to document metadata result in download of any of the fulltext versions. In Stanford, 63% of accesses lead to downloads (the above numbers exclude hits made by crawlers).

Despite expectations, client-based browsing proved to be not all too useful. In fact, in a year of deploying the server in Stanford, the browsing feature was used only around 1500 times, i.e. less than 2% of the number of downloads of fulltext versions of documents. Thus, the savings in terms of server load by using client-side processing were minimal. This figure also suggests that browsing in an online document collection seems much less important for the users than, for example, fulltext search.

6. Conclusions

We have presented the design of the document server DOL providing flexible and powerful management for a variety of documents. It supports decentralized administration, different workflows and a high degree of interoperability. Furthermore, DOL offers multi-lingual web interfaces and is controlled by an XML configuration file permitting easy adaptation to different needs. The experiences so far are very positive and show that DOL is a robust service which is heavily used, especially by information users.

We have received numerous requests from people of other institutions who are interested in running a DOL installation and we thus plan to make the implementation publicly available in the near future. We further plan to extend the DOL functionality by supporting multi-part documents and additional interfaces, e.g. to global Ph.D. collections.

Acknowledgements

The development of the document server was supported by the Ministry of Culture and Science of the Free State of Saxony within the HSP3 program. We thank Günther Radestock, Karlsruhe University, for providing us the Pscript tool used for document format conversion.

7. References

[1] A. Crespo and H. Garcia-Molina. *Archival Storage for Digital Libraries.* Third ACM Conf. on Digital Libraries, Pittsburgh, PA, USA, 1998

[2] DublinCore Home Page. http://dublincore.org/ , 2001

[3] Dienst Overview and Introduction. http://www.cs.cornell.edu/cdlrg/dienst/DienstOverview.htm 2001

[4] DissOnline Home Page, http://www.dissonline.de/ , 2001

[5] CACM, Vol. 44(5), Special Issue on Digital Libraries, May 2001

[6] R. Lasher and D. Cohen. *A Format for Bibliographic Records*, RFC 1807, June 1995

[7] I.H. Witten, R. J. McNab, S. J. Boddie and D. Bainbridge. *Greenstone: a Comprehensive Open-Source Digital Library Software System.* Proc. Fifth ACM Conf. on Digital Libraries, San Antonio, Texas, pp 113-121, 2000

[8] NCSTRL Home Page. http://cs-tr.cs.cornell.edu/ , 2001

[9] NDLTD Home Page, http://www.ndltd.org/, 2001

[10] G. Radestock. The Pscript Home Page. http://www.ubka.uni-karlsruhe.de/~guenter/pscript/ , 2001