Network Working Group
Request for Comments: 1691
Category: Informational

W. Turner
LTD

August 1994

The Document Architecture for the Cornell Digital Library

Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Abstract

This memo defines an architecture for the storage and retrieval of the digital representations for books, journals, photographic images, etc., which are collected in a large organized digital library.

Two unique features of this architecture are the ability to generate reference documents and the ability to create multiple views of a document.

Introduction

In 1989, Cornell University and Xerox Corporation, with support from the Commission on Preservation and Access and later Sun Microsystems, embarked on a collaborative project to study and to prototype the application of digital technologies for the preservation of library material. During this project, Xerox developed the College Library Access and Storage System (CLASS), and Cornell developed software to provide network access to the CLASS Digital Library.

Xerox and Cornell University Library staff worked closely together to define requirements for storing both low- and high-resolution versions of images, so that the low-resolution images could be used for browsing over the network and the high-resolution images could be used for printing. In addition, substantial work was done to define documents with internal structures that could be navigated. Xerox developed the software to create and store documents, while Cornell developed complementary software to allow library users to browse the documents and request printed copies over the network.

Cornell has defined a document architecture which builds on the lessons learned in the CLASS project, and is maintaining digital library materials in that form.

Document Architecture Overview

Just as a conventional library contains books rather than pages, so the electronic library must contain documents rather than images. During the scanning process, images are automatically linked into documents by creating document structure files which order the image files in the same way the binding of a book orders the pages. Thus, the digital book as currently configured consists of two parts: a set of individual pages stored as discrete bit map image files, and the document structure files which "bind" the image files into a document. In addition, a database entry is made for each digital document which permits searching by author and title (i.e., bibliographic information). Beyond the order of the pages, the arrangement of a physical book provides information to readers. The title page and publication information come first; the table of contents usually precedes the text; the text is divided into sections or chapters; if there is an index, it follows the text. The reader often refers to these components of a book when browsing the library shelves, in order to determine whether to read the book.

The document structure provides direct access to the components of an electronic document, storing the information that would otherwise be lost when the book is disbound for scanning.

Document Architecture Requirements

Listed below are the requirements that were initially set down for the Cornell Digital Library Architecture.

1. The architecture must be open (i.e., published and freely available).

2. The architecture should be as simple as possible (to facilitate product development).

3. The architecture should assume data storage in UNIX file systems.

4. The architecture should allow for standard data usage, such as via FTP and Gopher servers (i.e., pages of a document must exist in a single directory, and the naming convention used must order them in the standard collating sequence, such as the series "0001.TIF, 0002.TIF,..., 0411.TIF" (NOTE: a series such as "1.TIF, 2.TIF,..., 10.TIF" would be ordered "1.TIF, 10.TIF, 2.TIF, ..." which is not acceptable).

5. The architecture should provide for storing the same information in different formats. For example, when a page of a document is available at several different resolutions.

6. Low-resolution "thumbnail" images of each page must be stored to facilitate browsing and sharing of data.

7. The architecture must support distribution of files so that similar files may be stored together, permitting optimization of storage use and performance.

8. The architecture must support documents that are composed of references to all or part of other documents.

9. The architecture must support document components which are stored on separate servers distributed across the network.

10. The architecture must support not only an hierarchical structure for each document, but the ability to define multiple views of each document.

11. The architecture should accept, rather than dictate, directory structures in which documents will be stored. This will permit documents created in other ways to be added to the Digital Library simply by adding database information rather than by copying or moving files.

Document Architecture Description

A digital library consists of a Digital Library Server, networked storage, and a referencing database. A single digital library will contain one or more collections. Each collection will contain one or more documents.

The referencing database allows searching for documents by author, title, and document ID. In the current implementation, the referencing database is a relational SQL database, and each collection is epresented by a table in the database. It is planned to migrate to Z39.50 database searching as the preferred method, as this protocol has been established as the standard for library applications.

Authorization will be primarily collection-based, although the design will permit authorization checking at any level down to the individual file. Notification would come only when the patron attempted to open the document or access the particular component.

Each document consists of three components: the logical structure; the physical references; and the data files.

The logical structure is a logical description of the document. Conceptually, a document is a tree, with the leaves being the data files (pages). At a minimum, all documents have a logical structure which lists the pages in the document and the order in which they appear. Usually, documents will have a more elaborate structure. The logical structure relates the logical structure of a document to the physical references which make up the document.

These physical references map the lowest levels of the document's logical structure (the leaves of the tree) to the files that contain the data. Where there are multiple representations of a page, such as images at various resolutions, these are linked together in the physical references file.

The data files contain the data making up a document. Any format can be accommodated: image files, ASCII text, PostScript, etc. However, one-to-one correspondence between data files for a given physical reference is assumed. That is, if there are multiple file types for a single page, these files should represent exactly the same information.

Physical References File

The Physical References file is the component of the document which relates logical structures (logical components of documents) to physical files. Document references, by which a document can be composed of all or part of other documents possibly residing on different servers, are handled in the Physical References file.

A document may contain multiple document objects, each of which contains one or more data objects. When a document contains actual physical data (for example, it is created by scanning or importing images), a Master Document Object is created. When a document incorporates components of other documents, a Reference Document Object is created for each of the other documents. The Document Objects are numbered with internal reference numbers, which are included in the corresponding Data Object lines.

Data Object lines include the Document Object number, the file reference number, and the file type. The Document Object number refers to a Document Object line, from which the library name, collection name, and document ID can be retrieved. The tuple

<libraryID>+<collectionID>+<documentID>+<filetype>+<file reference>

is guaranteed to locate a file. Each Data Object line refers to a single file; where multiple file types of a single document page exist, there will be multiple Data Object lines for that page.

In the file, all Document Object lines will preceed all Data Object lines for a given document. Document Object lines may be either grouped together at the beginning of the file, or may immediately preceed the first Data Object line for the Document Object. Document Object lines will appear in order by Document Object number. Data Object lines will appear in order by sequence number, NOT by Document Object number.

The fields in the Physical References file are delimited by vertical bars.

Document Object Lines

Field	Description	Comments
1	Document Object number	0 => Master Document Object 1-9 => Reference Document Object
2	Library name	Server name
3	Collection name	
4	Document ID	8-digit number
5	Author name	
6	Volume	
7	Title	
8	Edition	

Data Object Lines

Field	Description	Comments
1	Document Object number	Corresponds to above
2	Sequence number	
3	File reference	Reference number used to locate file in filing system
4	Physical reference number	Equal to Logical Structure file
5	File type	1 = TIFF 600dpi 2 = TIFF thumbnail 3 = ASCII version of page (i.e., OCR output) 4 = ASCII notes 5 = Other 6 = TIFF 300dpi
6	Note	

Physical References File Example

```
+0|CORNELL|OLINLIB|00000001|Boole, Mary Everest||Philosophy Of Algebra||

|0|1|00000002|5|1||   (File ref. #2 = Phys. ref. #5 = 600dpi TIFF image)
|0|2|00000003|5|2||   (File ref. #3 = Phys. ref. #5 = 100dpi TIFF image)
|0|3|00000004|6|1||   (File ref. #4 = Phys. ref. #6 = 600dpi TIFF image)
|0|4|00000005|6|2||   (File ref. #5 = Phys. ref. #6 = 100dpi TIFF image)
```

Note that in the above, it is guaranteed that file references 2 and 3 are two different versions of the same page, as are file references 4 and 5.

Logical Structure File

The Logical Structure file is the component of the document structure which offers "views" of a document and links images together logically to define documents. The file is actually an unloaded tree; when a document is "opened", the file is read and the tree reconstructed. By convention, all Logical Structure files contain one logical structure "PAGES" which defines the document by listing the pages in the order in which they appeared in the original document.

Document Structure lines

Field	Description	Comments
1	Parent structure number	Structure is a child of...
2	Sequence number	
3	Logical Structure name	Label for this structure
4	Structure number	Equal to Physical Reference file
5	Logical Children	# of logical children of this structure

Document Structure lines (continued)

Field	Description	Comments
6	Physical Children	# of physical children of this structure
7	References	# of references to this structure within this document (for how many structures is this a substructure)

Logical Structure File Example

```
|0|0|ROOT|0|4|0|0|            Structure 0, ROOT, has 4 logical children
|0|1|PAGES|1|100|0|1|         Str. 1, PAGES, has 100 logical children
|0|2|CONTENTS|2|22|0|1|       Str. 2, CONTENTS, has 22 logical children
                              ...has no physical children
 ...
|1|1|Production note|5|0|2|2| Str. 5 is child of structure 1
                              ...has a label "Production note"
                              ...has no logical children
                              ...has 2 physical references
                              ...is referenced twice in this document
|1|2||6|0|2|1|                Str. 6 has no label
|1|3||7|0|2|1|                Str. 7 has 2 physical references
|1|4||8|0|2|1|                Str. 8 is referenced only here
|1|5||9|0|2|1|                Str. 9 is 5th sequential child of PAGES
 ...
|1|99||103|0|2|2|
|1|100||104|0|2|2|
|2|1|Production note|105|1|0|1|          Str. 105 is a child of str. 2
|2|2|Title page|106|1|0|1|               Str. 106 has 1 logical child
|2|3|Table of contents|107|2|0|1|
|2|4|Chapter 1. From Arithmetic to Algebra|108|6|0|1|
|2|5|Chapter 2. The Making of Algebras|109|4|0|1|
|2|6|Chapter 3. Simultaneous Problems|110|4|0|1|
|2|7|Chapter 4. Partial Solutions...|111|3|0|1|
|2|8|Chapter 5. Mathematical Certainty...|112|3|0|1|
|2|9|Chapter 6. The First Hebrew Algebra|113|8|0|1|
|2|10|Chapter 7. How to Choose our Hypotheses|114|9|0|1|
|2|11|Chapter 8. The Limits of the Teachers Function|115|5|0|1|
|2|12|Chapter 9. The Use of Sewing Cards|116|4|0|1|
 ...
|2|20|Chapter 17. From Bondage to Freedom|124|5|0|1|
|2|21|Appendix|125|2|1|1|
|2|22|advertisements|126|4|1|2|
|105|1|Production note|5|0|2|2|          Str. 5 is a child of str. 105
|106|1|Title page|11|0|2|2|              2nd reference to str. 11
|107|1|7|15|0|2|2|
|107|2|8|16|0|2|2|
 ...
|126|4||104|0|2|2|
```

Implementation Details

The tuple <library ID>+<collection ID>+<document ID>+<filetype>+ <file reference> is guaranteed to locate a file. A file locator program will translate between this tuple and the fully-qualified path and file name in the underlying file system. While a library will always have a hierarchical nature corresponding to UNIX file systems, the order of the hierarchy will be flexible to accommodate optimization efforts. Each level of the hierarchy will have an INFO file that describes the order of the lower levels of the hierarchy. The file locator program will read these files as it navigates the directory structure of the file system when a library, collection, or document is opened. Two examples follow:

Example 1. Hierarchy is LIBRARY, COLLECTION, DOCUMENT, FILETYPE.

```
/<library name>
        LIBINFO.TXT                     Description of library
        /<collection name>
               COLINFO.TXT              Description of collection
               /<document ID>
                     DOCINFO.TXT        Description of document
                     LOGSTR.000         Logical structure file
                     PHYSREF.000        Physical reference file
                     /<filetype1>
                             00001.TIF
                             00002.TIF
                             ...
                     /<filetype2>
                             00001.TIF
                             00002.TIF
                             ...
```

Example 2. Hierarchy is LIBRARY, FILETYPE, COLLECTION, DOCUMENT.

```
/<library name>

        LIBINFO.TXT                         Description of library
        /<filetype1>
                /<collection name>
                       COLINFO.TXT          Description of collection
                       /<document ID>
                             DOCINFO.TXT    Description of document
                             LOGSTR.000     Logical structure file
                             PHYSREF.000    Physical reference file
                             00001.TIF
                             00002.TIF
                             ...
        /<filetype2>
                /<collection name>
                       COLINFO.TXT          Description of collection
                       /<document ID>
                             DOCINFO.TXT    Description of document
                             LOGSTR.000     Logical structure file
                             PHYSREF.000    Physical reference file
                             00001.TIF
                             00002.TIF
                             ....
```

This implementation involves some redundancy, but it permits complete copies of a collection to be mounted on different file systems for performance considerations. In particular, the second scheme would facilitate storing all low-resolution images on high-speed magnetic disk for fast access, and all high-resolution images on slower, less expensive storage. This will also facilitate authorizing access to low-resolution images by other software systems (FTP, Gopher) while restricting access to high-resolution images.

Security Considerations

Security issues are not discussed in this memo.

References

[1] Turner, W., "Cornell Digital Library Document Architecture, Version 1.1 - 3/22/94", Library Technology Department, Cornell University.

Author's Address

William Turner
Library Technology
502 Olin Library
Cornell University
Ithaca, NY 14853

Phone: 607-255-9098
Fax: 607-255-9346
EMail: wrt1@cornell.edu