Digital Library Curriculum Development
Module 5-d: Protocols
(Last Updated: 2008-11-24)

1. **Module name:** Protocols

2. **Scope**

   This module addresses the concepts, development and implementation of digital library protocols and covers the roles of protocols in Information retrieval systems (IR) and Service Oriented Architectures (SOA).

3. **Learning objectives**

   By the end of this lesson, the student will be able to:

   a. Explain the basic concepts and methods related to DL protocols.

   b. Design and implement a protocol for harvesting metadata from a digital library.

   c. Design and implement a digital library service based on the principles and guidelines of DL protocols.

4. **5S Characteristics of the module**

   - Stream: Protocols enable the data in digital libraries to be harvestable. Protocols specify the flows of streams of data.
   - Structure: Protocols enable digital libraries to expand and disseminate digital objects and provide various information organization services.
   - Spaces: Protocols connect spatially separate digital libraries.
   - Scenario: Different protocols implement specific scenarios of communication among digital libraries.
   - Society: Different protocols implement and design for different use cases and communities.

5. **Level of effort required**

   a. Class time: 3 hour

   b. Student time outside class: 8 hours

      - Reading before the class starts: 4 hours
      - Homework assignment: 4 hours

6. **Relationships with other modules**

   a. 4-b: DL protocols are harvesting metadata stored in a digital library. 4-b should be taught before this module.

   b. 5-a: This module should be taught around the same time in the semester.

   c. 5-b: This module should be taught around the same time in the semester.

7. **Prerequisite knowledge required**:

   a. Knowledge of client-server, P2P, SOA architectures

   b. Understanding of client request and server response

   c. Basics of other networking protocols

8. **Introductory remedial instruction**: Quick overview of software engineering architectures and basics of simple networking protocols.

9. **Body of knowledge**

   1. Introduction: This module discusses the following digital library protocols:

      a. OAI-PMH

      b. SOA

      c. P2P

      d. VIDI

      e. Z39.50

      f. DLIOP

   Later in the module parallel and distributed information retrieval is discussed.

   2. Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)

      a. OAI-PMH version history

         i. It is a protocol developed by the Open Archives Initiative.

         ii. The Santa Fe Convention led to the first incarnation of the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH).

         iii. OAI-PMH 1.0 introduced the unqualified Dublin Core element set as a baseline for metadata interoperability. OAI-PMH 1.1 was a revision of the 1.0 specification taking account of changes to the emerging XML Schema specification. Both v.1.0 and 1.1 were experimental in nature.

iv. OAI-PMH 2.0 is a major revision of the protocol and it is a stable protocol.

b. Flexible deployment:
   i. Multiple service providers can harvest from multiple data providers.
   ii. Aggregators can sit between data providers and service providers.
   iii. The harvesting approach can be complemented with searching.

c. Main ideas of OAI
   i. world-wide consolidation of scholarly archives
   ii. free access to the archives
   iii. consistent interfaces for archives and service providers
   iv. low barrier protocol / effortless implementation

d. Structure model
   i. It is a simple protocol based on HTTP and XML.
   ii. This protocol is carried within HTTP POST or GET methods.

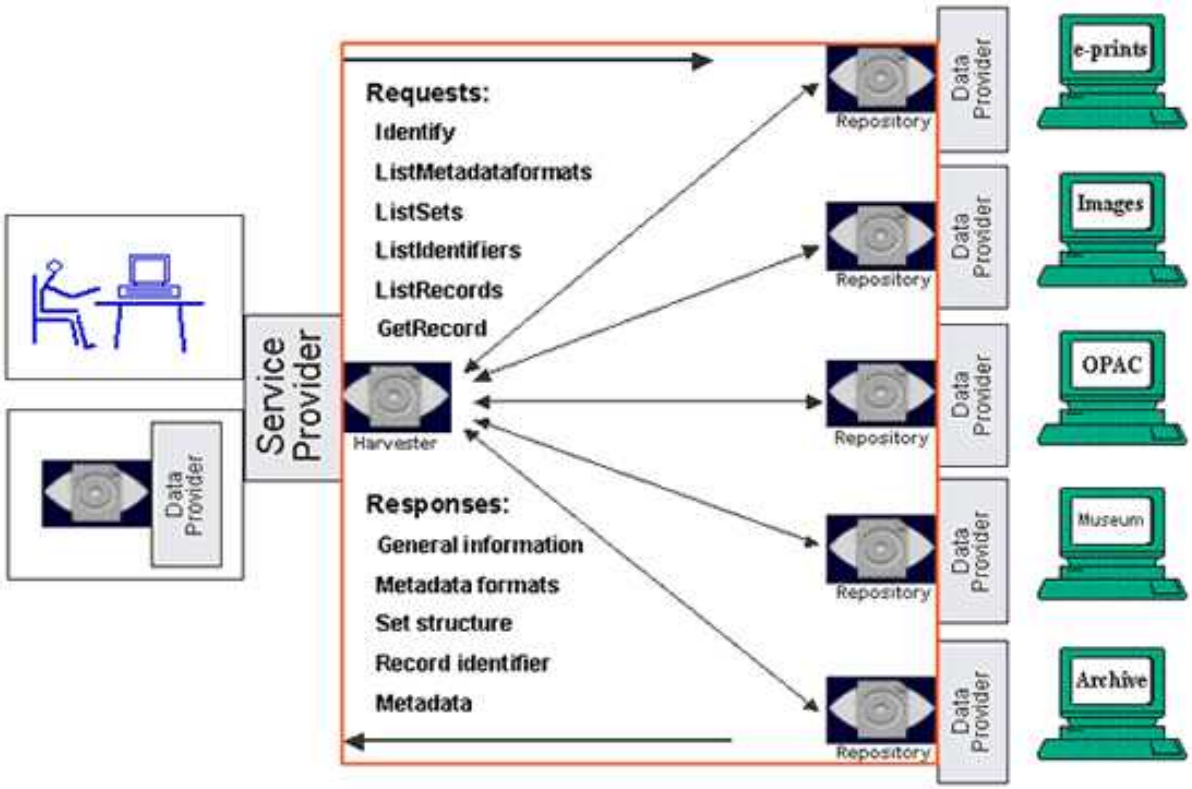

Figure 1. OAI-PMH structure model

e. Protocol details
   i. **Records**
   A record is the metadata of a resource in a specific format. A record has three parts: a header and metadata, both of which are mandatory, and an optional about statement.
   ii. **Datestamps**
   A datestamp is the date of last modification of a metadata record. Datestamp is a mandatory characteristic of every item. It has two

possible levels of granularity:
YYYY-MM-DD or YYYY-MM-DDThh:mm:ssZ.

iii. **Metadata schema**
OAI-PMH supports dissemination of multiple metadata formats from a repository. The properties of metadata formats are:
  – ID string to specify the format (**metadataPrefix**)
  – metadata schema URL (XML schema to test validity)
  – XML namespace URI (global identifier for metadata format)

iv. **Sets**
Sets enable a logical partitioning of repositories. They are optional.

v. **Request format**
Requests must be submitted using the **GET** or **POST** methods of HTTP, and repositories must support both methods. At least one key=value pair: verb=RequestType must be provided. Additional key=value pairs depend on the request type.

vi. **Response**
Responses are formatted as HTTP responses. The content type must be text/xml. HTTP-based status codes, as distinguished from OAI-PMH errors, such as 302 (redirect) and 503 (service not available) may be returned. Compression codes are optional in OAI-PMH, only identity encoding is mandatory. The response format must be well-formed XML with markup

vii. **Flow control**
OAI-PMH supports partitioning. Those managing a repository make the decisions on partitioning: whether to partition and how.

viii. **Errors and exceptions**
Repositories must indicate OAI-PMH errors by the inclusion of one or more **error** elements.

ix. **Request types**
There are six different request types:
- Identify
- ListMetadataFormats
- ListSets
- ListIdentifiers
- ListRecords
- GetRecord

f. **Implementing protocol**

i. Before implementing
- Data Provider:
  - Which data do you want to deliver?
  - Which Service Providers do you want to provide with data?
- Service Provider:
  - From which Data Providers do you get the metadata?
  - Which services do you want to provide, and to whom?

    - What kind of metadata do you want to provide through your services?
- Other things one also needs to consider:
    - Update frequency
    - Metadata format
    - Subject schema

ii. Data Provider

- Requisites:
    - Web server
    - Programming interface / API
    - Archive identifier / base URL
    - Metadata format
    - Datestamps for metadata
    - Unique identifier for each item
    - Logical set hierarchy
    - Flow control
- Components:
    - **Argument Parser** validates OAI requests.
    - **Error Generator** creates XML responses with encoded error messages.
    - **Database Query / Local Metadata Extraction** retrieves metadata from the repository, according to the required metadata format.
    - **XML Generator / Response Creation** creates XML responses with encoded metadata information.
    - **Flow Control** realizes incomplete list sequences for 'larger' repositories. It uses resumption tokens as the control mechanism.

iii. Service Provider

- Requisites:
    - An Internet-connected server
    - A database system on equivalent
    - A programming environment
- Components and architecture:
    - **Archive management** involves the selection of repositories to be harvested.
    - **Request Component** creates HTTP requests and sends them to OAI repositories (Data Provider).
    - **Scheduler** realizes timed and regular retrieval of the associated archives.
    - **Flow Control** is implemented via resumption tokens, partitioning of the result list into incomplete sections with a new request to retrieve more results.

- **Update Mechanism** realizes the consolidation of metadata which have been harvested earlier (merge old and new data).
- **XML Parser** analyses the responses received from the repositories, with validation using the XML schema, and transforms the metadata encoded in XML into the internal data structure.
- **Normaliser** transforms data in different metadata formats into a homogenous structure.
- **Database** receives the output of the normaliser mapping the XML structure of the metadata into a relational database that will handle multiple values of elements. An alternative is to use an XML database.
- **Duplication Checker** merges identical records from different data providers.
- **Service Module** provides the actual service to the 'public'. The basis for a service provided is the harvested and stored records of the associated archives.

3. Service Oriented Architecture (SOA) for Digital Library
    a. Simple definition: SOA provides methods for system development and integration where systems group functionality around business processes and package these as interoperable services
    b. The development of SOA
        i. It is an architecture developed with XML and Web Services. It is commonly built using Web Services standards.
        ii. SOA mostly refer to a platform that could realize the concept of partitioning an enterprise into a series of autonomous services.
    c. SOA deployment
        i. Run over standard Web protocols: SOA uses XML and HTTP packaging.
        ii. Communication: SOA uses SOAP (Simple Object Access Protocol) to exchange information, uses WSDL (Web Service Definition Language) to describe Web Services and uses UDDI (Universal Description Discovery and Integration) to register Web Services.
    d. SOA for digital library
        i. Services Technology is widely used in Digital Libraries: Fedora, DSpace and CiteSeer
        ii. The requirement for mapping a Digital Library application/service into web service:
            - An API interface to that application/service
            - A standard access layer such as SOAP
            - A layer describing the service in a standard fashion encoded using WSDL
        iii. Role of digital library: Service provider.
    e. Case Study: Fedora

i. Fedora Repository system is a Web Service:
   - Access/Search (API-A) and Management (API-M) use web service
   - Service descriptions use WSDL
   - Fedora uses both SOAP and HTTP bindings
   - Fedora acts as mediator to these services

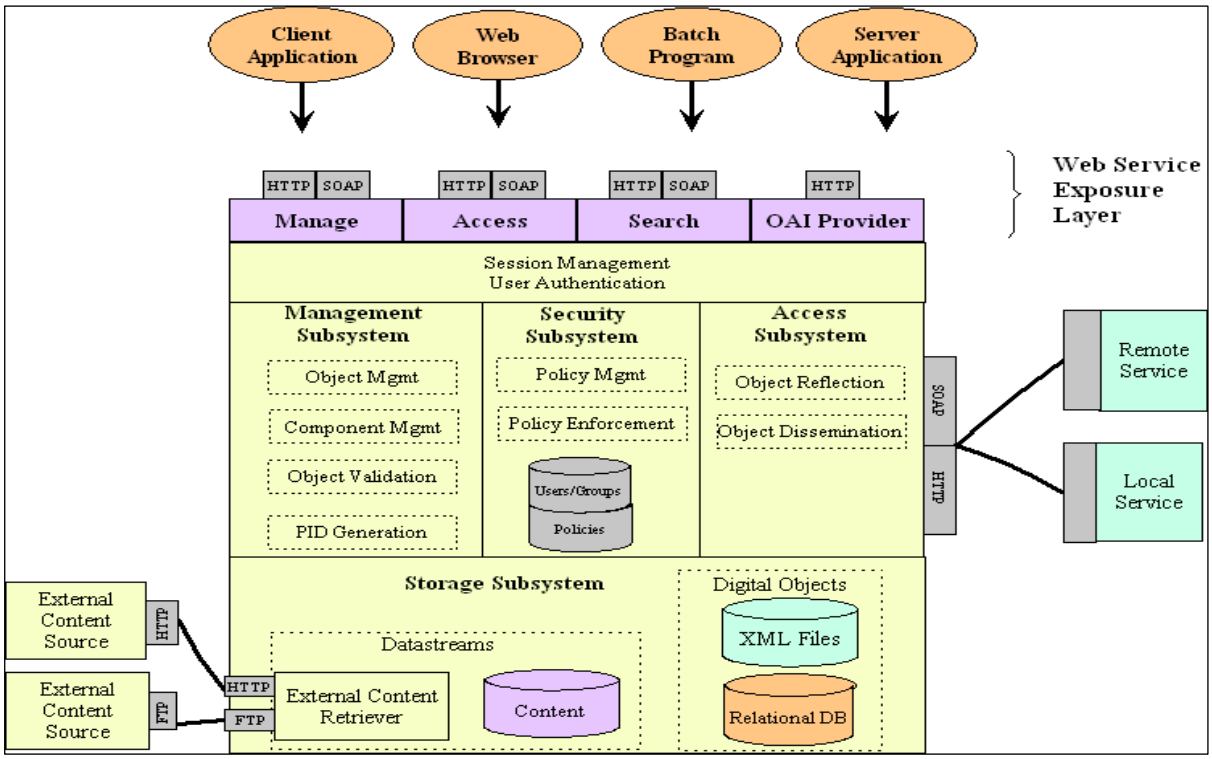

Figure 2. Fedora Repository System

ii. Fedora Repository Service Interfaces
   - Access Service (API-A and API-A-LITE)
     - Search - search repository for objects
     - Object Reflection - what disseminations can the object provide?
     - Object Dissemination - request a view of the object's content
   - Management Service (API-M)
     - Ingest - XML-encoded object submission
     - Create - interactive object creation via API requests
     - Maintain - interactive object modification via API requests
     - Validate – application of integrity rules to objects
     - Identify - generate unique object identifiers
     - Security - authentication and access control
     - Preserve - automatic content versioning and audit trail
     - Export - XML-encoded object formats

4. Peer-to-Peer Computing (P2P Computing) for Digital Library
    a. Simple definition:
        i. Peers act as clients and servers
        ii. Peers communicate directly
    b. The development of P2P
        i. Unstructured P2P (such as Gnutella, Napster) to structured P2P (such as Pastry, Chord)
        ii. P2P used for provide resources: services, for example, storage services, computational services, streaming media services
    c. P2P for Digital Library
        i. Main idea
            - Build Digital Library on the P2P structure, use P2P to store digital objects
            - Use P2P to share and route digital documents
        ii. Implementation
            - Automatically digital documents can be discovered and routed
            - Uniform client-level metadata query services that are compatible with heterogeneous underlying collections
            - Sharing of digital documents on P2P structure rather than storing the documents on a center server
    d. Case Study: The ADEPT Digital Library Architecture
        i. ADEPT (Alexandria Digital Earth Prototype) architecture
            - A framework for building distributed digital libraries of georeferenced information
            - Distributed, heterogeneous, scalable
        ii. Objects
            - Items: Items are fundamental objects in ADEPT, referring to digital objects. Items only have identity, no other innate properties
            - Collections: Collections are set of items, information about individual items is maintained at the collection level
            - Libraries: Libraries are set of collections, which expose a single standard set of interfaces to the collections
        iii. ADEPT Collection Discovery Service
            - ADEPT takes unstructured P2P mechanism: Reason: simple, sufficiently manageable and scalable
            - ADEPT uses central CDS server and Register server.
            - Collection registry polls known library CDS server

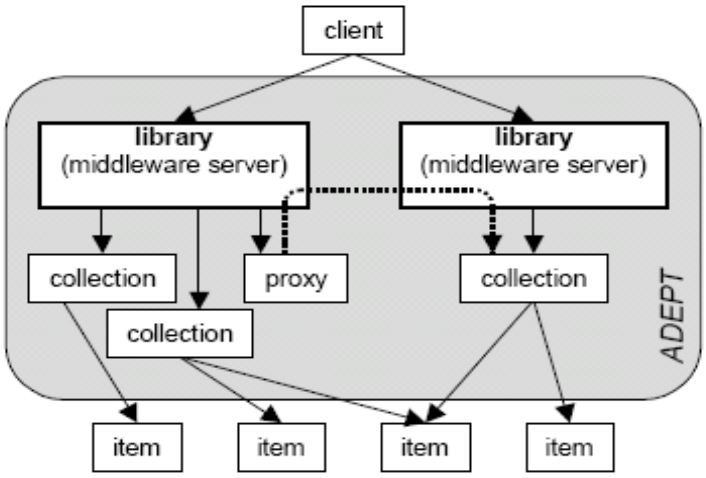

Figure 3. ADEPT Architecture

5. VIDI
   a. Simple definition: A lightweight protocol between visualization systems and digital libraries.
   b. VIDI Development
      i. Extended from OAI Protocol for Metadata Harvesting
      ii. Based on Digital Library likes MARIAN and Visualization Systems likes Infosphere
   c. Protocol

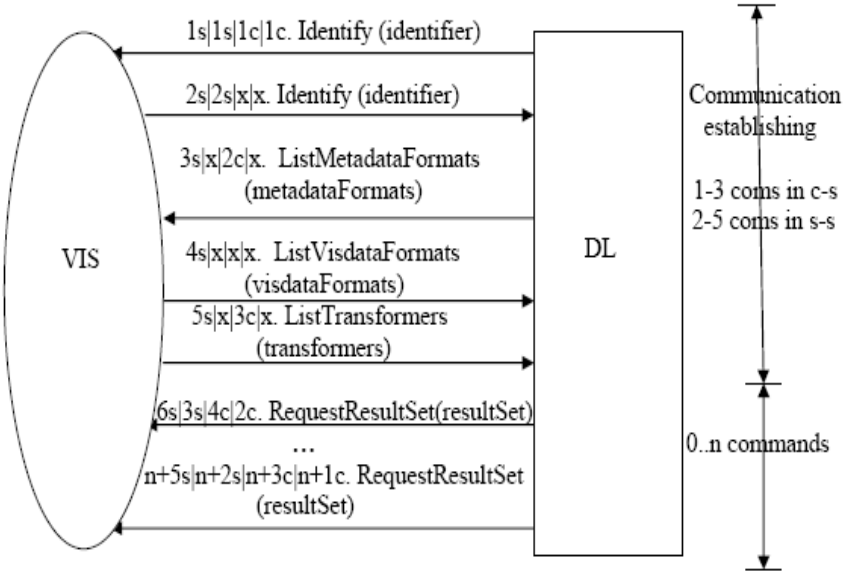

Figure 4. Flow graph of VIDI protocol

      i. The labels:

- ‘s’ means server-server structure
- ‘c’ means client-server structure
- ‘x’ means in this specific scenario, this command is unused

ii. Four scenarios from left to right
- Most commands in server-server structure
- Least commands in server-server structure
- Most commands in client-server structure
- Least commands in client-server structure

iii. Most commands in client-server structure scenarios

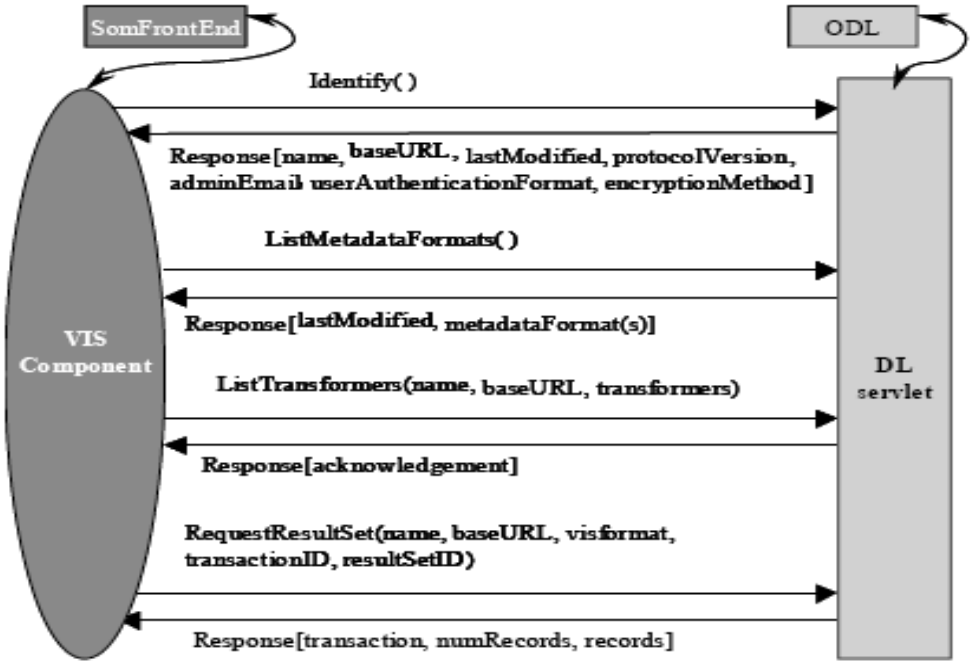

Figure 5. Most commands in client-server structure scenarios in VIDI protocol

d. VIDI Implementation

i. Use of HTTP Request and Response
- The VIDI protocol requests are expressed as HTTP requests
- HTTP requests may be expressed using either HTTP GET or POST

ii. Protocol Entities Definitions: Metadata Format, Visdata format, Transformer, Result Set, Result Record, Result Set Identifier, Unique Identifier

iii. Dates and Times
- The date used in VIDI protocol is encoded using “Complete date” variant of ISO8601, the format is YYYY-MM-DD

- The time used in VIDI protocol is encoded using the “complete date plus hours, minutes and seconds” variant of ISO8601, the format is YYYY-MM-DDThh:mm:ssTZD

iv. VIDI Commands

- Identify: Retrieve system information about a DL or VIS
- ListMetaFormats: Retrieve the metadata format available from a DL
- ListVisdataFormats: Retrieve the metadata format available from a VIS
- ListTransformers: Retrieve the transformers that VIS supports in order to transform the metadata format to the visdata format
- RequestResultSet: Transfer query data

6. Special Protocols: **Z39.50**

a. Introduction:

i. A client server protocol for searching and retrieving information from remote computer databases
ii. Covered by ANSI/NISO (National Information Standards Organization) standard Z39.50
iii. Accepted by ISO (International Standards Organization) as standard 23950
iv. The standard is maintained by the Library of Congress (LOC)
v. First version of Z39.50 in 1988, Z39.50 – 1992 (version 2), Z39.50 – 1995 (version 3)

b. Important components:

i. **Target:** Implements the abstract database and is a ready made server module
ii. **Gateway:** A program that has two interfaces. First, it acts as Origin to a Z39.50 Target. Second, it handles communication with a client application. Client protocol may be HTML, Telnet, Z39.50, etc. Advanced Gateway like the one in the figure below can connect to several Z39.50 targets for parallel search, serial search, merging of results. Advanced Gateways can handle several different protocols on both interfaces SQL, LDAP, HTML, DNS, etc.
iii. **Origin:** Normally part of graphical client which hides complexity from the user. It can access several targets simultaneously. There are clients with a “raw” Origin interface

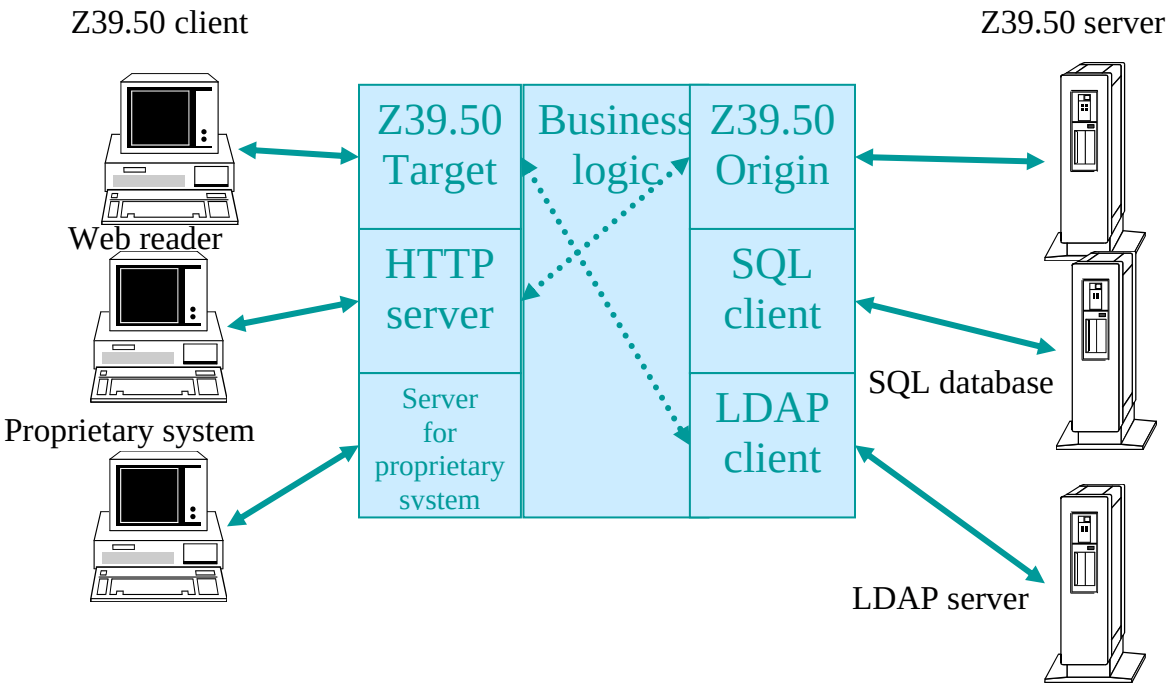

Figure 6. Advanced Gateway

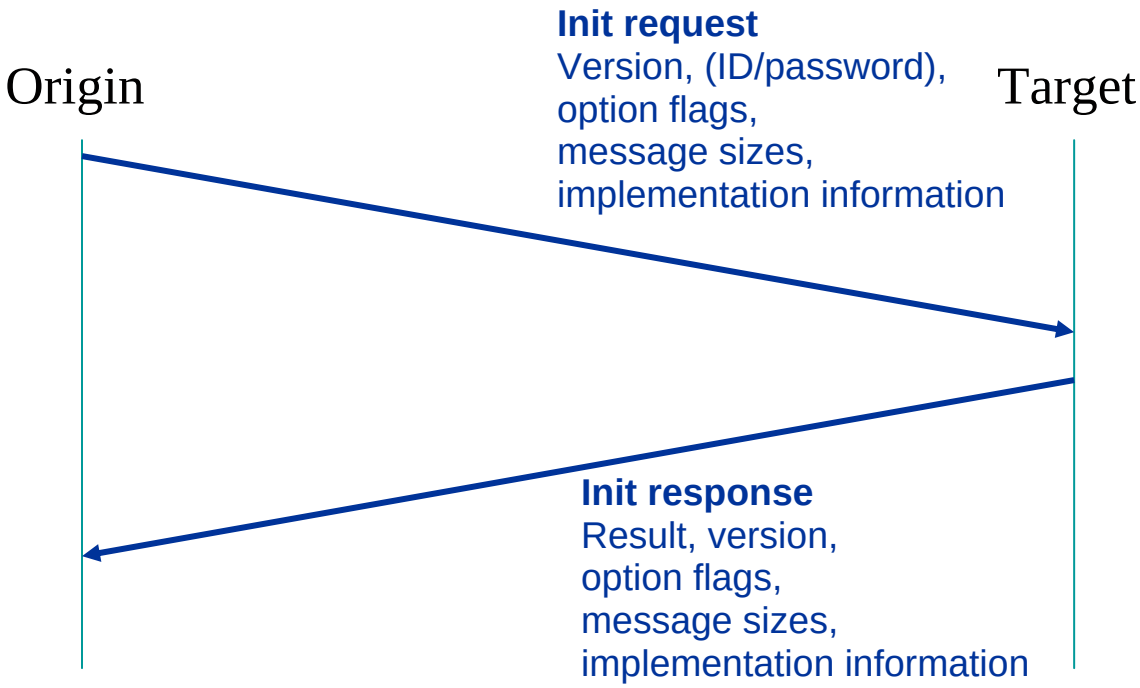

Figure 7. Initialization Facility – establishes Z association

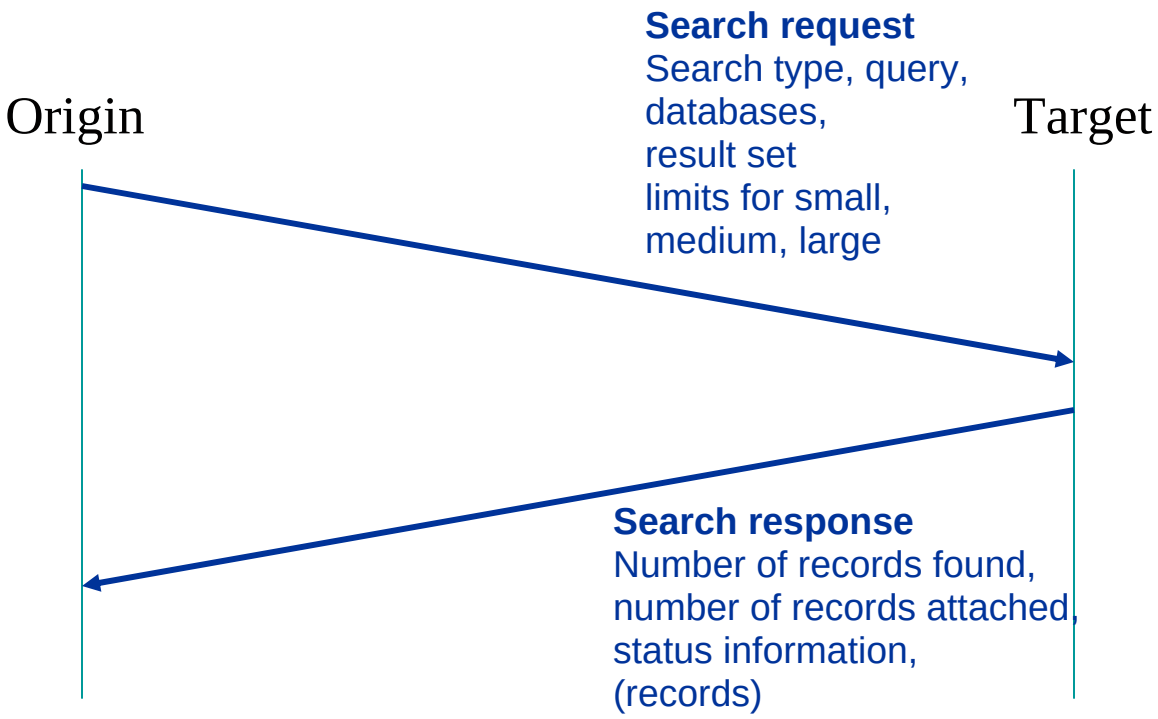

Figure 8. Search Facility

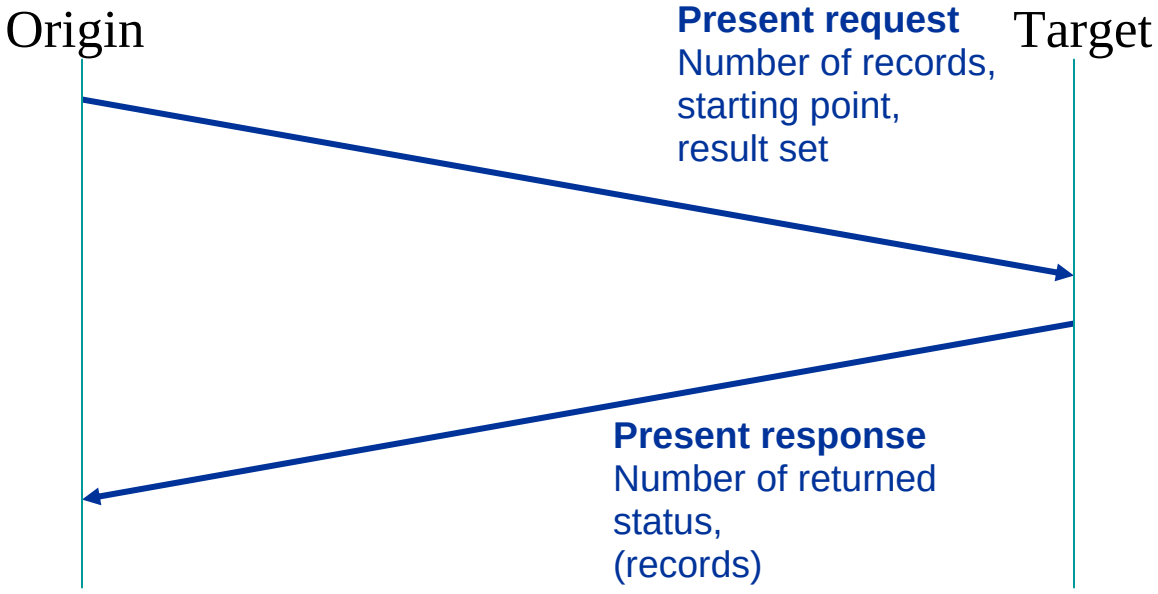

Figure 9. Retrieval Facility

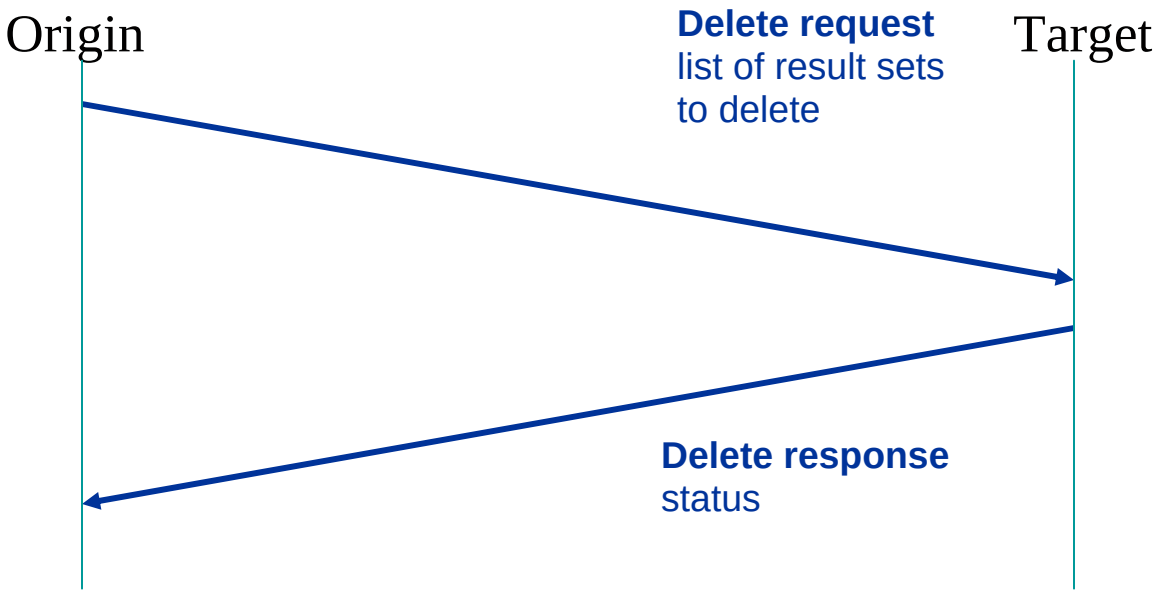

Figure 10. Result-set-delete Facility

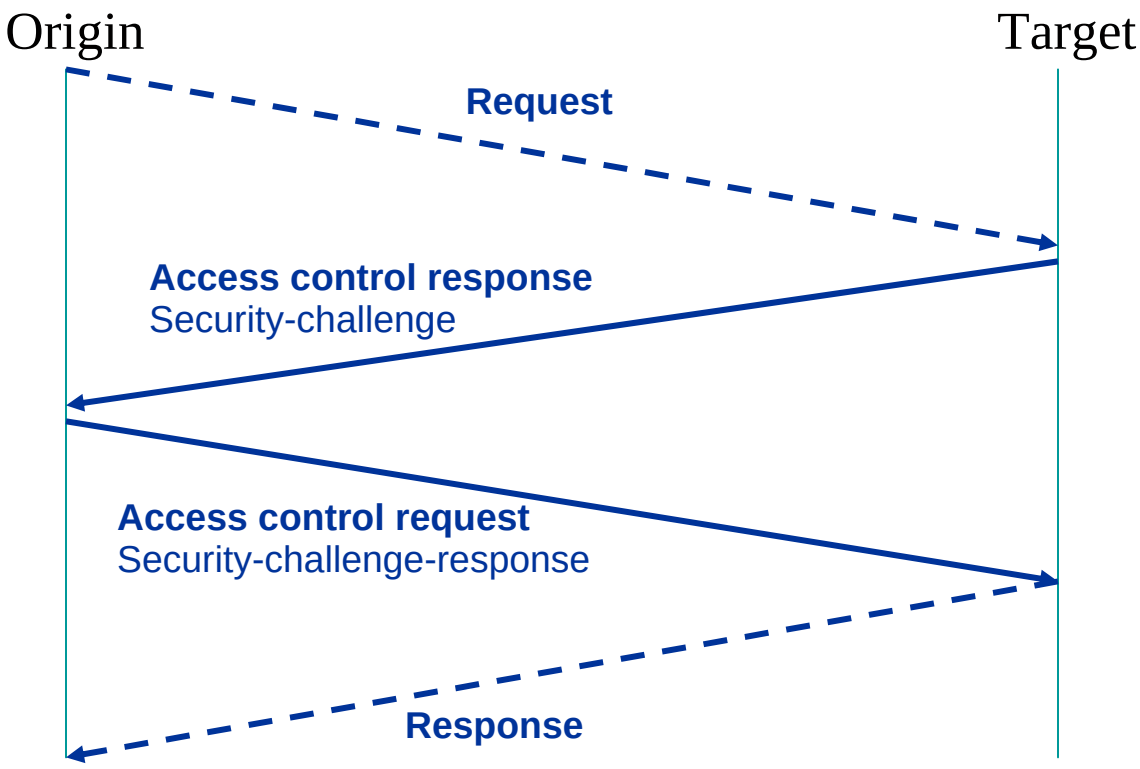

Figure 11. Access Control Facility

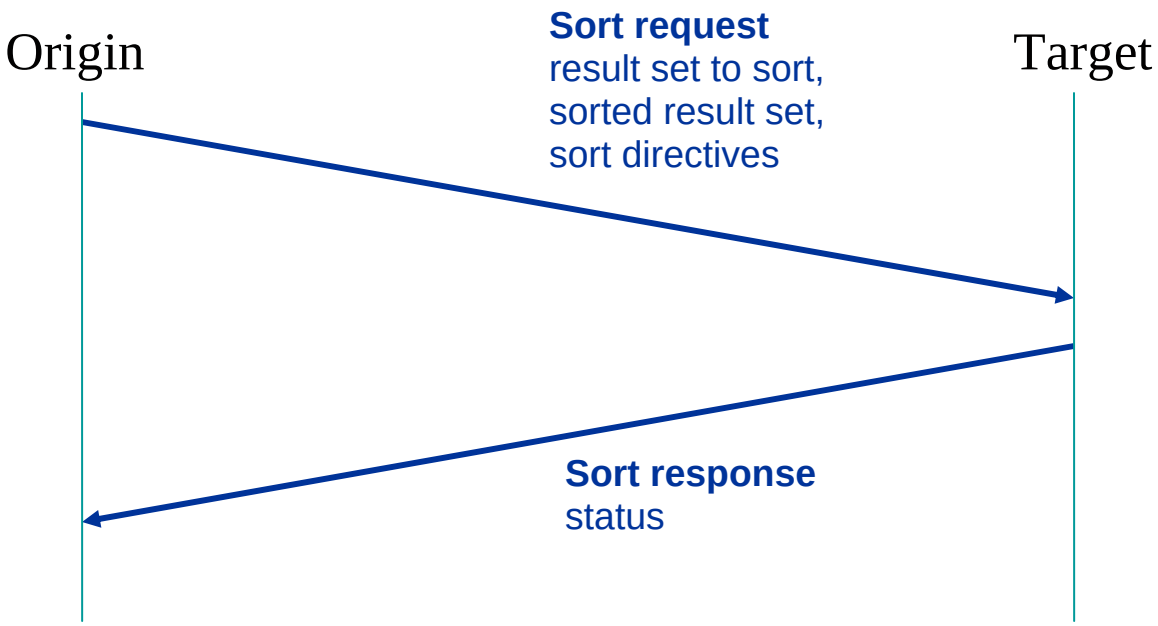

Figure 12. Sort Facility

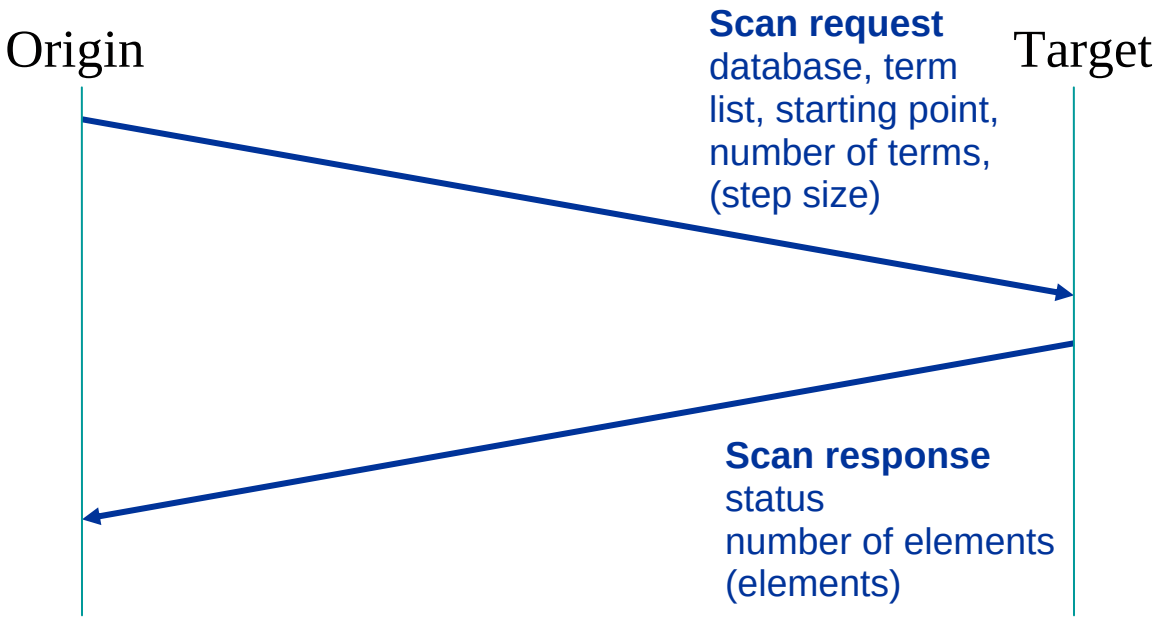

Figure 13. Browse Facility

7. **Special Protocols**: Stanford's Digital Library Interoperation Protocol (**DLIOP**)
   a. **Stanford Digital Library test bed**
      i. Stanford Digital Library test bed was a platform for experimentation with interoperation among online services.
      ii. The figure 13 shows the InfoBus, central to the architecture of the Stanford test bed. It is based on a hardware bus metaphor to

suggest that services, repositories, and clients are 'plugged in', and interoperate by taking advantage of interoperability mechanisms built into the testbed.

iii. DLIOP is an asynchronous protocol, providing robustness in the face of network or server outages

iv. The approach is to use distributed objects to allow integrated access to heterogeneous service across networks.

v. The distributed approach allows the interaction of processes on different machines, with different architectures, implemented in different languages.

vi. It uses CORBA to provide communication between remote processes. Xerox PARC's ILU, a free implementation of a CORBA superset, is used.

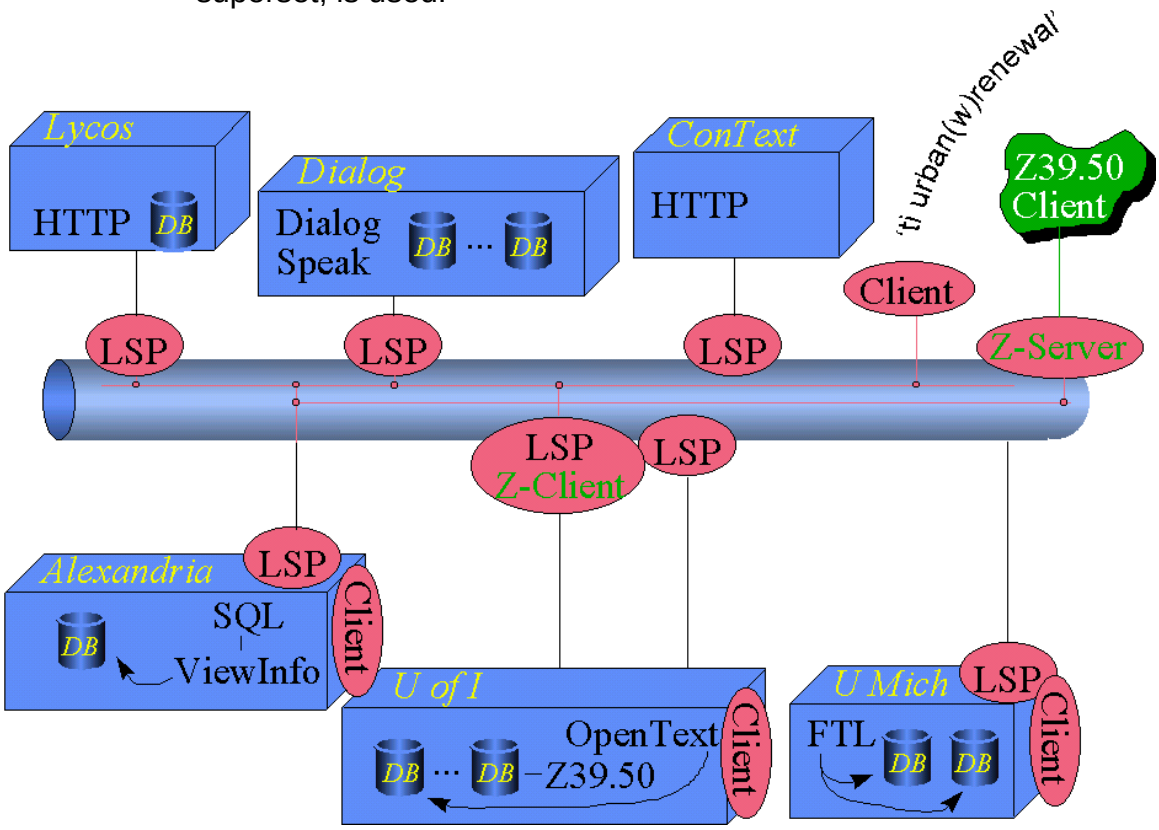

Figure 14. InfoBus

b. **Z39.50 Interoperability Example of InfoBus**: Top part of Figure 14 shows how a Z39.50 client communicates with other services (can be Z39.50 or any other Service). Bottom part of Figure 14 shows how non-Z39.50 clients consume services provided by a Z39.50 Service. The rounded rectangles in the figure are parts interfacing with the InfoBus communicating via DLIOP to provide the interoperable framework.

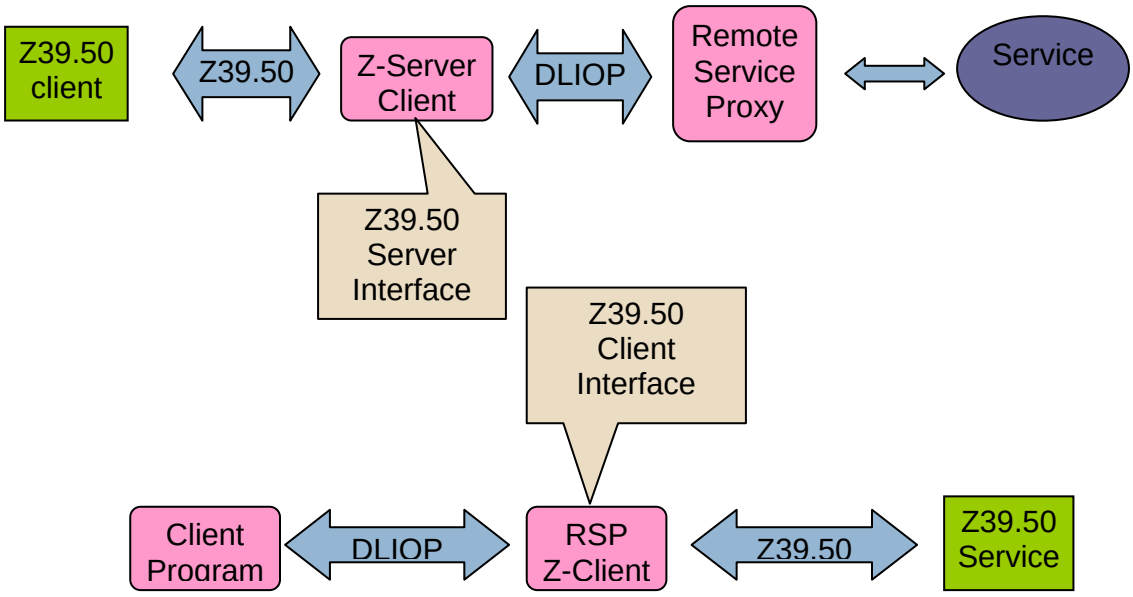

Figure 15. Z39.50 Interoperability Example of InfoBus

8. Parallel and Distributed Information Retrieval
    a. Motivation
        i. Exponential growth in size of online scientific data.
        ii. Managing the size and growth of data demands scalable and multitasking algorithms.
    b. Requirements for Distributed IR Protocols
        i. Enable retrieval of data from digital collections that are **geographically** separated.
        ii. Provide methods for linking and accessing **heterogeneous** collections of scientific data.
    c. Features of Parallel IR
        i. Computation model divides the main task into sub-tasks and executes the sub-tasks in parallel.
        ii. Sub-tasks generally run on homogeneous systems and each system works on the same problem.
        iii. Parallel IR uses shared memory models.
        iv. Parallel IR broadcasts every request to every process.
        v. Main objective is to achieve speed-up.
    d. Features of Distributed IR
        i. Like Parallel IR, computation model in Distributed IR divides the main task into sub-tasks and executes the sub-tasks in parallel.

ii. Distributed IR sub-tasks are run on different processing units where inter-process communication is via network protocols.

iii. Sub-tasks may run on heterogeneous systems.

iv. Distributed IR employs a procedure to select a subset of processes to broadcast a request.

v. Main objectives in Distributed IR are to achieve scalability and availability and speed, but may not be efficient.

e. Case Study on Parallel IR

i. Paper on **Inverted File Partitioning Schemes in Multiple Disk Systems** (IEEE transactions on Parallel and distributed systems, Vol 6, Feb 1995) by Byeong-Soo Jeong and Edward Omiecinski.

ii. Paper discusses two schemes for Parallel IR implementation.

iii. Goal of the paper is to reduce average response time by partitioning the inverted file.

iv. The paper identifies I/O time as a major cost factor in IR system.

v. It exploits the potential of I/O parallelism and balances I/O work-load for better response time by partitioning and distributing files.

vi. The paper discusses two partitioning schemes: based on term-id and based on document-id for inverted file systems.

f. Case Study on Distributed IR

i. Paper on **Methodologies for Distributed Information Retrieval** (18th international conference on Distributed Computing Systems – 1998) by Alister Moffat, Justin Zobel, Owen De Kretser and Tim Shimmin.

ii. This paper discusses three different methodologies for Distributed IR and compares their effectiveness, efficiency and response time.

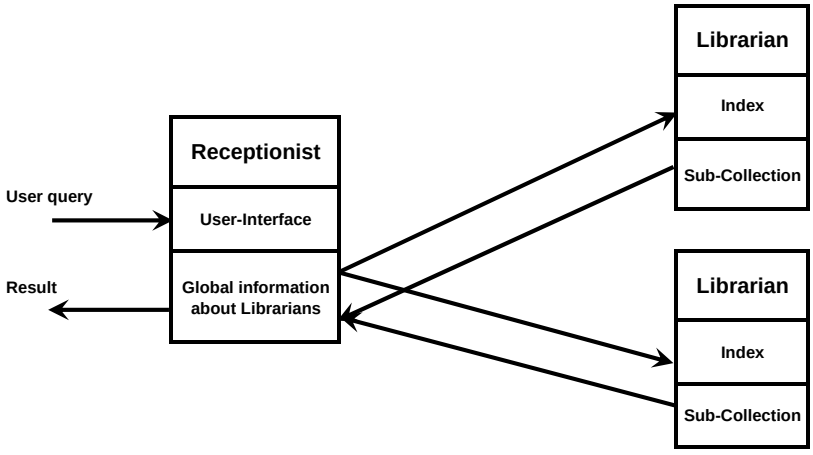

Fig16. Distributed IR Model described in the paper

iii. Three different methodologies are defined based on the global information stored at the receptionist.

- Central Nothing – CN

  The only global information maintained by the receptionist is a list of librarians.

- Central Vocabulary – CV

  Global information stored by the receptionist is the vocabularies of the sub-collections.

- Central Index – CI
  Receptionist has full access to the indexes of sub-collections.

iv. Comparison between three different methodologies for Distributed IR (advantages and disadvantages).

10. Resources

a. Required readings for students

i. Carl Lagoze, Herbert Van de Sompel. The Open Archives Initiative: Building a Low-Barrier Interoperability Framework. JCDL, pp.54-62, First ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'01), 2001. http://www.openarchives.org/documents/jcdl2001-oai.pdf

ii. Byeong-Soo Jeong; Omiecinski, E. Inverted File Partitioning Schemes in Multiple Disk Systems. Parallel and Distributed Systems, IEEE Transactions on Volume 6, Issue 2, Feb. 1995 Page(s):142 - 153 Digital Object Identifier 10.1109/71.342125. http://csdl.computer.org/comp/trans/td/1995/02/l0142abs.htm

iii. De Kretser, O.; Moffat, A.; Shimmin, T.; Zobel, J. Methodologies for Distributed Information Retrieval. Distributed Computing Systems, 1998. Proceedings. 18th International Conference on 26-29 May 1998 Page(s):66 – 73. Digital Object Identifier 10.1109/ICDCS.1998.679488. http://csdl.computer.org/comp/proceedings/icdcs/1998/8292/00/82920066abs.htm

b. Recommended readings for students

i. Metadata Preparation for the Gateway to Educational Materials. http://www.thegateway.org/about/documentation/metadatapreparation/metaprep/

ii. OAI for beginners – the Open Archives forum online tutorial http://www.oaforum.org/tutorial/english/intro.htm

iii. PREMIS Data Dictionary for Preservation Metadata. http://www.loc.gov/standards/premis/v2/premis-2-0.pdf

iv. Suleman, H., and Fox, E. A. Beyond Harvesting: Digital Library Components as OAI Extensions, Technical Report, Department of Computer Science, Virginia Tech, 2002. http://eprints.cs.vt.edu:8000/archive/00000625/

v. Suleman, H. Open Digital Libraries, Dissertation, Department of Computer Science, Virginia Tech, 2002. http://scholar.lib.vt.edu/theses/available/etd-11222002-155624/

vi. Carl Lagoze, Y Payette, Edwin Shin, Chris Wilper, Fedora: An Architecture for Complex Objects and their Relationships, International Journal on Digital Libraries, Vol. V6, No. 2. (April 2006), pp. 124-138.

vii. Yves Petinot, C. Lee Giles, Vivek Bhatnagar, Pradeep B. Teregowda, Hui Han, Isaac Councill, A Service-Oriented Architecture for Digital Libraries (2004), In Proc International Conference on Service Oriented Computing, http://www.personal.psu.edu/staff/i/g/igc2/papers/petinot04service.pdf

viii. Wang, Jun. VIDI: A Lightweight Protocol Between Visualization Tools and Digital Libraries, Master's Thesis, Virginia Tech (May 2002). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.4.6318

ix. B Ahlborn, W Nejdl, W Siberski, OAI-P2P: A Peer-to-Peer Network for Open Archives, 2002 International Conference on Parallel Processing.

x. M Bender, S Michel, C Zimmer, G Weikum, Towards Collaborative Search in Digital Libraries Using Peer-to-Peer Technology, In DELOS Workshop: Digital. Library Architectures, pages 61–72, 2004.

xi. I Podnar, T Luu, M Rajman, F Klemm, K Aberer, A Peer-to-Peer Architecture for Information Retrieval Across Digital Library Collections, LECTURE NOTES IN COMPUTER SCIENCE, 2006. http://globalcomputing.epfl.ch/alvis/research/publications/2006ECDL-Podnar.pdf

xii. Greg Janee, James Frew, The ADEPT digital library architecture, Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries, 2002, Pages: 342 – 350. http://portal.acm.org/citation.cfm?id=544220.544306

xiii. Gerard Salton and Chris Buckley, Parallel Text Search Methods- paper by Salton and Buckley, Technical Report: TR87-828, 1987. http://portal.acm.org/citation.cfm?id=866232&coll=GUIDE&dl=GUIDE&CFID=11874001&CFTOKEN=41172959#

xiv. Robert Kahn, Robert Wilensky, A framework for distributed digital object services, International Journal on Digital Libraries, Volume 6, Issue 2, 2006, Pages: 115 – 123. http://portal.acm.org/citation.cfm?id=1124646.1124650&coll=GUIDE&dl=GUIDE&CFID=11874001&CFTOKEN=41172959

xv. Ray R. Larson, Distributed IR for Digital Libraries, 2003, Pages: 487-498. http://www.springerlink.com/content/eujdc8265v8f55a9/fulltext.pdf

xvi. International Standard Maintenance Agency Z39.50 http://www.loc.gov/z3950/agency/

xvii. Stanford Digital Library Testbed Development http://dbpubs.stanford.edu:8091/~testbed/

xviii. Andreas Paepcke, Summary of Stanford's Digital Library Testbed Design and Status, 1996. http://www.dlib.org/dlib/july96/stanford/07paepcke.html

xix. University of Michigan Digital Library Project http://www.dlib.org/dlib/july96/07atkins.html

c. **Suggested readings for instructors**

i. Van de Sompel, H., and Lagoze, C. The Open Archives Initiative Protocol for Metadata Harvesting. Protocol Version 2.0 of 2002-06-14, Document Version 2004/10/12T15:31:00Z http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm

11. Concept map (created by students)

Note: IHMC Cmap Tools is an open source client tool to create concept maps. CmapServer enables the users to collaborate and share concept maps anywhere on the internet. Both software can be downloaded freely for educational purposes from http://cmap.ihmc.us/download/index.php

12. Exercises / Learning activities

- **Homework assignment:**
    - Form student groups
        - Evaluate the interoperability of DL services using the Stanford and University of Michigan Digital Library Testbeds
    - Peer-to-Peer architecture is widely used to store digital objects. So, considering a peer-to-peer structure which enables a digital object repository, such as ADEPT, what kind of digital library service should be implemented?
- **In-class activities:**
    - In class, break students into groups of 3~4
        - Each group discusses what kind of services their digital library should provided. Does the digital library act as data provider or service provider?
        - Discusses what kind of metadata format and subject schema will provide, and what update frequency.

- Discusses choice of kind of digital library architecture and the components to construct these services.
- Discusses what kind of protocols will be implemented and why?
- The syntax of the underlying database is abstracted in Z39.50, what are the advantages and disadvantages of that?

13. Evaluation of learning achievement

In their answers to the discussion questions, students demonstrate an understanding of

- Different DL protocols
- What role does DL protocols acted in DL architecture?
- What the requirement for IR protocols?

Evaluate the DL harvesting/providing service which student created.

14. Glossary

- Dublin Core (DC): A shorthand term for the Dublin Core Metadata Element Set, the primary work of the Dublin Core Metadata Initiative (DCMI); a set of metadata elements intended to be generic and universally useful for object description.
- Interoperability: The ability of two or more systems or components to exchange information and to use the information that has been exchanged.
- Metadata: Data about data; structured description of information objects; used to aid the understanding, administration, and use of information objects.
- Resource discovery: Identifying previously unidentified (to the user) data objects
- CORBA: The Common Object Requesting Broker Architecture is a standard defined by the Object Management Group (OMG) that enables software components written in multiple computer languages and running on multiple computers to work together.

15. Additional useful links

None

16. Contributors

Ajeet Singh (Virginia Tech, developer), Yinlin Chen (Virginia Tech, developer), Srinivasa Santhanam (Virginia Tech, developer), Weihua Zhu (Virginia Tech, developer), Edward A. Fox (Virginia Tech, evaluator)