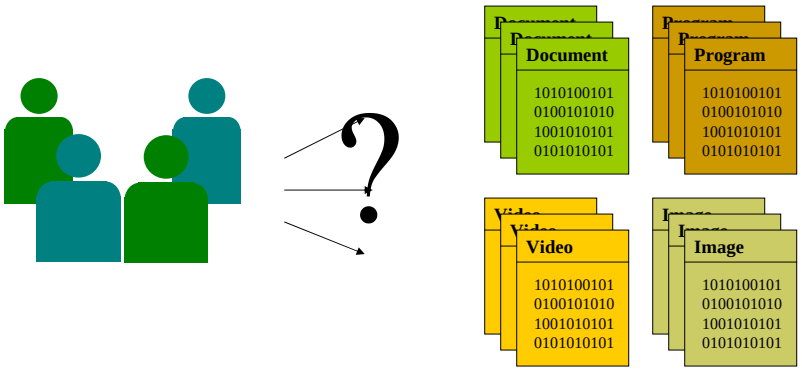
Introduction to Digital Libraries

hussein suleman
uct cs honours 2003

Open Digital Libraries: a Component Model

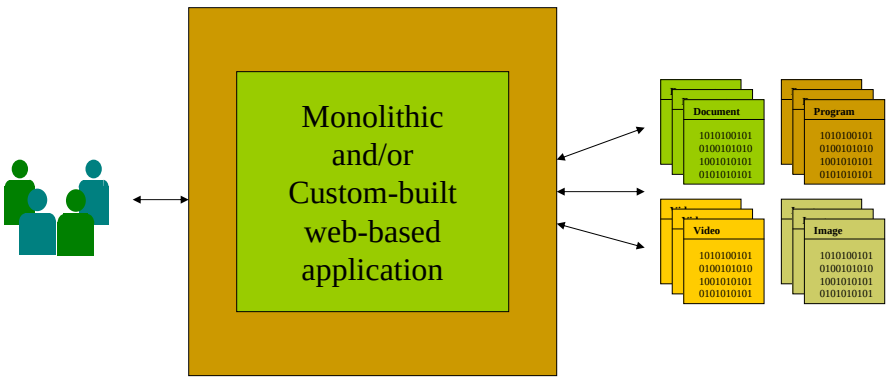
Introduction



users

digital objects

Introduction …



digital library

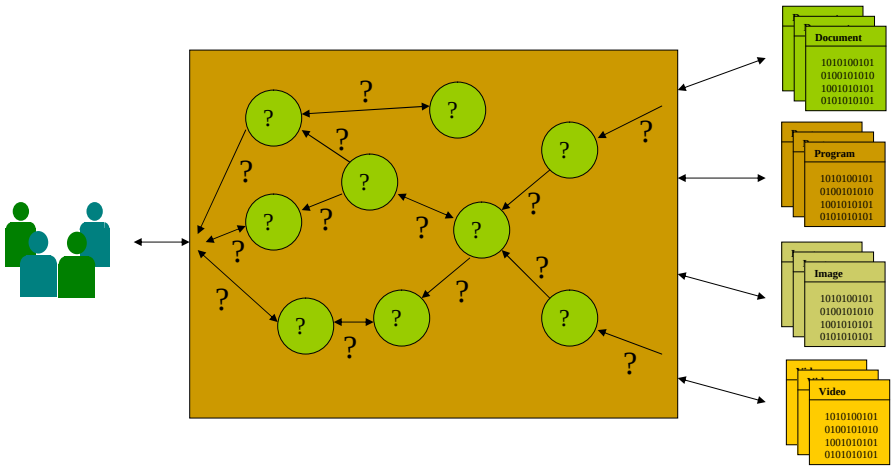
Problems

- Digital Libraries are difficult to build – lots of standards and evolving architectures
  - e.g., Dienst, EPrints
- Interoperability is hard
  - e.g., NCSTRL, Z39.50
- Software development is time-consuming
  - e.g., CSTC, WCR, EPrints

More Problems

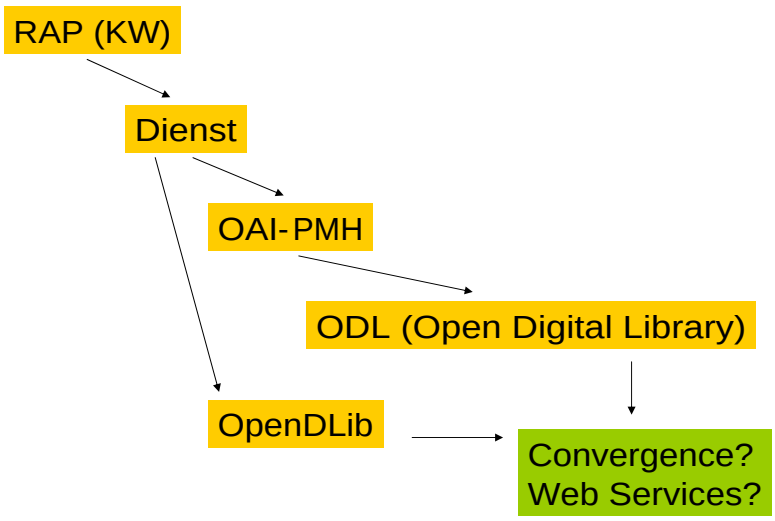
- Poor software engineering
  - Tight coupling
  - Too much complexity
  - Inadequate testing methods
- Lessons from Internet development ignored
  - Simplicity
  - Independence
  - Layering
  - etc.

Solution ?



componentised digital library

Some Component Architectures



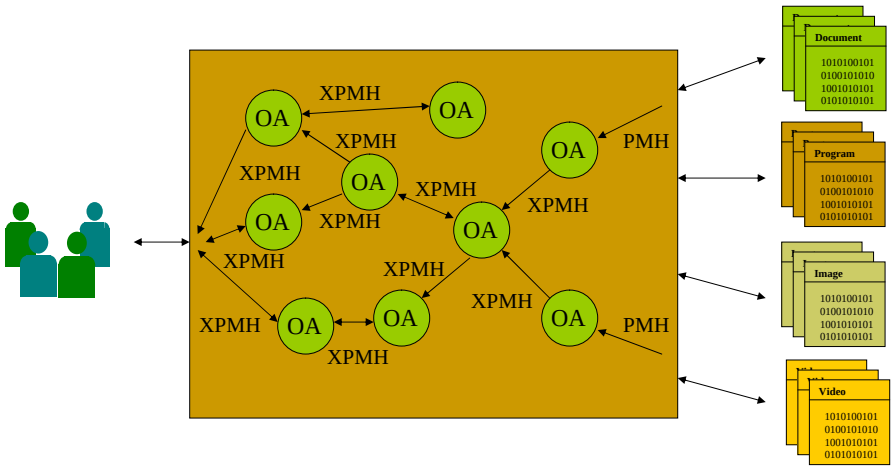
Open Digital Library (ODL)

- Digital Libraries can be modeled as networks of extended Open Archives, where each extended Open Archive is a source of data and/or a provider of services.
- Each component is independent and has well-defined external interfaces that are Web-based, e.g., OAI-PMH.

Open DL Design

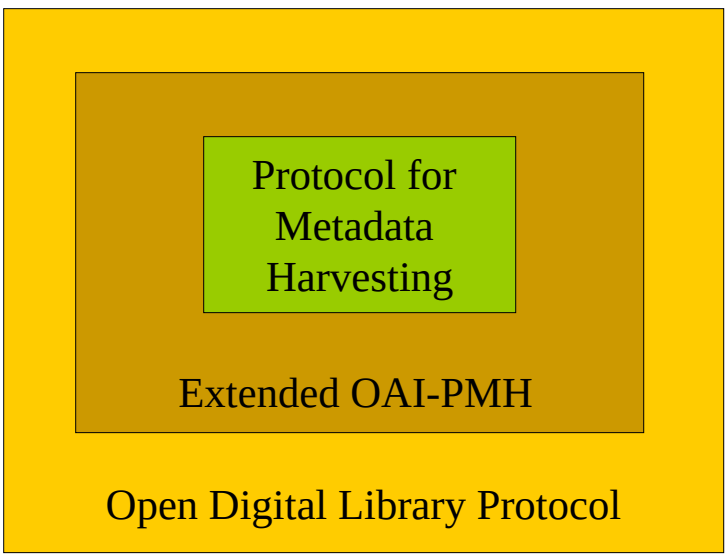
- Each component is encapsulated in an extended Open Archive.
- Communication with other components and user interfaces use specialised versions of the extended OAI-PMH (XOAI-PMH).
- Digital Libraries are constructed as networks of extended Open Archives.

Problem Revisited

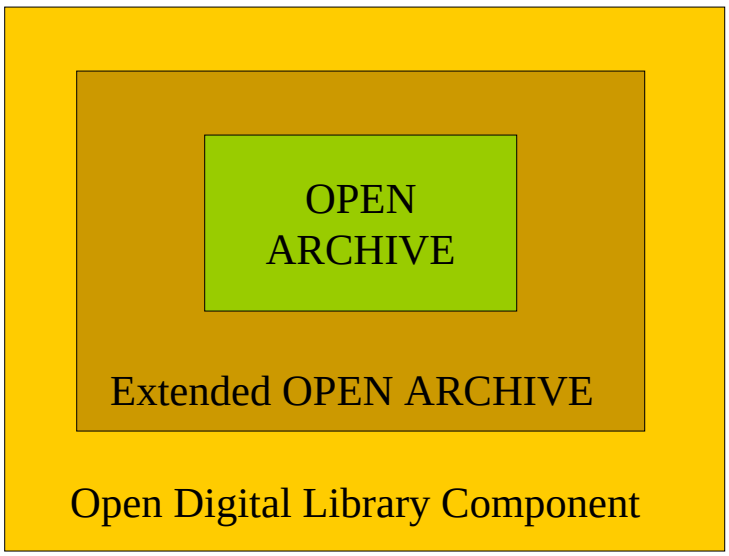


open digital library

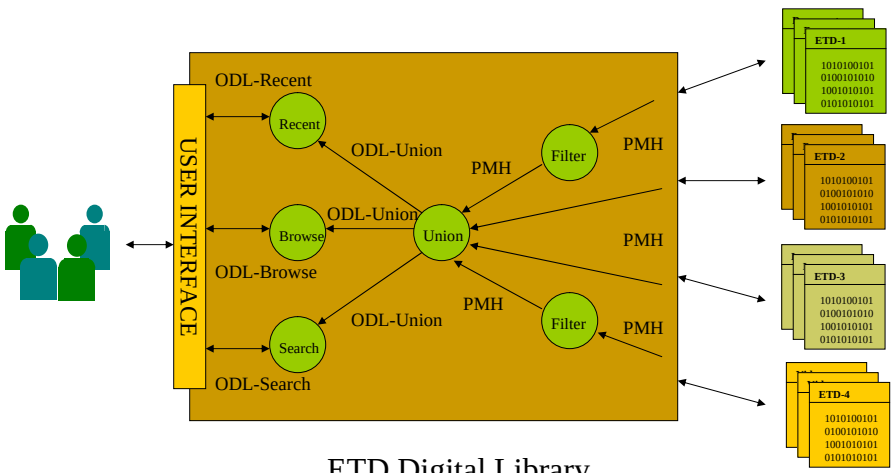
Protocol Layers



Component Layers



Example Open Digital Library



Students and researchers

ETD Digital Library

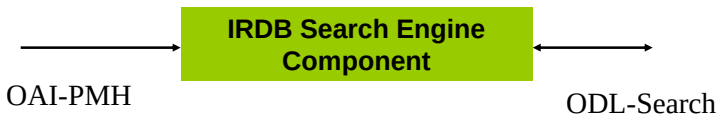
ETD collections

Protocols and Components

Protocol	Component	Description
ODL-Union	DBUnion	Merge archives together
ODL-Search	IRDB	Search engine
ODL-Browse	DBBrowse	Category-based browser
ODL-Recent	WhatsNew	Tracker for recent entries
ODL-Submit	Box	Archive supporting submit and retrieve operations
ODL-Annotate	Thread	Threaded annotation engine
ODL-Recommend	Suggest	Recommendation system
ODL-Rate	DBRate	Ratings manager
ODL-Review	DBReview	Peer review workflow manager

Example: IRDB Search Engine

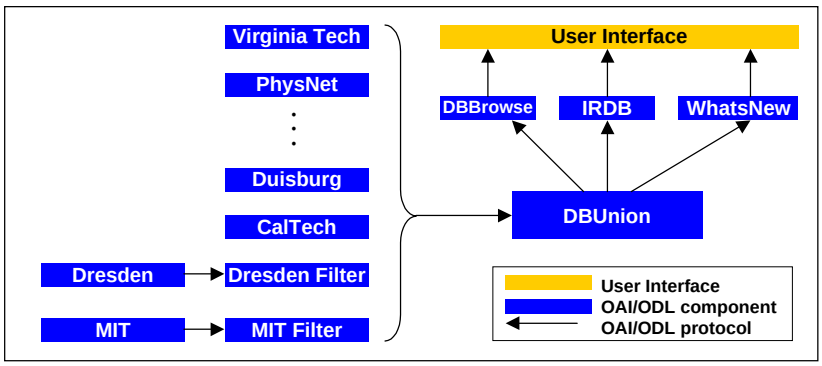
- Encapsulate search capability in an OA
- OAI-PMH to gather data for indexing
- ODL-Search to submit queries and get results



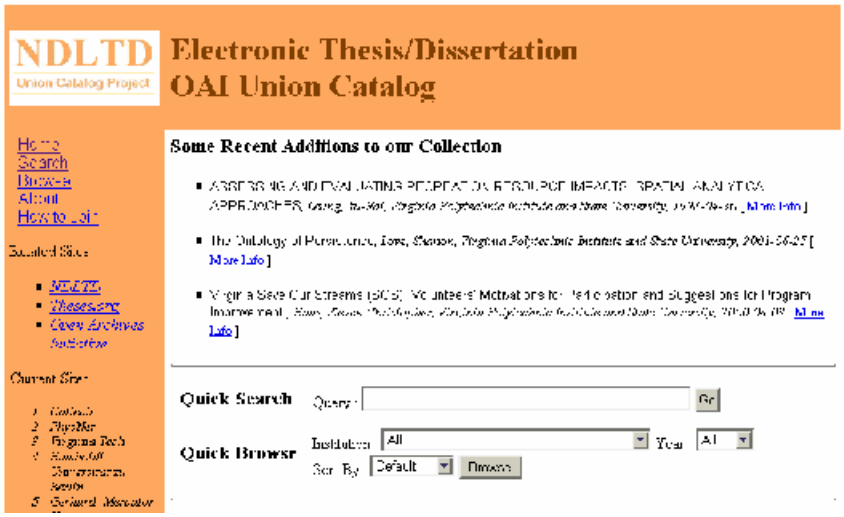
Example: ODL-Search Protocol

- Parameters
  - query - list of searchable keywords
  - query language – “odlsearch1”
  - start/stop - subset of ranked list
- Encoding
  - `verb=ListIdentifiers&set=odlsearch1/query/start/stop…`
  - `verb=ListRecords&set=odlsearch1/query/start/stop…`
- Results
  - Standard OAI response - list of identifiers or records
- Example
  - `verb=ListRecords&set=odlsearch1/computer science/1/10…`

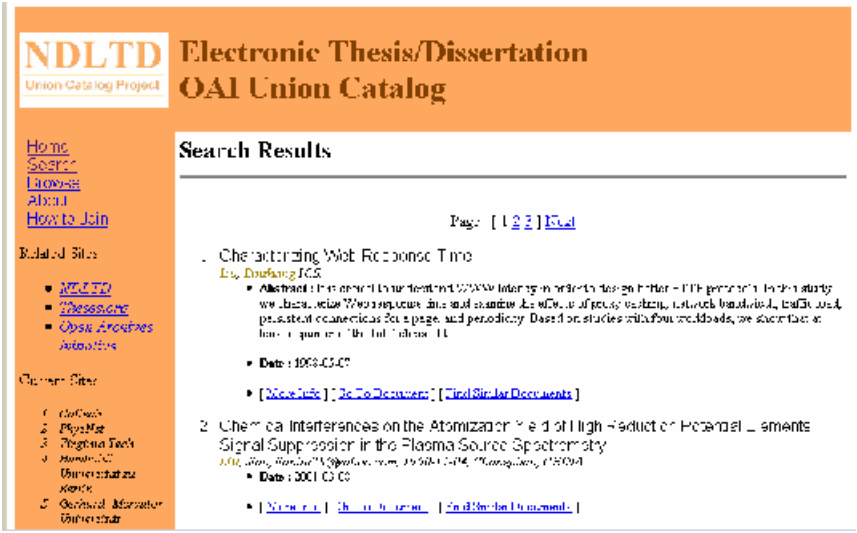
Case Study: ETD Union Catalog



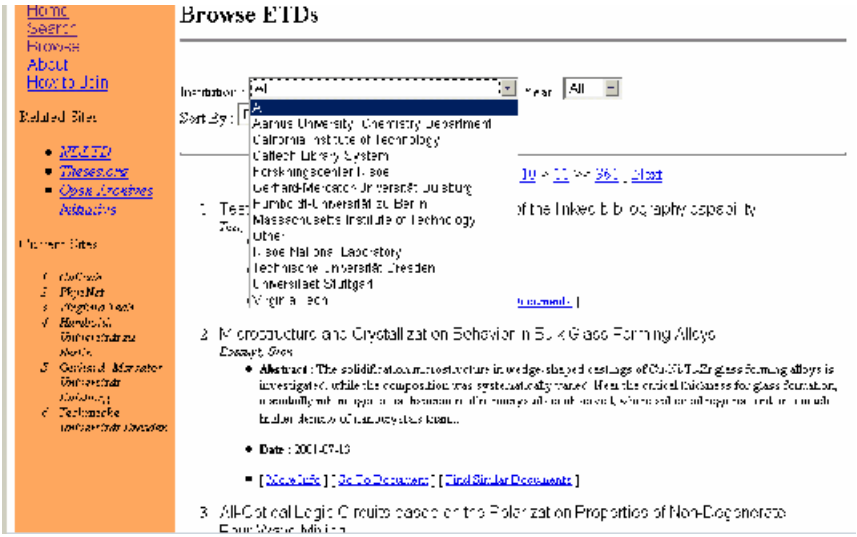
ETD Union Catalog - Front



ETD Union Catalog - Search



ETD Union Catalog - Browse



The Ultimate Goal

- Package different configurations of components into instant DL systems
- DL building = component configuration
- All DLs speak the same language(s)
- Basic services are trivial to provide so more effort is spent on advanced capabilities of DLs
- Information is more accessible to users

Repository+ Component Models

Repository Access Protocol (RAP)

- A repository can be defined as a network-accessible server.
- RAP specifies a simple interface to access and manage digital objects in a repository.
- RAP is an abstract model, with concrete implementations in the Dienst, OpenDLib, OAI and ODL projects.
- This is usually referred to as the "Kahn/Wilensky architecture".
  - does Kahn ring any bells?

RAP Operations

- ACCESS_DO
  - Return a manifestation (dissemination) of a digital object based on its identifier and a specification of what service is being requested.
- DEPOSIT_DO
  - Submit a digital object to the repository, assigning or specifying an identifier for it.
- ACCESS_REF
  - List services and their access mechanisms for the repository.

RAP: Naming of Digital Objects

- Each digital object must have a location-independent name (handle), made up of a repository identifier and a local name.
  - Example:
    - berkeley.cs/csd-93-712
    - where berkeley.cs is the repository and csd-93-712 refers to a technical report.
- Handles are resolved by a handle server to redirect a service provider to a repository containing an object identified only by its location-independent handle.

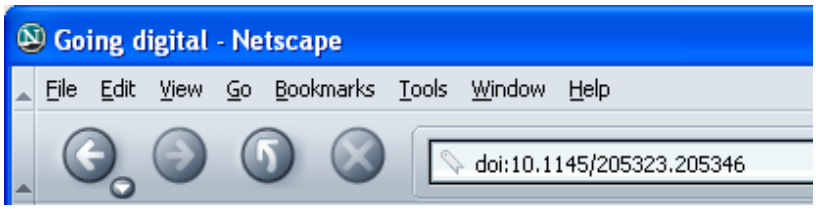
Handle Servers

- A handle server stores the association between handles and physical locations of objects.
- Handle servers follow a DNS model:
  - they are distributed and replicated
  - there are global and local servers
  - handles may be cached locally after being resolved to minimise resolution traffic
  - management of servers/handles requires an authority system for management, accountability, delegation, etc.

Handle Example

Digital Object Identifiers (DOIs)

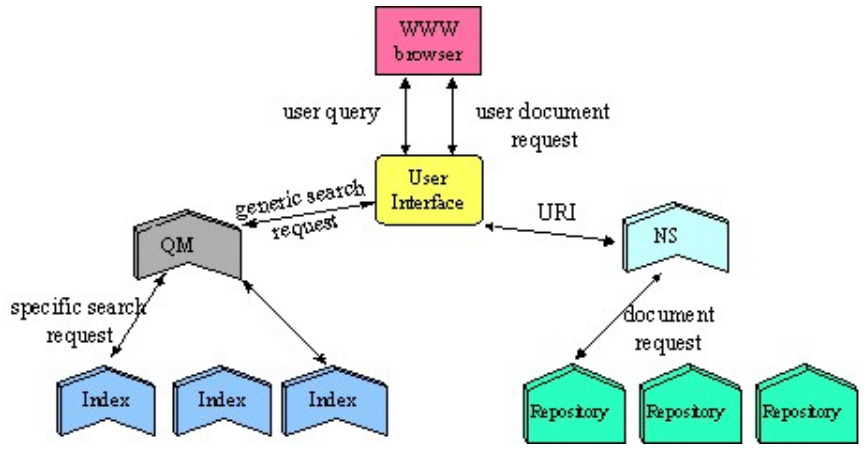
- DOIs are a standardised implementation of the handle concept.
- Handles/DOIs are URIs that refer to digital objects while URLs are URIs that refer to network services.
- Handle/DOI resolution can be performed transparently using a browser plug-in.



Dienst

- Dienst (German for "service") is a suite of protocols and components to build distributed digital libraries.
- Dienst is the software suite that supported document management at each of the older NCSTRL (Networked Computer Science Technical Reference Library) sites, and transparently linked them into an international federation of sites.
- Dienst uses federation for interoperability, with a "backup server" for robustness.

Dienst Service Architecture Example



from Dienst website at http://www.cs.cornell.edu/cdlrg/dienst/architecture/architecture.htm

Dienst Example

- Example Request:
  - List the handles in the high energy (hep) partition within the physics partition.

```
/Dienst/Repository/4.0/List-Contents?partitionspec=physics;hep
```

- Example Response:

```
<?xml version="1.0" encoding="UTF-8"?>
  <List-Contents version="4.0">
    <record>
      handlecorp/970101
    </record>
    <record>
      handlecorp/970102
    </record>
  </List-Contents>
```

from Dienst website at http://www.cs.cornell.edu/cdlrg/dienst/protocols/DienstProtocol.htm

Dienst → OAI-PMH

- Dienst formed the foundation for the current OAI-PMH – hence the terminology is sometimes similar.
- NCSTRL has moved to a model based on harvesting and OAI-PMH is being used to connect sites together. In 2001, data from the existing NCSTRL sites was harvested and archived (for preservation) using an early version of an ODL component!
  - see http://www.ncstrl.org

Dienst → OpenDLib

- OpenDLib is a component model similar to ODL, but based on Dienst rather than OAI-PMH.
- OpenDLib attempts to define services (mediators) and repositories based on Dienst and updated best practices in DLs.
- OpenDLib uses a well-defined document model for structured content: the Document Model for Digital Libraries (DoMDL).

Other repository/component models

- FEDORA (Flexible Extensible Digital Object and Repository Architecture) defines a generic interface to manage digital objects at a lower layer in an information system.
  - see http://www.fedora.info/
- SODA (Smart Objects Dumb Archive) packages digital objects into buckets containing the data along with the code to mediate access, display the objects, enforce rights, etc.

References

- Suleman, H. and E. A. Fox (2001) "A Framework for Building Open Digital Libraries", in D-Lib Magazine, Vol 7., No. 12, December 2001. Available http://www.dlib.org/dlib/december01/suleman/12suleman.html
- Kahn, Robert and Robert Wilensky (1995) "A Framework for Distributed Digital Object Services", CNRI. Available http://www.cnri.reston.va.us/home/cstr/arch/k-w.html
- Lagoze, Carl and James Davis (1995) "Dienst: an architecture for distributed document libraries", Communications of the ACM, ACM, Vol. 38, No. 4, p. 47.
- Castelli, Donatella and Pasquale Pagano (2002) "OpenDLib: A Digital Library Service System", in Proceedings of Research and Advanced Technology for Digital Libraries: 6th European Conference (ECDL 2002), Rome, Italy, September 2002, Lecture Notes in Computer Science 2458, p. 292-307. Maristella Agosti, Costantino Thanos (eds.). Springer, 2002.
- Maly, Kurt, Michael L. Nelson and Mohammed Zubair (1999) "Smart Objects, Dumb Archives: A User-Centric, Layered Digital Library Framework", in D-Lib Magazine, Vol. 5, No. 3, March 1999. Available http://www.dlib.org/dlib/march99/maly/03maly.html