



Norwegian University of
Science and Technology

Adding a local node to a global georeferenced digital library

A local administrator's revelations

Kai Torgeir Dragland

Submission date: May 2005

Supervisor: Ingeborg Torvik Sølvberg, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Adding a local node to a global georeferenced digital library

A local administrator's revelations

Kai Torgeir Dragland

Master Thesis
Information Management group
Department of Computer and Information Science
Norwegian University of Science and Technology

Supervisor: Professor Ingeborg Torvik Sølvsberg

February 2005

Preface

I would like to thank my supervisor Ingeborg Sølvsberg for her support and guidance on the path that became my master thesis. I would have been long lost and still in the mist wandering, if she had not held the guiding light to illuminate my path and mind, her advice and council have been like a beacon lighting the way through the fog. She has been a source of inspiration and has supported me beyond my expectations. Thank you very much!

I would also like to thank the developers of the Alexandria Digital Library, in particular Rudolf Nottrott, David Valentine, Greg Janée, and Håvar Valeur. Without their help I would never have been able to add a local collection to the local NTNU ADL node. Thank you for taking the time to answer my questions and making the configuration for adding the Galleri Nor collection. The National Library of Norway should also be thanked for allowing us access to the Galleri Nor collection.

The IT department of IDI, Trond Aalberg, Stein Eldar Johnsen, and Tord Fredriksen have all offered technical help at some point in the installation process of the ADL node on the Fenris Solaris server. Sometimes only a single letter (*gtar* instead of *tar*) solved the problem. Thank you for the help!

My fellow students that shared the same office should also be mentioned, often revelations can be had through good discussions concerning problems and solutions. They may have learned more about digital libraries (and configuration issues and possible solutions) than they bargained for, but their support and the dialogues have been greatly appreciated.

Eskil Solvang and Stein Dragland have read through the thesis and corrected typos and other errors. Eskil has suggested improvements for some formulations and headings.

Last, but not least, I would like to thank my dear Kjellfrid and my little girl Kristine for the support through the time it has taken to write this thesis.

Trondheim, 9th of February 2005

Kai Torgeir Dragland

Summary

Global digital libraries depend on the cooperation between many independent/autonomous organizations. The configuration of the local sub-systems in a global distributed digital library depends both on the requirements from the global system as well as from the local environments. The local administrators in the different organizations play a crucial and often neglected role in spreading global distributed digital libraries.

The main goal of this thesis has been to add an operational local node, with local collections, to a global georeferenced distributed digital library network. The pilot installation was done with Alexandria Digital Library as the global georeferenced distributed digital library network, and Galleri Nor from the National Library of Norway was used as a local collection for the NTNU ADL node.

The conclusions of this thesis have been reached through an investigation of theory concerning distributed georeferenced digital libraries and an investigation of problems and solutions associated with the pilot installation of the NTNU ADL node with Galleri Nor as the local collection.

The main findings and recommendations from this work can be summarized as follows:

1. *Digital libraries are characterized by the use of many different standards, formats, and technologies. This has an impact on the configuration of these systems.*

The complexity of digital libraries must be dealt with so that local administrators can install and configure digital library nodes without needing to be experts in all the different standards, formats, and technologies used in the digital libraries. A configuration tool can be used to add a layer of abstraction between the local administrator and the complexity of a digital library. Lowering the requirements of skills needed in order to install and configure a digital library with local collections can increase the expansion of the digital library technology. An increase in the number of nodes in a distributed digital library will most likely increase the number of available collections, and the increase of available collections will increase the value of the distributed digital library network for information seekers. As more information seekers realize the value of the distributed digital library network more organizations will seek to become part of such a global network.

Addressing the problems the local administrators face in the configuration and installation process in order to make a more user-friendly system is the first step in the right direction on the path towards a global distributed digital library system.

2. *Documentation must be written with a complexity and a terminology that can be easily understood by the user groups of the system. It is important to know for whom the documentation is written.*

The documentation is an important part of the puzzle for solving the problems related to the configuration of a digital library with local collections. In digital libraries, as in all other software systems, the user groups have to be identified. The documentation must meet the needs and use the right terminology at the right knowledge-level for each user group. Three user groups are identified in this thesis: software developers, local administrators, and information seekers.

3. *It is possible to add existing Norwegian collections to a global georeferenced distributed digital library, and to make the local collections available for a global audience.*

One main difference between georeferenced digital libraries and digital libraries in general, is that georeferenced digital libraries provide new access points to collections where spatial literacy can be used to solve problems and satisfy information needs. The problems described in this thesis are not specific for global georeferenced digital libraries, but will affect all global digital libraries with distributed nodes in the same way.

Existing Norwegian collections can be added to a global georeferenced distributed digital library network regardless of the content of the collection, or which metadata formats and standards used. The prerequisite of this thesis was to use the Alexandria Digital Library and add a collection from the National Library of Norway to the local NTNU ADL node. Galleri Nor was chosen as the collection from the National Library.

The theory part of this thesis investigates digital libraries and what the georeferenced information paradigm offers in the context of digital libraries. The theory gives an overview of georeferencing in digital libraries and which technologies and problems associated with this relatively new paradigm. Special attention is given to the Alexandria Digital Library system. The possibilities this system offers are discussed in the *ADL* chapter. The *State of the Art* part of the thesis compares different georeferenced systems and investigates the concept of a fully fledged georeferenced digital library and how different systems compare to this concept.

The *Installation of the NTNU pilot system*, *Discussion*, and *Possible improvements and recommendations* chapters give an overview of the installation and issues related to the installation and configuration of a local node (from a local administrator's point of view), and how these issues may be addressed through possible solutions and further work.

Table of Contents

Preface	i
Summary	iii
Table of Contents	v
1 Problem description:	
Installation of a local node in a distributed digital library system	1
1. 1 Problem statement	1
1. 2 Prerequisite	2
2 Theory and enabling technologies	3
2. 1 Digital Libraries	3
2. 1. 1 Digital Libraries - Overview	3
2. 1. 2 Multidisciplinary	4
2. 1. 3 Digital Libraries, a specific kind of information system?	4
2. 1. 4 Persistent information	7
2. 1. 5 Distributed Digital Libraries	13
2. 1. 6 Applications of Digital Libraries	16
2. 2 Georeferencing	17
2. 2. 1 What is Georeferencing?	17
2. 2. 2 Why should we use georeferencing?	18
2. 2. 3 Types and characteristics of georeferenced objects	19
2. 2. 4 How to represent locations/coverage geospatially	20
2. 2. 5 Declaring some key terms in georeferencing	21
2. 2. 6 Geodetic Datum	24
2. 2. 7 Gazetteers	25
2. 2. 8 Geospatial information retrieval (spatial matching)	26
2. 2. 9 Issues in Georeferencing	29
2. 3 Georeferenced Digital Libraries	37
2. 3. 1 Georeferenced Digital Libraries	37
3 State of the Art	39
3. 1 A fully fledged georeferenced digital library	39
3. 2 Comparison of georeferenced systems	40
3. 2. 1 Investigated system aspects	40
3. 2. 2 Norgesglasset (Norwegian map service)	40
3. 2. 3 Primus (Norwegian digital library for museum collections)	43
3. 2. 4 EDINA: Go-Geo!	44
3. 2. 5 Galleri Nor	47
3. 2. 6 DLESE (Digital Library for Earth System Education)	49
3. 2. 7 ADL (Alexandria Digital Library)	54
3. 2. 8 Brief overview	56

4 ADL	59
4. 1 ADL (Alexandria Digital Library)	59
4. 1. 1 ADL Overview	59
4. 1. 2 Georeferenced information	60
4. 1. 3 Heterogeneous collections	61
4. 1. 4 The Bucket framework	61
4. 1. 5 Technology	65
4. 1. 6 Thesaurus	66
4. 1. 7 ADL Gazetteer	67
4. 1. 8 Webclient	68
4. 1. 9 Middleware services	69
4. 2 The ADL distributed architecture.	70
4. 2. 1 Collections	70
4. 2. 2 Adding a local collection to a local node	72
4. 2. 3 Making a local collection available to the distributed network	73
4. 2. 4 Making use of collections from other ADL nodes	73
4. 2. 5 Glimpse of the future	75
4. 3 Operational ADL nodes.	76
4. 3. 1 Map and Imagery Laboratory (MIL)	76
4. 3. 2 Environmental Information Lab (EIL)	76
4. 3. 3 New Zealand ADL (NZADL)	77
4. 4 ADL - a service or a total system	77
4. 4. 1 ADL as a total system	77
4. 4. 2 The middleware as a service	78
5 Installation of the NTNU pilot system	79
5. 1 Components of the node.	79
5. 1. 1 Webserver	79
5. 1. 2 The ADL middleware	80
5. 1. 3 A local collection	81
5. 2 The NTNU ADL node prototype	85
5. 2. 1 Searching the local collection	85
5. 2. 2 New access points for the Galleri Nor collection	86
5. 2. 3 A brief installation history	87
6 Discussion	89
6. 1 Problems and solutions in the installation phase	89
6. 1. 1 A webserver	89
6. 1. 2 The ADL middleware	89
6. 1. 3 Adding a local collection to the NTNU node	90
6. 1. 4 Making local collections available to the distributed network	93
6. 1. 5 Other problems encountered	94
6. 2 ADL as a research project.	94
6. 2. 1 The georeferencing paradigm	94
6. 2. 2 Cooperation with the ADL developers	95
7 Possible improvements and recommendations	97
7. 1 General recommendations	97
7. 1. 1 Documentation	97
7. 1. 2 Configuration Management	100
7. 1. 3 Network communication through firewalls	102
7. 1. 4 Communication in distributed organizations	102

7. 1. 5 Standardization of configuration file documentation	105
7. 1. 6 High cohesion and low coupling in configuration files	106
7. 1. 7 A graphical user interface (GUI)	106
7. 2 Special recommendations	107
7. 2. 1 Possible solution for configuration of the ADL middleware	107
7. 2. 2 Possible solution for adding local collections to a local node	109
7. 2. 3 Possible solution for the sharing of local collections	111
7. 2. 4 Summary of the different solutions	111
7. 3 Further work	114
7. 3. 1 Making the GUI based configuration tool	115
7. 3. 2 Bulletin board/Discussion board	116
7. 3. 3 A collection discovery service for ADL	117
8 Conclusion	119
9 References	123
Appendix A	129
A. 1 Use Case description for a configuration tool	129
A. 1. 1 Use Case UC1: Configuration of the ADL Middleware	129
A. 1. 2 Use Case UC2: Configuring/adding and Management of Collections	130
Appendix B	132
B. 1 Suggestion for DTD for the collection_opml.xml file	132
Appendix C	133
C. 1 Configuration files, suggestion for description format for ADL	133
Appendix D	141
D. 1 Optimal Database layout for the Galleri Nor metadata	141
Appendix E	143
E. 1 Configuration files for the ADL and the Galleri Nor collections	143
E. 1. 1 collection_opml.xml	143
E. 1. 2 domain.js	144
E. 1. 3 webclient.conf	144
E. 1. 4 access-report-template.xml	146
E. 1. 5 browse-report-template.xml	146
E. 1. 6 bucket-report-template.xml	146
E. 1. 7 bucket99.conf	148
E. 1. 8 database.properties	152
E. 1. 9 drivers.conf	152
E. 1. 10 metadata.xml	153
E. 1. 11 query-translator.py	156
E. 1. 12 web.xml	161
E. 1. 13 frame_blank.html	167
E. 1. 14 CollTest.properties	168

1 Problem description: Installation of a local node in a distributed digital library system

Software systems are becoming increasingly complex, and although modern techniques and methods encourage sound practices, the sheer volume of components, their relationship and the diversity of human interaction are often impossible to control without an approach that is systematic and automated.

Distributed digital libraries are software systems that make use of a variety of different technologies and are therefore becoming increasingly more complex as more functionality and services are added to the total system.

“A distributed digital library should be considered a distributed application or a specific type of distributed information system. It uses the underlying distributed system of networks, machines and other resources to provide an integrated environment where data and functionality are located on multiple computers that communicate through a network but appear as a coherent resource.”[1]

A global digital library consists of components, and these components are not controlled or managed by one organization alone; a global digital library consists of autonomous digital libraries that again can include various autonomous collections.

1.1 Problem statement

This master thesis' main goal is to illuminate problems that can arise when installing a local node of a large distributed digital library system, and to discuss different solutions for making the installation of digital library systems more understandable and less error prone. To make the installation of large distributed digital library systems more user-friendly, more understandable, more efficient, and less error prone may aid in the distribution of digital libraries. Thus, making more resources available to the public and saving time and money needed to deploy large digital library systems.

Since the world of digital libraries is large and a variety of different digital libraries exists we will narrow this solution to georeferenced distributed digital libraries, but the lessons learned here should be true for most large digital library systems whether they are georeferenced, distributed or not.

Georeferenced digital libraries are specialized digital libraries where the documents are georeferenced to one or more locations on the face of the Earth. A document's geographic location can be

used in the document retrieval process. Georeferencing is a key concept in georeferenced digital libraries, instead of just being a feature used for retrieval of documents.

In this thesis we will do a case where we link to a distributed digital library by installing a local node of the distributed digital library and adding local collections to it. In this context it is feasible to explore how collections can be made searchable when the databases and nodes are local. We will explore if it's feasible to develop simple tools for lowering the complexities of adding new collections to a node.

A summary of the problem statement:

- What characterizes a georeferenced digital library compared to other digital libraries?
- When adding a local node of a georeferenced distributed digital library with local collections, how do the local collections become visible and searchable for the whole system?
- What problems arise when configuring georeferenced distributed digital libraries, and how can these problems be solved in an efficient and user-friendly way?
- Is it feasible to develop tools to aid in the configuration of digital library systems? What can be achieved with operational tools, when available?

1. 2 Prerequisite

Prerequisite for my work is to use:

- The Alexandria Digital Library middleware
- A collection from the National Library of Norway that contains georeferenced items

Department of Computer and Information Science (IDI) at the Norwegian University of Science and Technology (NTNU) is cooperating with the Alexandria Digital Library (ADL) developers at the University of California, Santa Barbara (UCSB). The ADL middleware is an architecture for georeferenced digital library systems that is being developed at the UCSB. It can be used to give access to databases with georeferenced digital objects. IDI wants to install an ADL node at NTNU that is connected to the system in Santa Barbara. We want to include Norwegian collections into the ADL node at NTNU, and to make those collections searchable from a georeferenced digital library.

When autonomous collections are linked to an ADL node, and all the ADL nodes are linked together in a global information infrastructure we say that ADL is a global digital library.

The database Galleri Nor at the Norwegian National Library contains more than 70.000 norwegian photos from the time period 1880 to 1950. This collection contains georeferenced digital objects, and is therefore a good first choice for appending to the ADL node at NTNU. The digital photos in Galleri Nor are georeferenced using placenames as the main key for georeferencing.

2 Theory and enabling technologies

*“Rejecting theory as useless in order to work only on everyday things is like proposing to cut the roots of a tree as they do not carry fruit.”
- Condorcet*

2. 1 Digital Libraries

*“What's another word for thesaurus?”
- Steven Wright*

2. 1. 1 Digital Libraries - Overview

To define what digital libraries are, is anything but trivial. The following definition is an extension of Christine Borgman’s definition[2] of digital libraries, giving a more holistic view of digital libraries.

“Digital libraries comprise persistent digital information in every medium (text, images, sounds; static and dynamic images) and make the information resources available in distributed networks. The research field of Digital Libraries is multidisciplinary and extends, enhances, and integrates, the results and experiences from, e.g., information retrieval, librarianship, information systems, database technology, and human-computer interaction. The content of digital libraries include data, metadata that describe various aspects of the persistent information, e.g., its representation, creator, owner, reproduction rights, and metadata consisting of links and relationships to other information objects and other metadata, whether internal or external to the digital library. A digital library offers a set of tools and capabilities to locate, retrieve, and utilize the information resources.” [3]

The above definition of what digital libraries are, takes into account all of the aspects of digital libraries. This definition along with the three key characteristics which according to [4] defines digital libraries and distinguish them from other system, forms a concept of what digital libraries are:

- *Functionality:* Digital libraries offer integrated services to a comprehensive digital collection of cultural or scientific information that is available primarily for reading and secondarily for expanding upon as well as annotating.
- *Purpose:* It is mainly used for learning and research.
- *Lifetime:* It provides access to information whose value is preserved across long periods of time.

Learning and research are the two main purposes for using digital libraries. Digital libraries can be used as components in many applications areas, but if we abstract why digital libraries are used by information seekers, it is for the purposes of research and learning.

The [4] workshop report lists features that are emphasized in digital library information systems, for example: rich information needs, multiple sources of related information, heterogeneous and multimedia information, collaboration, and many more.

2. 1. 2 Multidisciplinary

Digital libraries exist in the intersection between different disciplines and interests, and this of course leads to many views of what a digital library is. Christine Borgman discusses two different meanings of digital libraries in [5]:

- Digital library as an organization
- Digital library as a service

The library practitioners of the field often promote the “*digital library as an organization*” viewpoint, and computer science and some other research communities often promote the “*digital library as a service*” viewpoint.

So what is the substantial difference between these two viewpoints? The “*digital library as an organization*” viewpoint is based on the assumption that the traditional library institutions are the organizations in charge of information management. The the main concern of practitioners in the traditional libraries is to enable these organizations to manage, store, and make available digital content to a networked world. The “*digital library as a service*” viewpoint promoted by computer science and some other research communities focuses on enabling technologies (information retrieval, databases, network architectures, and so on). The digital library for them becomes the umbrella that on behalf of the users bring all the technologies together as a service for collecting and organizing content by use of information technology. The term digital library is an umbrella term and a meeting place for a lot of different user needs, information resources and technologies.

Borgman’s dualistic view of a separation between the understanding of what digital libraries were, held more true in the past. As both the traditional library world and the research communities embrace more of the same concepts and enabling technologies for digital libraries, this dualistic view will eventually be a thing of the past. Different research communities may focus on different aspects of digital libraries, and their focus may affect their understanding of what digital libraries are. Digital libraries are primarily funded on concepts, methods and models from the domains of information retrieval, librarianship, information systems, database technology, and human-computer interaction.

2. 1. 3 Digital Libraries, a specific kind of information system?

Information systems, corporate memory, museums, digital libraries, and archives are increasingly making use of the same technologies and concepts for development, so the boundaries that used to be clear between the different disciplines are slowly fading. All of these disciplines are dealing with information management and organization, preservation, user-groups, metadata, and so on.

The policy and focus may vary between the disciplines, but they also share several characteristics so maybe in the future the convergence will increase even more and become total.

When looking on a definition of an information system (IS) most digital library researchers will be able to recognize the aspects of information systems, and see that they are also aspects in digital libraries.

“A computer hardware and software system designed to accept, store, manipulate, and analyze data and to report results, usually on a regular, ongoing basis. An IS usually consists of a data input subsystem, a data storage and retrieval subsystem, a data analysis and manipulation subsystem, and a reporting subsystem. Widely used in scientific research, business management, medicine and health, resource management, and other fields that require statistical reporting, information systems can be broadly classified as spatial or nonspatial, depending on whether the data refers to a system of spatial coordinates. See also: geographic information system, management information system, and spatial information system.” [6]

The definition above captures the technical essence of information systems, but one key component is missing in [6] definition of information systems.

“Information systems support business information needs through effective use of software and computers... Information system development methods must to some extent be holistic, because all relevant aspects have to be considered, be they of a technical, sociological or political nature.”[7]

Humans and organizations are key components in any information system.

“Information systems must be planned in a total context, taking all relevant actors into account. Only in this way can a sufficiently wide framework be provided to enable the development of information systems with enough desirable features. The three major subsystems that have to be considered are the computer (and communication) system, the (application) software system, and the (human) organization that is being served by the two other subsystems.”[7]

If the definition from [6] is extended to a holistic view so that it includes users and organizations as part of an information system, the definition will come close to the digital library domain.

“A DL is a special kind of an information system, and consists of several components such as a collection, a computer system (a technical system), persons, and the environment (or usage), for which the system is built. For DLs it is important to integrate systems, information/collection and humans, as well as support the viewpoints of the different shareholders in DL research.”[8]

Digital libraries, informations systems, digital archives, digital museum systems, and corporate memories all convergence on some characteristics. The core components in all these systems are: users, data/collections, and technology.

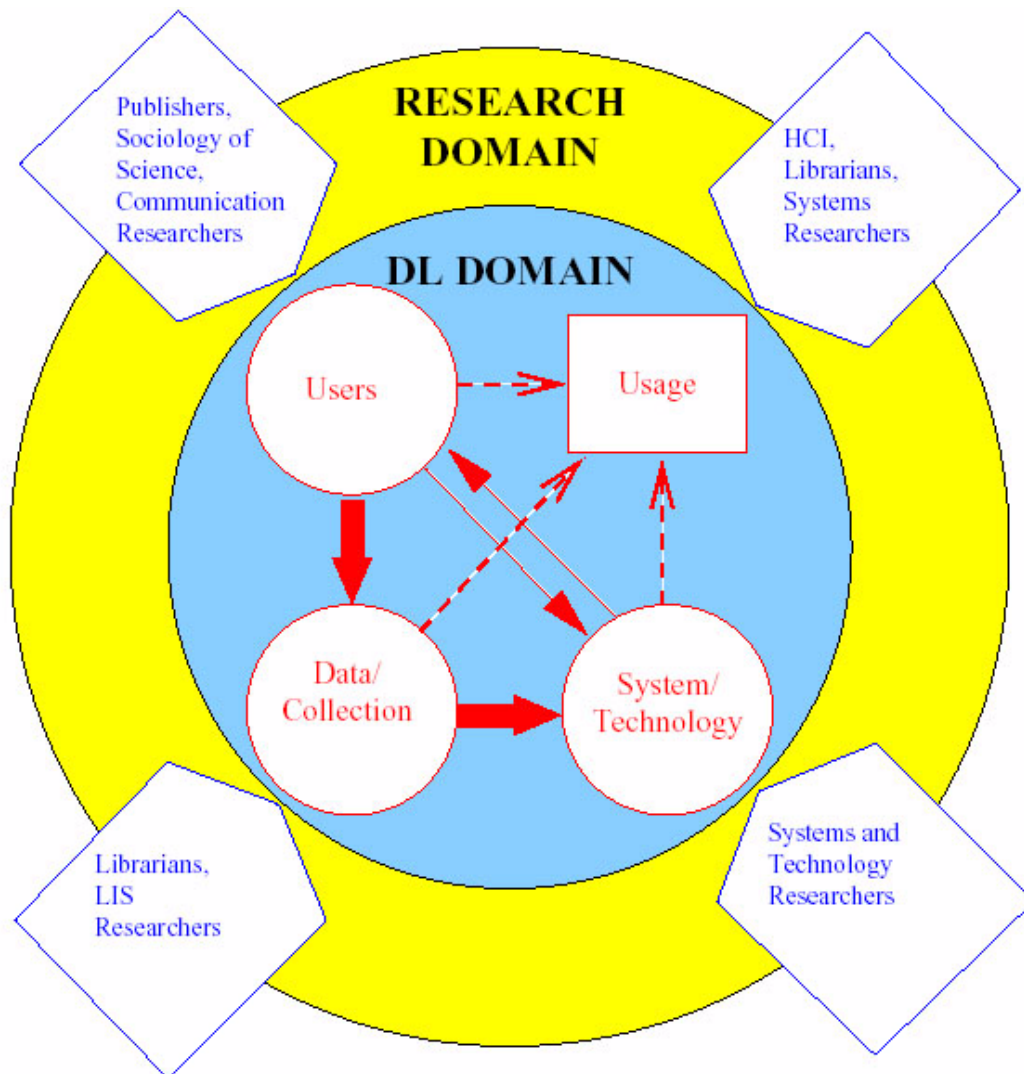


Figure 2-1. A generalized schema for a Digital Library [8]

“Definition of the set of users predefines the appropriate range and content of the collection (block arrow connecting “users” to “collection”). The nature of the collection predefines the range of appropriate technologies that can be used (block arrow from “collection” to “technology”). Finally, the attractiveness of the collection to the users and, secondarily, the ease of use of the technologies by the user group, determine the extent of the usage of the digital library (thin arrows show the human-computer interactions, while the dotted arrows show the collective contribution of user, collection and technology interactions to observed overall usage).”[8]

The quote above is about the generalized schema for digital libraries illustrated in figure 2-1, this observation holds true for the other systems as well. The inner circle called the “DL Domain” in the figure above contains the three components (Users, Data/Collections, and System/Technology) that are shared characteristics for the different systems, and these three components are reflected in the usage of the system. Instead of focusing mainly on differences between the differ-

ent types of systems, it can be smart to look at the similarities, peaking over the fence to see what goes on in other disciplines that are similar, and maybe learn from each other.

The main purpose for digital libraries Adam and Yesha explain with the following:

“Digital Libraries are concerned with the creation and management of information sources, the movement of information across global networks and the effective use of this information by a wide range of users...”[9]

Adam’s and Yesha’s definition is broad, information systems, digital archives and museums will also be covered by that definition.

2. 1. 4 Persistent information

The discussion about persistent information leads us briefly into the research fields of:

- Identity
- Information object
- Digital storage formats
- Metadata and catalogues
- Collections

Lifetime: It provides access to information whose value is preserved across long periods of time.[4]

One of the characteristics of digital libraries is the fact that they provide access to information across long time periods, unlike the world wide web where change happens rapidly. Digital libraries can also store dynamic information that may change rapidly. Train tables are an example of dynamic information that will change quite rapidly (often several times a year) compared to the holdings of the Bible. But there may be a policy for the digital library to store versions and instances of old train tables when they are updated, maybe someone wants to perform research on changes of train tables in to future so this can become valuable information.

Corporate memories and other information management systems used in the business world sometimes also store information across long time periods. All information may be stored for 10 years in case of lawsuits.

Are digital libraries limited to only digital information?

“...digital libraries provide for collection development, organization, access, annotation and preservation, and deal both with information in digital form as well as digital management of information residing on physical media.”[10]

Stephen Griffin’s definition broadens the understanding of what digital libraries are. Digital libraries can also contain digital artifacts that cannot be represented or distributed in printed formats, as well as containing document surrogates for physical media.

Identity

Identity is one of the key features in digital libraries, what use would persistent information have if retrieving or referring to the information would be near to impossible? Persistent identifiers is the “glue that keeps it all together” in digital libraries. By using persistent identifiers it is possible to link metadata descriptions to information objects in the digital library.

Metadata can be used to identify information based on a set of distinguishing characteristics, another method of identification is through the use of tokens that are associated with entities to serve as identifiers. Various identifier schemes exist in the digital library domain, some schemes use dumb identifiers while other schemes have identifiers that carry information about the resource they identify (like the Serial Item and Contribution Identifier (SICI) scheme for the identification of specific contributions within serial publications[11]).

URLs (Uniform Resource Locator) are the most used identifier on the Internet today. A URL points to the location where a document is located and how it can be retrieved. URL and other location-based identifiers use location to identify a document. Location-based identification is not a reliable mechanism since the web address for a specific document may change, or the content of the web address may change.

The persistent identifiers should be independent of location and time, they should be unique across time and space (Universal Unique Identifier(UUID)[12] is an example of such a identifier scheme), even if an information object is moved an identifier that points to it should still be valid afterwards. Identifiers should be tied to the information object and not to characteristics of the information object or its physical location. If the identifiers are liberated from the method of retrieval and the location where the information object is stored, a translation service will be needed to translate persistent identifiers into return values; the physical location the information object and how to retrieve the information object. A system that is developed for this purpose is the Digital Object Identifier(DOI)[13].

Identification, naming and addressing are issues that are fundamental in all information systems, both local and distributed systems. The Uniform Resource Identifier specification[14] illuminates these terms in the light of a distributed information system context. The specification defines identifiers as strings that identify abstract or physical resources. Locators are identifiers that identify resources via a representation of their primary access mechanism, for example a network location. Names are characterized by their purpose of providing logical and persistent identifiers independent of physical location. The distinction between the terms can vary somewhat among different technologies and communities, another view of names and identifiers is listed below.

“Both terms [names and identifiers] reflect the general concept of symbols that denotes conceptual and physical entities. The term *name* often implies a symbol that it is interpretable by humans (and machines). The term *identifier* implies that the symbol uniquely identifies something”. [1]

Identifiers are used within digital libraries to identify metadata and collections, these identifiers are often hidden from end users and other systems. The internal identifiers are important to provide exchange and reuse of metadata records and collections within digital libraries.

In the georeferencing paradigm being able to identify locations on the face of the earth is the key feature. Placenames can change over time, but the location will remain the same. More about georeferencing in chapter 2.2.

Information objects

Traditional libraries are associated with books, journals and other information sources, so it is natural to assume that digital libraries are associated with the digital realms information sources: the digital document-like objects. Even though the content of digital libraries can be highly heterogeneous, the digital document-like objects can be organized into collections and described by metadata records.

Digital document-like objects can be “born digital” (only exist in digital formats and version) or it can be a digital copy of for example a traditional book. It is possible to make digital copies of the content of a traditional library, but not all digital document-like objects can be printed out on paper.

It is possible to make an abstraction of the information in digital libraries:

“In its simplest form, the model states that everything in a digital library should be considered an *information object* and that the main mechanism for expanding upon these objects and communicating about them is through the use of names or identifiers.”[1]

Digital libraries contain entities with a certain identity of their own, this is emphasized in the *object* part of the information object expression. The fact that these objects can be used as sources of information or knowledge is reflected in the *information* part of the information object expression. Identifiers are used to expand and enrich the information objects by making it possible to communicate to and refer to the information objects. Metadata describe information objects, more about metadata later in the chapter.

The *information object* can be compared with the established convention of the *document*. Documents are artifacts that carry knowledge or information. Documents are used to store and exchange information. The documents have been used to pass on information from generation to generation, making it possible to capture and store facts and ideas. It would be very hard if not impossible to do research and advanced reasoning without any form of documents. In the past documents were limited to written word, now new media are available to capture and store information. Images, sound recordings, moving images, and museum artifacts all contain information and therefore the document concept can be expanded to include all artifacts that later can be used as a source of information and knowledge.

Digital storage formats

New digital formats become available as new technologies and research are continually moving forward. Some of the new digital formats can offer new possibilities of usage for information

objects stored in “old” formats in a digital library. Converting from one format to another in digital libraries can be motivated by either a preservation intent or by a request for new functionality by converting to a new format with more possibilities. The hardware/software used for storage will become obsolete (sooner or later), forcing a move to a newer format to be able to use the resource without the outdated technologies. This is also an important area of competence within digital libraries to ensure the preservation of information over time, giving it a long life span.

“Digital storage of a material makes it possible to copy the material from one medium to another, and makes it possible to convert from one format to another without loss of quality. This means that the material is made independent of the medium that stores it, and can therefore exist long after the original medium is broken or outdated.”[15]

By converting from old digital formats and mediums to new ones the digital libraries ensures the preservation of digital information over time, making sure it will also remain available in the future.

Metadata and cataloging

In digital libraries, metadata is the preferred term when referring to information that describes other information.

“Metadata is data associated with objects which relieves their potential users of having to have full advance knowledge of their existence or characteristics.”[16]

This definition made by Dempsey and Heery exemplifies a general and inclusive view of metadata. The definition says metadata can be used as a surrogate for the actual content of an information object, so metadata can be viewed as document surrogates. The metadata term covers all formats of descriptions that can be found in any domain, it is not a digital library specific term, but domain independent.

Metadata records can be used to make structured descriptions of information objects. These structured descriptions are essential for finding relevant information in huge collections, and they also make it possible to evaluate different information objects without accessing the actual content of the information objects. There is no *de facto* standard for metadata, a metadata standard that is adequate in one setting or for one digital library may not be as useful for other collections or digital libraries. Since a variety of different metadata standards and formats exists, it is important that distributed digital libraries support heterogeneous metadata. Examples of metadata standards are the MARC (MACHINE-Readable Cataloguing) [17] formats and the Dublin Core[18] metadata element set. The Dublin Core metadata element set is a standard for cross-domain information resource description. The MARC formats are standards for the representation and communication of bibliographic and related information in machine-readable form.

According to Anne J. Gilland-Swetland[19] there are five subtypes of metadata:

- Administrative (management and administration, digital rights, location)
- Descriptive (identification or description of the information object)
- Preservation (preservation management of the information object)
- Technical (how a system functions, or metadata behaves)

- Use (level and type of use of an information object, use and user tracking)

The information included in a metadata record may serve purposes like[1]:

- Searching, discovering, or selecting information
- Accessing or retrieving information
- Identification or verification.
- Contextual information about the information
- Management and preservation

Metadata enables explicit knowledge about content objects to be stored and maintained. Metadata about location, organization, management, usage, and authentication is not an implicit part of an information object, but it is still part of the metadata required for information objects in digital libraries.

Descriptive cataloguing is by many considered to be the true art of the library practice, most of the services libraries can offer are based on the bibliographic records of the library catalogue. The catalogue offers users access points to the information contained in the collection of a library. In most modern libraries the card catalogue used in the past has been replaced by databases and machine-readable bibliographic records. Databases provide more flexible use of the bibliographic records.

Collections

A collection is an intellectual resource, it represents a selection based on an evaluation of which items are most valuable and have the required potential for reuse. Collections represent one of the basic structuring paradigms used in libraries, archives, and museums. Digital libraries carry on the collection legacy from the library world.

Collections are used within libraries to physically organize and maintain the holdings of libraries. The total holdings of a library can also sometimes be referred to as a collection. A collection can be based on requirements for certain materials to be maintained and processed, or the collection itself can represent some logical organization for management and use in a convenient way. Some collection can also be based on the completeness of the topic it tries to cover. Partial or complete completeness can characterize a collection.

Quality assurance of information in both digital libraries and traditional libraries are upheld by policies for acquisition of new information, this is part of the collection building. Another aspect of collections is relationships and linking, this is one of the key research topics of collection building. Collections can be collections of content objects, but can also be collections of metadata records.

One difference between collections in digital libraries compared to a traditional library collection is that digital library collections reflect virtual organization instead of physical organization, and the collections in digital libraries are not necessarily based on ownership of information. Digital libraries can make use of various collections from different collection providers to make a virtual collection. A digital library does not need to own collections, just make it possible to search and retrieve information from the collections, the collections can be owned and maintained by collec-

tion providers that give access to their collections for a fee or for free. Collections of this type are often referred to as autonomous collections.

Autonomous means that something is independent and has the power to make its own decisions. The collection provider that maintains one or more collections that is appended to a distributed digital library has the power to make its own decisions for its collections and the decisions are made independently of the organization of the distributed digital library.

When an autonomous collection is added to a distributed digital library, the distributed digital library only makes it possible to search and retrieve information from that collection. All other tasks for that collection (adding, updating, deleting, or preserving information) are managed by the collection provider.

A distributed digital library can have collections from various collection providers made available for searching and retrieval through the digital library. Collection provider supplies autonomous collections, where they are responsible for the collection, and all the digital library does is to support searching and retrieval from the collections in a uniform way.

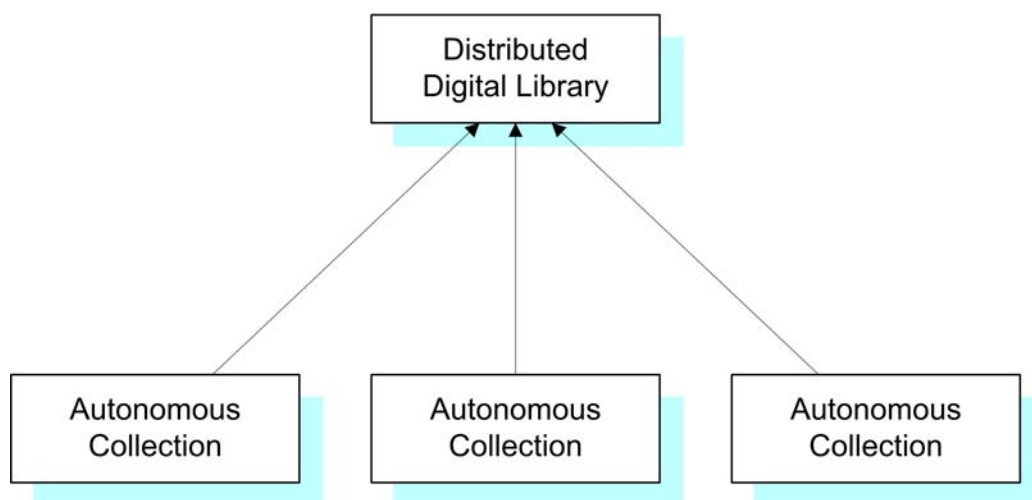


Figure 2-2. A distributed digital library has one or more collections appended to it

Why add autonomous collections to a distributed digital library instead of locally managed collections?

There are two main reasons for adding autonomous collections to distributed digital libraries:

- Scalability
- Virtual collections

Scalability

If the responsibility for maintaining and developing collections is given to the collection providers, then the responsibility is distributed and managers at digital libraries can pick and choose from the best collections available in the given research areas. If the responsibility for the collec-

tions is given to one organization, the organization will need to expand and become bigger as the number of collections maintained in the organization increase. There are limits to how large an organization can become and still remain effective. It is much easier to scale upwards if collections are not managed by one organization, but by multiple collection providers.

If a distributed digital library seeks to expand, it is crucial that the responsibility for collections is distributed and the collection providers have to do as little as possible to append a collection to the library. The distributed digital library seeks to append autonomous collections.

Virtual collections

It is possible to pick and choose collections from various collection providers, and then “merge” a selection of collections into a new virtual collection with a better completeness of the topic the virtual collection covers.

The downside of having multiple collection providers is that they most likely will not use the same metadata standard to describe the items contained in the collections. Some collection provider may even use local methods that don’t adhere to any standard to describe metadata. So the distributed digital library must have ways to account for diversity in metadata formats and still provide uniform searching and retrieval to the users of the distributed digital library.

The architecture of the distributed digital library must be designed so it can deal with different metadata standards. It is important that it is as easy as possible to use source metadata from the collections when they are added to the digital library, allowing use of source metadata means less work when adding collections to the digital library.

An example of an architecture that supports use of source metadata when adding new collections to a distributed digital library is the ADL architecture. The ADL bucket framework maps source metadata to buckets via tuples. Doing it this way the source metadata is preserved, and searching can be done at the high level of the buckets or at the level of the source metadata. The ADL architecture is therefore a good solution for adding autonomous collections to a distributed digital library.

2. 1. 5 Distributed Digital Libraries

Definition of a distributed system: “You know you have one when the crash of a computer you’ve never heard of stops you from getting any work done.”
- Leslie Lamport

The following aspects of distributed digital libraries will be discussed in this section:

- Compound objects
- Components and services in distributed digital libraries
- Middleware

To be able to talk about distributed digital libraries we first need to know what a distributed system is:

“We define a distributed system as one in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages.”[20]

So distributed systems are messaging between a collection of autonomous computers linked by a network with software designed to produce an integrated environment. Sharing of resources and cooperation will in general be enabled by a distributed system. Openness can be achieved in distributed systems by using and adhering to well defined protocols and standards.

Now that the concept of a distributed system is established, it is time to establish the distributed digital library concept:

“A distributed digital library should be considered a distributed application or a specific type of distributed information system. It uses the underlying distributed system of networks, machines and other resources to provide an integrated environment where data and functionality are located on multiple computers that communicate through a network but appear as a coherent resource.”[1]

Most large digital libraries today are distributed digital libraries, collections and different services in the digital libraries can reside on different autonomous computers that are linked by a network. By being able to replicate services on different machines it is possible to have transparent error handling, so if one machine goes down the services it offered can be replicated by another machine so the end user will never know anything was in error.

The information objects found in distributed digital libraries can often be *compound objects*.

Compound Objects

Content and metadata in digital libraries often consists of a set of subparts, this compound structure can be of a logical and/or physical nature. According to Trond Aalberg[1] this compound structure is slightly comparable to the concept of collections, but whereas collections aggregate a set of objects that plays the same role, the subparts of a compound object may be of a different kind and/or serve different purposes.

- The *logical structure* is the structure as it appears to the user of the information. A good example of a logical structure is a book. Books can be divided into chapters, subsections, tables/illustrations and an index. Other information objects like for example reports, movies, music files all have logical structures. The concept behind the logical structure is that the structure is an intentional one and that the structure remains intact, independent of the physical representations of the information object. The logical structure of a book is the same whether it is viewed on screen or is printed out.
- The *physical structure* is a structure that is based on the physical organization of the content. Digital information is often managed and stored in several different files. What appears in the logical structure as a whole book can have a physical structure where the chapters are stored in separate files. When the book is retrieved the files are integrated into a whole that is presented as the logical structure to the end user.

The physical structure describes distributed characteristics found in digital libraries. The virtual collection is another example of how a collection provided as a logical structure to the end user can be a compound object where the physical structure includes various collections from different collection providers that are merged into a virtual collection that is presented to the end user.

Components and services in distributed digital libraries

The terms *component* and *service* are often used interchangeably by the digital library architecture researchers. A component is usually a software artifact whereas a service is a resource that a client can make use of in a dynamic way, so they actually denote different concepts.

What is characteristic of *components* is that they are reusable and replaceable, this is a paradigm that has its background in the software and hardware industry. In this paradigm it is possible to choose from different components to build a composite system, the system can be upgraded one component at the time so upgrades can be done in small steps. Object oriented programming where one object can be replaced with another object that does the same job in a different way is an example of software that use components.

It is harder to grasp what a *service* is, the term is used in the client-server model for distributed systems. A process or machine that manages and makes available shared resources is called a server in the client-server model, so in this paradigm a service and a server process can be viewed as being equivalent.

“...most uses of the term “service” imply a decomposition of a larger application into smaller units for the purpose of reuse and dynamic integration.”[1]

So services are network accessible resources or functionality, while software components are artifacts that can be reused and replaces of other software components.

An example of the difference between service and components:

- *Service*: A common service in digital libraries are “Authorization and access management”, this is functionality that is found in many digital libraries.
- *Component*: In an object oriented programming language there can be lots of components used to implement the “Authorization and access management” service: one component to manage database connectivity, one component for the graphical user interface (GUI) presented to the users, and one component the verify users (takes input from the GUI, does a lookup in the database, and sends the results back to the GUI).

In the example above it becomes clear that the components are the software artifacts used to build functionality, the complete functionality of “Authorization and access management” is called a service, this service is a composite made by components. So in essence the service paradigm is taking the object oriented way of thinking one step further. Reuse of services is a step forward in the evolution of distributed systems. Openness and use of standards is important in a service oriented architecture in order to support the modularity in a plug-and-play fashion.

Middleware

Middleware is connectivity software that consists of enabling services that allow multiple processes running on one or more machines to interact across a network[21]. Middleware is essential to migrating mainframe applications to client/server applications and to providing for communication across heterogeneous platforms. This technology has evolved to provide for *interoperability* to support client-server architectures. The middleware term refers to a broad range of software and protocols that are used to provide platform independent development and deployment of software in distributed systems. Middleware is like an extended, more functional API (application programming interface) that masks the complexities of networks and distributes systems.

In digital libraries the middleware is used as part of the architecture for deployment and dynamic integration of services, it is used to make implementation efforts easier.

Middleware has manifested itself in different forms, that reflect different ways of thinking:

- *Message-Oriented Middleware (MOM)*; which provides program-to program data exchange, enabling the creation of distributed applications. MOM is analogous to e-mail in the sense it is asynchronous and requires the recipients of messages to interpret their meaning and to take appropriate action[21].
- *Remote Procedure Calls (RPCs)*; which enable the logic of an application to be distributed across the network. Program logic on remote systems can be executed as simply as calling a local routine[21].
- *Remote data access*; which provides protocols and APIs (Application Program Interface) for communicating with database servers by sending data manipulation language statements and receiving the results[1].
- *Distributed objects*; which are based on the object-oriented paradigm and implements distributed software as objects that are accessible over the network[1].
- *Distributed transaction processing*; which implements client/server interaction with transactional execution semantics[1].

One of the characteristics of middleware according to [1] is that:

“Any distributed solution, however abstract and implementation independent it initially is intended to be, is bound to be influenced by the underlying middleware it is built upon.”[1]

This of course leads to the problem that different middleware solutions will often not be interoperable with each other. If the services of distributed digital libraries are implemented using different solutions for middleware, the heterogeneous usage of middleware that is incompatible with each other will lead to interoperability problems. The middleware solutions need to be interoperable with each other if services shall be reused from one solution to the next.

2. 1. 6 Applications of Digital Libraries

Digital libraries are incorporated into the world today as a natural part of many disciplines. Wherever storage and management of information is needed over time, ease of use, efficient search and retrieval is needed, digital libraries can do the job.

Examples of disciplines that make use of digital libraries are:

- Government: Legal documents need to be preserved for the future. Management, search and retrieval are essential when handling legal documents.
- Health care: In hospitals and in health care digital libraries are used for patient records and management of other types of information.
- Education: Two of the main purposes of digital libraries are research and learning. Digital libraries are used to support education in schools and universities. Digital libraries support and expand the traditional library services that have been made use of through the centuries.
- Biology: Records of specimen are stored in digital libraries.

In organizations and the corporate world, corporate memories serve the same purpose as digital libraries. The abundance of information available in our time makes it important to be able to manage, store, search, and retrieve the information efficiently to be able to make use of it. The importance of information utilization is a key factor for making use of digital libraries, and as more disciplines recognize the importance of information, they make use of digital libraries to serve their purpose.

2. 2 Georeferencing

In the early days of the popularization of the Internet, it was fashionable to declare the death of location

“The global reach of electronic communication would make place irrelevant“[22]

After six years it seems the tide have turned:

“It was naive to imagine that the global reach of the Internet would make geography irrelevant. Wireline and wireless technologies have bound the virtual and physical world closer than ever.“[23]

The following subsections are based on “*Introduction to Georeferencing in Digital Libraries, Tutorial Document*”[24] by Linda Hill

2. 2. 1 What is Georeferencing?

Linda Hill defines georeferencing as:

“Georeferencing is relating information (e.g., documents, datasets, maps, images, biographical information) to geographic locations through placenames (i.e., toponyms) and place codes(e.g., postal codes) or through geospatial referencing (e.g., longitude and latitude coordinates).“[24]

Placenames, postal codes and geospatial referencing is just different types of identifiers that identify what location on the surface of the Earth the information is related to. There are different pros

and cons of using the different types of identifiers, more on this later. If information is related to a footprint, then that information is georeferenced. But what is a footprint?

“A *footprint* is an approximation, expressed in latitude/longitude coordinates, of the subset of the Earth’s surface occupied by the place. Note that a footprint need not be contiguous. For example, a footprint for the state of Hawaii might consist of a union of disjoint polygons, one per island. A gazetteer entry can have more than one footprint, in which case the footprints must represent different approximations or resolutions of the same conceptual footprint.”[25]

What then is a georeferenced resource?

“A *georeferenced resource* is a granule of information that is relevant to an identifiable subset of the Earth’s surface; we call the referenced subset the resource’s spatial coverage.”[26]

If a document is about a location on the Earth’s surface it is linked to that location and can therefore be said to be a georeferenced resource.

2. 2. 2 Why should we use georeferencing?

Michael Wegener once wrote that:

“Everything that happens, happens somewhere in space and time”

Since the location of an event establishes its context and allows it to be linked to other events that have occurred at that place, it is important that an information source such as a digital library can answer questions of the type “what information do you have about *there (then)?*”.

What Michael Wegener is talking about is the important concept of spatial literacy:

“the ability to interpret problems and their solutions in spatial terms”[27]

Making use of georeferenced information to solve problems in spatial terms can be a powerful aid in several different disciplines. The department of health can use georeferencing for mapping the incidents of a disease outbreak to help locate the source of contamination, limiting the spread of the disease if they find out how it is spreading, and what is causing it. The local police department can make use of georeferencing in mapping location of crime-scenes, so it will be easier to predict in what general area crime is concentrated and where the resources will best be put to use to prevent it. In biology georeferencing can be use to map the population of species, and where their habitats are located. Each time a specimen of a species is collected or a species is observed the location of the event is stored. If the information about the collection of the species or observation of it is georeferenced, it can then be used later when making new wildlife reserves, making protection of wildlife more effective, and based on facts of the species habitats. These are just three examples where georeferenced information can aid in research and problem solving by interpreting the problems in spatial terms.

Designing digital libraries so that users can find all resources and information about a location by identifying the location on a map or image of the area, without having to supply placenames or knowing the coordinates for the chosen area is a new access point to the information in digital libraries. Using interactive interfaces with maps or georeferenced images to locate and find information is truly a powerful concept. If an information seeker wanted to find information about “D-day” of world war 2, then he or she would locate the location of Normandy on an interactive map, and maybe enter the phrase “D-day” as a search term, and then execute the query.

A location on the face of the Earth may have several placenames, they may change over time. St.Petersburg used to be Stalingrad when Stalin was the leader of Soviet. The capital of Norway was named Oslo on the 1st of January in 1925, the 31th of December 1924 its name was Kristiania though. Oslo was the old name of the capital, so it was decided to replace the name Kristiania with the old name of the capital. Placenames for a location may also vary from language to language, Germany is for example called Tyskland in the norwegian language. This shows that multiple placenames can be used to describe the same location.

A placename can also refer to multiple different locations on the surface of the Earth, a search using the term “Berg” would retrieve lots of different locations in Norway.

2. 2. 3 Types and characteristics of georeferenced objects

- Maps

Maps are intellectual objects in that they represent/visualize or can be seen as a report of underlying data. They don't need to be in a printed map format.

“They are cartographic products and require the application of the design principles of cartography in terms of symbology, colors, placement of labels, style of lines, and legends.”[24]

- Georeferenced images can be ground based or aerial photographs. Most often satellite images of the Earth is what many think of when hearing about georeferenced images.

The new D2X camera from Nikon[28] can take photographs that automatically add GPS coordinates to the image data when the image is captured. The GPS Cord[29] is an optional accessory for the D2X camera.

“GPS data (NMEA 0183) format can be inserted into D2X image files during capture. Connects most GPS RS232 serial output cables to the 10-pin port of the D2X. Latitude, Longitude, altitude and UTC time data can be stored. “[29]

If a GPS module becomes a common part of the new camera models entering the market then automatically georeferenced images will become easily available and the growth of this kind of georeferenced objects will explode.

Longitude, latitude, altitude and UTC[30] data can currently be stored in the image files. If the angle of the camera and what compass point the camera is directed in (North-South-East-West)

had also been stored at time of capture, it would offer a more complete georeferenced description. The possibility of automatically adding GPS data to image files in a normal camera is a huge step in the right direction for generation of georeferenced images.

- Text in documents often contain placenames and/or associated references (near, west of, n meters from). These textual references can be in both informal and controlled ways.
- Geospatial and georeferenced datasets; Georeferenced datasets contain attribute data, and/or statistics about a location (for example economy, social, biological and so on). It uses place-names, geocoding or coordinate/grid references to refer to the location. Raster or vector data, or layers of each type of data are used to make geospatial datasets.

“A raster based system displays, locates, and stores graphical data by using a matrix or grid of cells...Each cell or pixel has discrete attribute data assigned to it...Generally, raster data requires less processing than vector data, but it consumes more computer storage space... [Remotely-sensed images] are examples of raster data.”[31]

“A vector based system displays graphical data as points, lines and curves, or as areas with attributes... Lines or arcs are a series of ordered points. Areas or polygons are also stored as ordered lists of points... by making the beginning and end points the same node the shape is closed and defined.... It requires complex data structures [but] requires less computer storage space and maintaining topological relationships is easier in this system. Digital line graphs (DLG) and TIGER files are examples of vector data.”[31]

- Museum artifacts are often referenced to where they were found or created using narrative descriptions of the location. These locations can often be site stops for expeditions or relative positions (200 meters from the sea and in the middle of the valley n)
- Music and art can be georeferenced by theme. A painting of the pyramids on the Giza plateau in Egypt can be georeferenced to the Giza plateau.

2. 2. 4 How to represent locations/coverage geospatially

The most common way to represent coordinates are through the use of longitude and latitude, although there exists many other coordinate systems. The two ways of representing longitude and latitude coordinates are by decimal degrees or using degrees, minutes and seconds. The coordinates of Trondheim location is:

Longitude: 10.41667(63 ° 25'0.012"W)

Latitude : 63.41667(10 ° 25'0.012"N)

The common denotation for using decimal degrees is “dd.dd”, for degrees, minutes and seconds it is “ddmmss.ss”.

The reason why “longitude, latitude” is used instead of the common expression “latitude, longitude” is because of some observations Linda Hill have made:

“...the spherical coordinate systems(i.e., longitude, latitude) when projected to flat surface are based on basic x, y graph representations... Since this has turned out to be an issue in the

interface between GIS and textual location references... we intend to reverse the common expression and refer instead to longitude, latitude coordinates.“[24]

Another way to represent locations is by using grids. Universal Transverse Mercator (UTM) is the best known grid system worldwide. The Earth is divided up into regular cells, this can also be done for a country (country-level grid system) on a smaller basis. Some grid systems are based on projections, mathematical transformations of an ellipsoidal Earth onto a two dimensional surface, while some systems are based on the Earth as a globe (unprojected). The shape of the cells is usually square, but some systems like Octahedron use triangular cells.

The UTM location for Trondheim is:

UTM Northing: 7032797.0 Easting: 570728.94 Zone: 32V

“Eastings are measured in meters from a central meridian in each 6 ° zone; northings are measured from the Equator (the zone letter is actually not needed).“[24]

NGO1948 is a local grid based system for Norway, it divides Norway mainland into 8 zones. It became available in 1948. Economic map series in Norway (Scale 1:5000 and 1:10000) uses this standard, most counties in Norway also use this standard for technical maps and other purposes.

C-squares is a new system that looks very promising, it represents geographic coverage using a text string to represent the grids, rather than using bounding boxes or polygons. This makes indexing easy, and it's possible to query just using a “*find*” function in a word processor or with any system that can execute text based searches. C-squares is globally applicable, rather than just local and applies on land and sea equally, it can also be used at any scale.

2. 2. 5 Declaring some key terms in georeferencing

The key terms in georeferencing we want to declared are:

- Scale
- Resolution
- Projection
- Accuracy
- Precision
- Fuzziness
- Bounding boxes

Often the term *scale* is used when we talk about maps and georeferencing. If we have a map with scale of 1:5000 it means that 1 cm on the map equals 5000 cm on the surface of the Earth. This ratio between linear distance on the map and the corresponding linear distance on the Earth's surface is reported as scale. There is often a misunderstanding of what the relative terms *large scale* and *small scale* means. Linda Hill gives us a good rule of thumb to remember the correct meaning:

“The best way to think of them is that *large-scale maps* show a **large amount of detail** in a small area, while *small-scale* show a **small amount of detail** in a large area.“[24]

When zooming in on a map there is a limit to how far one can zoom in as the pixel size of the map will eventually be reached. The pixel size represents the fineness of detail in an image, this pixel size is called *resolution*. The resolution of a map or image may be limited by processing steps to copy or represent data, or by the method used to collect the information. Scale and resolution are closely associated terms; resolution is the degree of details in the underlying datasets while scale is the fixed distance on ground scales.

Projection is another term we have already used that is important to understand in georeferencing. A projection is the:

“transfer of the features of the surface of the earth or another spherical body onto a flat sheet of paper [or other media]. Only a globe can represent accurately the shape, orientation, and relative area of the earth’s surfaces; any projection produces distortion with regard to some of these characteristics. The particular projection chosen for a given map will depend on the use for which the map is intended. Some projections preserve correct relative distance in all directions from the centre of the map (equidistant projection); some show areas equal to (equal-area projection) or shapes similar to (conformal projection) those on a globe of the same scale; some are useful in determining direction... Projections are classified as cylindrical, conic, or azimuthal according to the method of projection with a light source; many projections that can be constructed only mathematically are also classified according to this system“[32]

When in the geospatial information arena the term *accuracy* is used as an attribute to describe the degree to which a document’s geospatial representation adheres to the real world. *Precision* is an attribute that is closely related to the accuracy, precision represents the exactness or specificity of the geospatial representation (whether the coordinates are specified to the degree, the minute or the seconds level).

Fuzziness of locations is a significant factor for geospatial representation. Accuracy is sometimes important when representing boundaries. Prior to World War 2 when Soviet and Nazi Germany divided Poland between themselves, they used a pen that had a broad point and made a broad line, so the actual dividing line itself on the map covered several kilometers. After the campaign to take Poland had started, they had to meet again to make a more exact and accurate boundary between the two countries because of the granularity of the pen they had first used to draw the line between the countries.

But sometimes we don’t mean exact boundaries or locations. Inexact references can be very useful in some contexts. When describing where a person comes from inexactness can be very useful. If he says he is from Northern Europe most will know where that is. Scandinavia is a more accurate georeferenced location, but less would know where that is; Norway is more accurate but fewer still knows where that is; Troms is a county in Northern Norway, so more precise but mostly people in Norway would know the location of that county; Harstad is a city in Troms, most people in Northern Norway would know where that is; Grytøya is an island north of Harstad, most people in Harstad knows where that is; Lundenes is a region of Grytøya and all that live on Grytøya knows where that location is; Trøa is the placename of a location in the Lundenes region, but only the closest neighbors of this location knows of its name.

Various interpretations of a location's boundaries is also an issue fuzziness can handle. How a region is measured and what assumptions is used will affect where the boundaries of a region are placed and decide what a region constitutes of. Neither GIS nor cartography in general handle fuzziness well where a boundary must be placed somewhere, and users place more confidence in the lines on a map than they should.

“Specificity is forced on the representation even when the data may indicate a lesser level of confidence in the data points. For georeferencing usage outside of GIS and cartography, though, we would like to be able to live with levels of uncertainty and fuzziness more naturally, without being forced to specify a line somewhere. This is an issue for both representation and spatial retrieval.”[24]

Bounding boxes are most often used in georeferenced searching to limit the area to search in.

“Bounding boxes are rectangles based on the maximum and minimum longitude and latitude coordinates for the area of coverage - for the region.”[24]

According to Linda Hill the arguments for using bounding boxes are that they are:

- better than points and relative easy to obtain
- efficient to store and process for information retrieval
- “good enough” for a lot of purposes

She also lists the arguments against use of bounding boxes:

- they are not faithful to the shape of the area, especially when the area is; diagonal in orientation, irregular in shape, or consists of disjoint regions
- they cause lack of precision in spatial information retrieval because they retrieve items relevant to adjacent areas
- they apply to projected coordinates, and may thus not behave as expected on the Earth's surface (e.g., a bounding box for Antarctica)

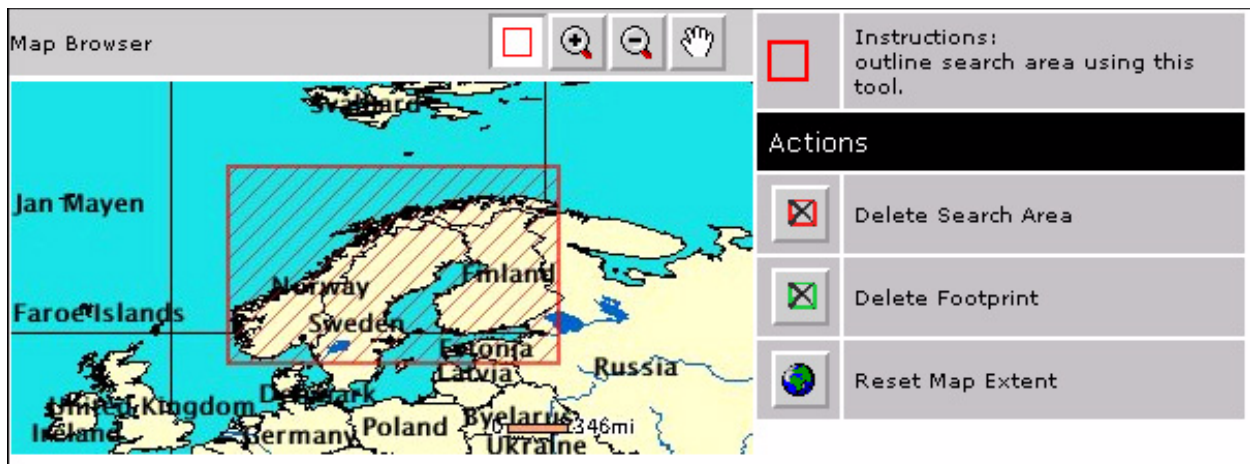


Figure 2-3. A bounding box that covers Norway also covers Finland, Estonia, and most of Sweden

Instead of using bounding boxes it's possible to make use of polygons or polylines to represent spatial values, they are much more faithful to the shape of an area, but are less efficient to store and process for information retrieval.

2. 2. 6 Geodetic Datum

Geodesy is the science of determining the size and shape of the Earth. Geodetic datums are referencing systems for points on the Earth's surface. There is a lot of different geodetic datums, and they are based on different measurements of the Earth as well as often being customized for the local or global applications they are to be used in. Peter Dana gives a good description of what geodetic datums are:

“Geodetic datums define the size and shape of the earth and the origin and orientation of the coordinate systems used to map the earth. Hundreds of different datums have been used to frame position descriptions since the first estimates of the earth's size were made by Aristotle. Datums have evolved from those describing a spherical earth to ellipsoidal models derived from years of satellite measurements.“

“Modern geodetic datums range from flat-earth models used for plane surveying to complex models used for international applications which completely describe the size, shape, orientation, gravity field, and angular velocity of the earth... cartography, surveying, navigation, and astronomy all make use of geodetic datums....“

“Referencing geodetic coordinates to the wrong datum can result in position errors of hundreds of meters. Different nations and agencies use different datums as the basis for coordinate systems used to identify positions in geographic information systems, precise positioning systems, and navigation systems. The diversity of datums in use today and the technological advancements that have made possible global positioning measurements with sub-meter accuracies requires careful datum selection and careful conversion between coordinates in different datums.“[33]

Datums have mostly been used in the GIS and mapping communities, but it is important to know which datum is used to describe a footprint for information that is georeferenced. Systems handling georeferenced information should also supply information about what datum is used (for each item in a collection or if same datum for all items then supply information for the whole collections datum)

WGS-84(World Geodetic System-1984) is a common datum used for general purposes.WGS84 is a datum maintained by NIMA (National Imagery and Mapping Agency), an organization in the US defence department. The GPS system uses coordinates based on the WGS84 datum.

EUREF89 is a datum based on regions for Eurasian (tectonic lithosphere plate). It became available in Norway in 1997. EUREF89 is meant to replace the datums NGO1948 and ED50 in Norway. The coordinates are expressed in a grid based system through UTM (Universal Transversal Mercator). Maps of Norway in the scale 1:50000, N50, uses the EUREF89 datum. “Statens

Kartverk” have decided that this is the datum that will be used in Norway when making public maps and measurements from now on.

2.2.7 Gazetteers

So what are gazetteers and are they significant in georeferenced information retrieval? Gazetteers provide translations between textual place references and geospatial locations. Gazetteers support a form of indirect geospatial referencing. If a user search for information regarding a specific geospatial location on the place of the Earth, he is not interested in finding only the documents containing the placename of that location, but **all** documents that contain information about that geospatial location whether the placename is in the document or not.

“A *gazetteer* is a list of geographic names, together with their geographic locations and other descriptive information. A *geographic name* is a proper name for a geographic place or feature, such as *Santa Barbara County*, *Mount Washington* and *St. Francis Hospital*. Imprecise areas such as *Southern California* can be included. Names such as *Abbeville 30x60 Minute Topographic Quadrangle*, *Grand Fort Tejon Earthquake Epicenter*, and *Habitat of the Red-legged Frog* are also legitimate gazetteer entries because they name identifiable geographic locations. *Geographic identifier* includes proper names for places as well as other types of identifiers: street addresses, postal codes, and *prepositional references* such as “across the Mall” and “5 miles south of the bridge” found in natural language descriptions.”

“There is remarkable diversity in approaches to description of geographic places and no standardization beyond authoritative sources for the geographic names themselves.”[24]

The National Research Council (NRC) *Distributed Geolibraries* workshop, June 15-16, 1998 identified gazetteers as a key component of geolibraries.[34]

Gazetteers play a key role when it comes to understanding georeferencing in text through use of programmatic text processing. The textual-geospatial georeferencing integration process is organized into a series of steps and each step have research and evaluation issues, but these are not focus areas in this paper.

Since the beginning of the DLI-1 funding period in 1994 the Alexandria Digital Library (ADL) has been engaged in major gazetteer development. ADL has developed a Gazetteer Content Standard and a Feature Type Thesaurus. ADL has reloaded the U.S. federal gazetteer data into this new model and added additional gazetteer data about earthquakes, volcanoes, topological map quadrangles, and political areas. This collection along with the ADL Catalog is used to answer both the “where is” and the “what’s there” queries in the ADL system.

The core elements that make up the gazetteer are:

- a name (placename, variant names)
- a location (footprint)
- a class/type (feature type categories)

Sometimes it's fruitful to link all names associated with a place to give some explanation or information of the names; the historical period it was used and reason for change of placename, language of the name, the etymology (derivation) of the name, or if it was an official name given by a naming authority is important knowledge for placenames.

The footprint of a place can be more complex than just a single footprint, but often it is sufficient to have a single footprint that indicates the general location of the place. Footprints representing the extent of the coverage and detailed boundaries of a place are often better and may often be needed for some purposes, but they are more difficult to obtain. A footprint for a place can in some cases vary over time: e.g., urban expansions, political redefinitions or if for example a river changes its river mouth.

The feature type is chosen from a scheme of classes, most often a hierarchical structure of classes (a thesaurus) or a list of categories is used. The ADL Feature Type Thesaurus is an attempt to create a scheme that can be used generally to categorize places, instead of using local sets of categories. Interoperability is difficult if local sets of categories are used, and it is therefore better to use a general scheme.

2. 2. 8 Geospatial information retrieval (spatial matching)

- Spatial matching operations and spatial similarity
- Geographic vs. spatial queries
- Defining spatial search regions

For more than two decades spatial searching have been available in the GeoRef bibliographic file[35]. Using the GeoRef bibliographic file it is possible to express a geospatial query as a text string, this however is very complicated and requires study to be done correctly. Following is an example of how to formulate a spatial footprint for a query in text form:

“Geographical coordinates are specified as follows: start from the southeastern-most point, then travel north, then west, until four corners of the area have been defined, e.g., Latitude 1, Latitude 2, Longitude 1, Longitude 2; SELECT N1=N36 AND N2=N38 AND W1=W118 AND W2=W120“[35]

The values and prefixes in the queries have to be entered in an expected order, this kind of query-building is not very user-friendly and error-prone (easy to make errors and hard to find and debug them). By using interactive maps and bounding boxes it is possible to hide complexity from end users, making the query-building process less error prone and more user-friendly.

Bounding boxes are efficient to process for retrieval purposes and have therefore been widely used in spatial retrieval and spatial representations. However, it is important to remember the limitations of bounding boxes; they are not faithful to the shape of the area, they lack precision because information from adjacent areas can be retrieved, and they apply to projected coordinates and may therefore behave unexpected on the Earth's surface.

More computation is required for other spatial matching operations than the bounding box, they are therefore more challenging to implement for large georeferenced datasets. Other ways to specify a spatial search region are:

- Polygon: Irregular shapes are used as bound areas. Bounding box searching is a simplified polygon search where the shape is regular. Polygons and bounding boxes can be used to formulate the *region queries* that Ray Larson talk about in [36].
- Point and radius: The user specify a location by a point, as well as the length of the search radius around the point. Points with a buffered radius can be used to formulate *distance and buffer zone queries*[36].

“Another treatment of points, as well as lines or paths, is to *buffer* them to form a generalized area surrounding the points and lines so that the location represents an area. Usually, the extent of the buffering is based on known characteristics of the locations (i.e., whether is a building, a lake, a small city or a large city).”[24]

- Point: Single points can be used to ask the question “What do we have at this X, Y point in the current coordinate system?” These kind of queries are called *Point-in-polygon* queries in [36].

Spatial matching operations and spatial similarity

The basic matching operations for spatial information retrieval according to Linda Hill are [24]:

- *overlaps*: finds documents whose footprint overlap the query region
- *is contained (within)*: finds documents whose footprints are contained within the query region
- *contains*: finds documents whose footprints completely enclose the query region

A query area and document footprints

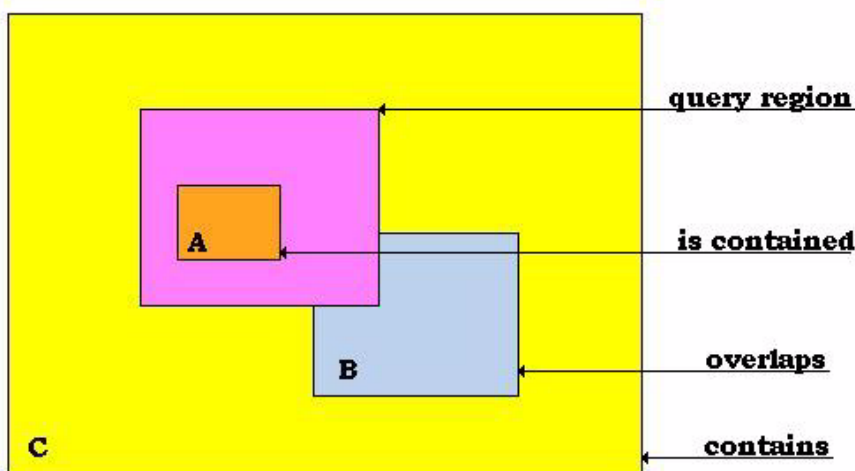


Figure 2-4. Spatial relationships[24]

Overlap is the most inclusive of the spatial matching operators, all the footprints in figure 2-4 are overlapping with the query region. If even a tiny bit of document A's footprint had been outside the query region it would not have been included in a "is contained" spatial match, the entire footprint of a document have to be within the query region to match such queries. A spatial match can be made if collections represent the geographic coverage of the documents with geographic footprints, and queries are expressed in terms of spatial coverage area (a geographic footprint) by using longitude and latitude coordinates. It is then possible to retrieve documents whose spatial coverage *overlaps*, *contains* or *is contained* within the query region. Gazetteers can be used to translate placenames, in both queries and documents, to footprints.

There are degrees of match when comparing a query region with the footprints of documents, and this can be used for ranking of retrieval sets. Ranking retrieval sets by spatial similarity measures is very useful, so it is possible to rank the most spatially relevant to the query as the top ranked information objects presented in the retrieval. One spatial similarity measure that is widely used is:

$$"2 \times (\text{area of overlap of A \& B}) / (\text{area of A} + \text{area of B})"[24]$$

The formula shows that the spatial similarity of two areas is based on degree of overlap between the two areas, and the size of the two areas. If area A and B are totally overlapping and equal size then the spatial similarity of the two areas equals 1.

Geographic vs. spatial queries

In [36] Ray Larson is talking about the differences between geographic and spatial queries. Spatial queries are the most general of these two, since they are based on spatial relationships (intersection, containment, boundary, adjacency, proximity) between entities geometrically defined and located in space. Spatial queries are resolved without regard to the nature of coordinate systems. Geographic querying assumes that the space is delineated by the well-defined coordinate systems of the "real world".

Geographical relationships in coordinate based systems imposed on the real world are geometric relationships, generally speaking. If geometry is used it is possible to solve many types of relationships between objects defined within a geometric framework.

"Spatial relationships can be both geometric and *topological* (spatially related but without measureable distance or absolute direction)."[36]

As examples of topological relationships Ray Larson include properties like adjacency, connectivity, and containment. It is possible to say that one house is adjacent to another without listing direction or distance between the two.

Defining spatial search regions

How is it possible to define query regions for spatial matching? Text-based spatial retrieval systems are complex, difficult for users to understand, and they makes it hard to enter spatial searches correctly, so what possibilities are there?

There are three common methods for specifying location in geolibraries[37]:

- Visually: by interacting with an onscreen map
- Formally: by specifying location in some appropriate coordinate system (longitude/latitude)
- Informally: by placename

When placenames are used, a gazetteer service is invoked to convert the placename to coordinates, this is usually done in a dialog with the user because of the need to resolve inherent ambiguities.

Visually drawing bounding boxes on an interactive map is an easy method of specifying a query region. Humans also have a liking for placenames when specifying what location they want to query. Very few humans know or remember longitude and latitude coordinates for different geographical search regions, placenames are much easier to remember.

The gazetteer service that translates placenames to footprints is analogous to the DNS (Domain Name Service) for the Internet that translate URLs (Uniform Resource Locator) to IP (Internet Protocol) addresses. Human cognitive recognition works much better with names than numbers, since numbers are far more abstract. It is much easier to remember placenames and link relationships to a placename than a longitude/latitude coordinate. If a person went on a vacation to Trondheim, he would use the placename Trondheim and not the coordinates for Trondheim:

Longitude: 10.41667(63 ° 25'0.012"W)

Latitude : 63.41667(10 ° 25'0.012"N)

when describing where he had been.

Through informal and visual methods for specifying spatial search regions, it becomes easier to define search regions for spatial and geographic queries.

Some user interfaces make use of geographic regions that are clickable. Such interfaces usually limit access to administrative divisions of the Earth: nations, counties, and municipalities. The content in collections is then usually indexed to these geographic areas that is clickable rather than using coordinates for a geospatial representation of the items in the collection. This is also a visual method for specifying location in geolibraries, another visual method is drawing bounding boxes on interactive maps.

2. 2. 9 Issues in Georeferencing

The following discussion about the seven issues in georeferencing is based on the [26] article written by Greg Janée.

Discovery

One of the most basic issues with georeferencing is **how georeferenced information can be discovered**. Traditionally textual placenames from the metadata of the information have been used for georeferenced discovery. For certain classes of information and in certain contexts this technique works well, but as a general technique it suffers from two serious drawbacks. **1)** Data gathered from moving sensors, like satellite imagery and aerial photography, have no associated placenames. So this class of georeferenced information carry no textual placenames, only techni-

cal metadata like coordinates and attitude of the camera at the time of exposure. 2) The unreliability of discovery based on textual placename matching. To illustrate this Greg Janée's example is used:

“Consider for a moment a user desiring a map of the *Flatirons* rock formations in the *Front Range of the Rocky Mountains* just outside of *Boulder, Arapahoe County, Colorado*. The highlighted words in the previous sentence all describe the area of interest with varying degrees of specificity, but it turns out that the most specific phrase (and the only *useful* phrase) that will retrieve data from the United States Geological Survey (USGS) for this area is *Eldorado Springs* (!), which happens to be the name of the topographic map quadrangle in the USGS's standardized grid system that covers the Flatirons. The USGS picks one prominent feature within each quadrangle to serve as the name for that quadrangle. If the USGS had included *Flatirons* in its cataloging, our hypothetical problem would be solved, but the USGS can't be faulted, for there is no easy or manageable way to associate every possible placename with every library resource. This is but one instance of a very general problem, namely, that there are many names—administrative, physiographic, technical (e.g., telephone and postal codes), temporal (e.g., earthquake and hurricane names)—for any given place, and a resource that is about a place can be cataloged by, at best, a handful of those names.”[26]

Controlled vocabularies are a structured (non-textual) way libraries often employ to deal with the general problem of a location having several placenames. To a certain extent a controlled vocabulary can solve the multiplicity problem with placenames, and with an advanced interface to the vocabulary it is possible to go a long way toward guiding a user in the selection of the proper term. But gaining agreement on a vocabulary is a classic problem, so is it to anticipate all the names users will associate with a place and adding them as alternatives or related terms. It is a considerable burden on users and library catalogers to agree upon, understand, and use a controlled vocabulary.

According to Greg Janée the state of the art is coordinate-based georeferenced discovery. Range searching over numeric longitude/latitude coordinates is more general and powerful. Range searching means searching with operators such as OVERLAPS and BETWEEN to empower searches. The downside with this is that it places the burden of expressing spatial coverages and query regions using coordinates on the users and libraries, but it also allows a user of a library to discover georeferenced information without the user and library agreeing on anything except the coordinate system.

Gazetteer integration in digital libraries

It is important to note that placenames are critical, important, and useful in georeferenced discovery, this is because human spatial cognition relies on relationships to and among known named features. But if humans are best at using placenames and library discovery is based on coordinate-based georeferenced discover, we need some kind of translation between the two discovery systems.

“Consequently, it is necessary that a *gazetteer* (a kind of dictionary that supports translation between placenames and coordinates...) be integrated into a georeferenced digital library to

help translate between the user’s mental model of geographical space and the library’s discovery system.”[26]

Integrating a gazetteer into a digitally library in a deep and significant way so that user queries are translated to the library’s discovery system in a hidden and seamless way would be the ideal way to make use of gazetteers. It is unfortunately not possible to support this level of functionality at this time, because gazetteer data is too incomplete, inaccurate, and imprecise. Gazetteer’s data is often gathered from existing maps, so significant geographic features like rivers and counties are represented by a single point even though they have significant extents. The point on the map may also be inaccurate because it may reflect the location of the label of the feature on the map, not the feature itself. The figure 2-5 below shows how a map(N50) and a nautical chart(S50) mark the point of the location of Grytøya differently.

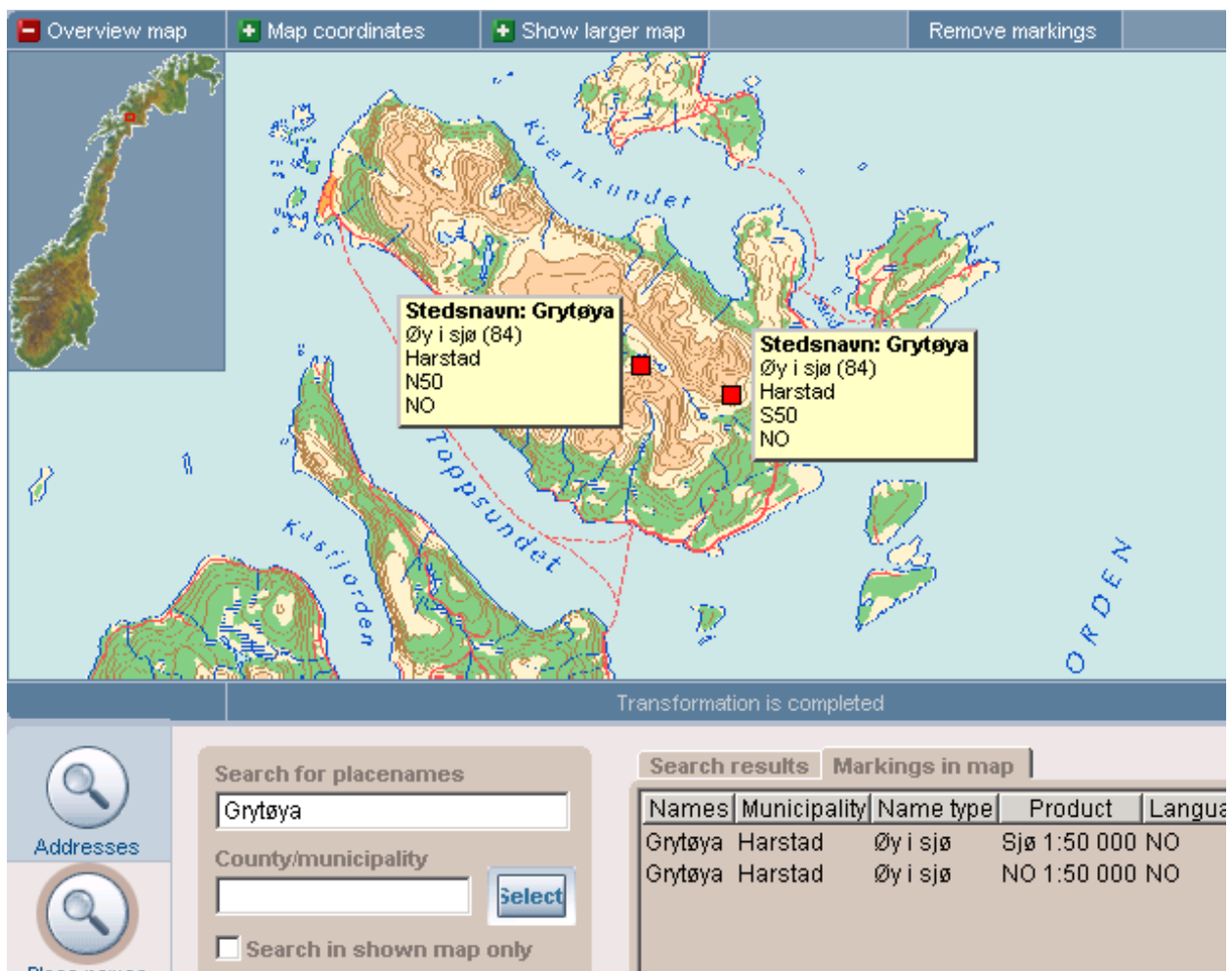


Figure 2-5. Norgesglasset[38]

Since seamless deep integration of gazetteers in digital libraries yet seem elusive, there is another way to make use of the gazetteers in an effective and easily achieved way; human-mediated gazetteer integration. Instead of making it a deep and significant part of the digital library it is possible to make it part of the library’s graphical user interface by treating the gazetteer as an

extension. The user enters a placename and uses the gazetteer extension with a background map to search for and locate the actual place. The extension then forms an appropriate coordinate-based spatial query-region to submit to the library.

Heterogeneous data

There are various metadata formats and content standards used to describe spatial coverage of georeferenced information, all use slightly different methods to describe it. Mapping mechanisms between the different formats and standards must be provided at some level if a digital library is going to accommodate for the heterogeneous data. There are multiple formats and standards in common use for geometry languages, coordinate systems, and georeferencing techniques; different geodetic datums, different types of projections, and different metadata formats to describe spatial coverage are examples of this.

Data typing

Usually digital libraries operate on text when they perform retrieval. Raw text searches performed by web search engines are not considered to be digital libraries. Information retrieval is usually performed searching metadata descriptions (text-based) of the information in the digital library. Metadata is a document surrogate that characterizes a document. There also exist Content Based Image Retrieval (CBIR) digital libraries like Infromedia-II that perform retrieval based on the content rather than a context/metadata description of information.

“Infromedia-II seeks to improve the dynamic extraction, summarization, visualization, and presentation of distributed video, automatically producing “collages” and “auto-documentaries” that summarize documents from text, images, audio and video into one single abstraction.”[39]

Text-based searching is the usual data type for digital libraries, even if some digital libraries use content based searching. The discovery and ranking is usually based on counting occurrences of given terms in metadata descriptions. When all that is searched is text-based the system only have to deal with one data type, text, but when adding a more structured discovery method like using geographic coordinate based range searching, more than one data type is needed. Spatial coverages cannot be treated as undifferentiated text, to use data types for spatial coverages it is easier to verify correct and incorrect input data. To have multiple data types means they have to be distinguished and handled, this makes a system more complex and can lead to complications compared to text only data type digital libraries. Greg Janée lists some of the ramifications of this:

- “Input validation is required. With text, a digital library can take advantage of any structure it recognizes (metadata fields and other markup, for example), or it can fall back to a base level interpretation of text as a byte stream. With geographic metadata (descriptions of the spatial coverages of library items, for example) there is no such fallback: either the metadata is recognized and in a usable form, or it is not.
- Relatively complex internal structures and external representations are required to describe spatial coverages and geographic query regions, particularly if higher-order kinds of regions such as polygons are supported.
- Syntactic and semantic heterogeneity of external representations becomes an issue. There are multiple kinds of geographic regions, multiple ways of describing regions, and multiple appli-

cable metadata standards in use. Even for the simplest kind of geographic region, boxes, there are both syntactic and semantic differences between... metadata formats. In addition, there are the classical cartographic problems of projection and datum conversion... A digital library that strives to accommodate this heterogeneity will have to provide mapping mechanisms at some level.

- The library must provide the means to express and execute different types of query constraints (spatial, textual, etc.) as well as Boolean combinations of different types of constraints. The latter are particularly valuable, but can be particularly difficult to implement, as discussed in the next section.
- Specialized user interface components are required to input and to view any kind of strongly typed data. In the case of geographic coordinates, this means integrating an interactive map browser into the library's user interface, which in turn requires underlying map and gazetteer servers. “[26]

Scalability

Often geospatial data is generated automatically by satellites and remotely sensed imagery, therefore it is easy to accumulate geospatial data. This means that scalability becomes an issue when accumulating a very large number of library items.

“Georeferenced discovery techniques are nicely scalable in theory..., but in practice, scalability is more limited. At the time of this writing, commercial georeferenced search engines reach the limits of practical use (say, one day of compute time utilizing several gigabytes of RAM and unlimited disk space) when indexing as few as one million items. More items can be accommodated, of course, but only with custom engineering and exponentially increasing expense.”[26]

Another important issue with databases and their scalability is database join operations on large datasets. To perform boolean combinations of spatial constraints with textual and other types of constraints means databases will have to do join operations on huge datasets.

“When large numbers of items are involved, generating efficient query plans becomes paramount, but query plans are unavoidably sensitive to the queries and to the distribution of the data, and the data is inevitably not uniformly distributed. Whether the library is working within the framework of a single relational database or across multiple, distributed search indexes, the heterogeneous join problem is a significant practical problem, and a research problem as well... “[26]

Context for spatial searching

Users need context to be able to orientate themselves in an information space. The context allow users to understand what the information is about and makes them able to formulate queries and interpret the resultset of those queries.

If a digital library supplies an interactive background map that is integrated with the user interface to aid in geospatial searching, it is important that the map supply sufficient details so users can recognize landmarks and relate their locations to the landmarks. The map should give the user the

context needed to formulate queries. It is also important that the result sets from queries give enough context so the user understands what is retrieved.

Searching in a system containing geospatial information a search for the mountain *Toppen* might for example return the map below in figure 2-6. The mountain is on the map, but it is nearly impossible to find it because it has not been labeled with the placename *Toppen* and it's not marked where it is on the map. The map shows several mountains, some have been labeled with a name, *Toppen* is not labeled and is therefore nearly impossible to find unless already familiar with its' location. Context is important because it makes a user able to understand what has been retrieved, and the relevance of the retrieved information to the user.



Figure 2-6. Map of Grytøya. Source: Harstad municipality's map[40]

Now consider the context the map in figure 2-7 provides, and find Toppen there.

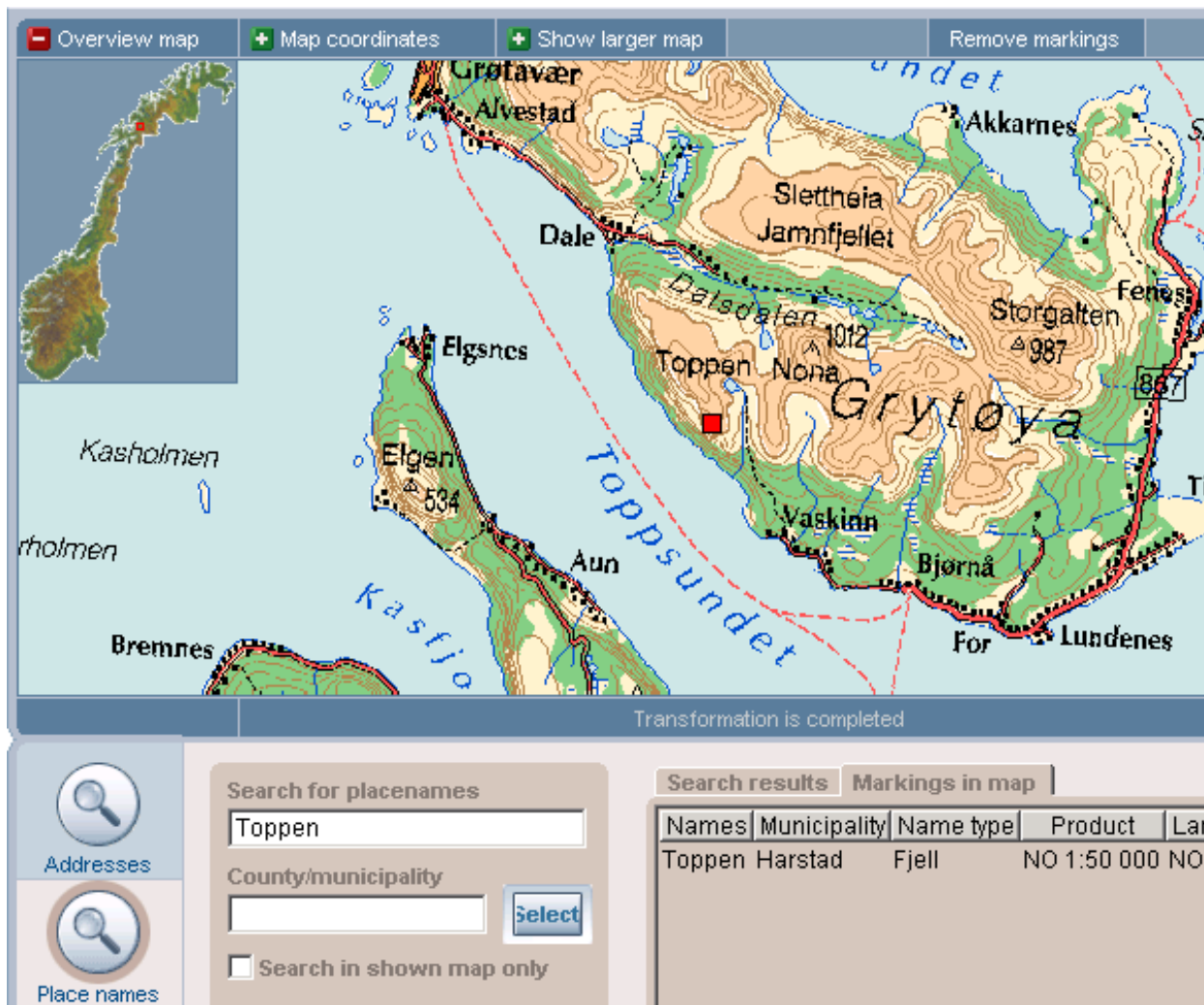


Figure 2-7. Map of Grytøya[38]

How much use would the satellite picture in figure 2-8 be of to a general user in its current state without any context?

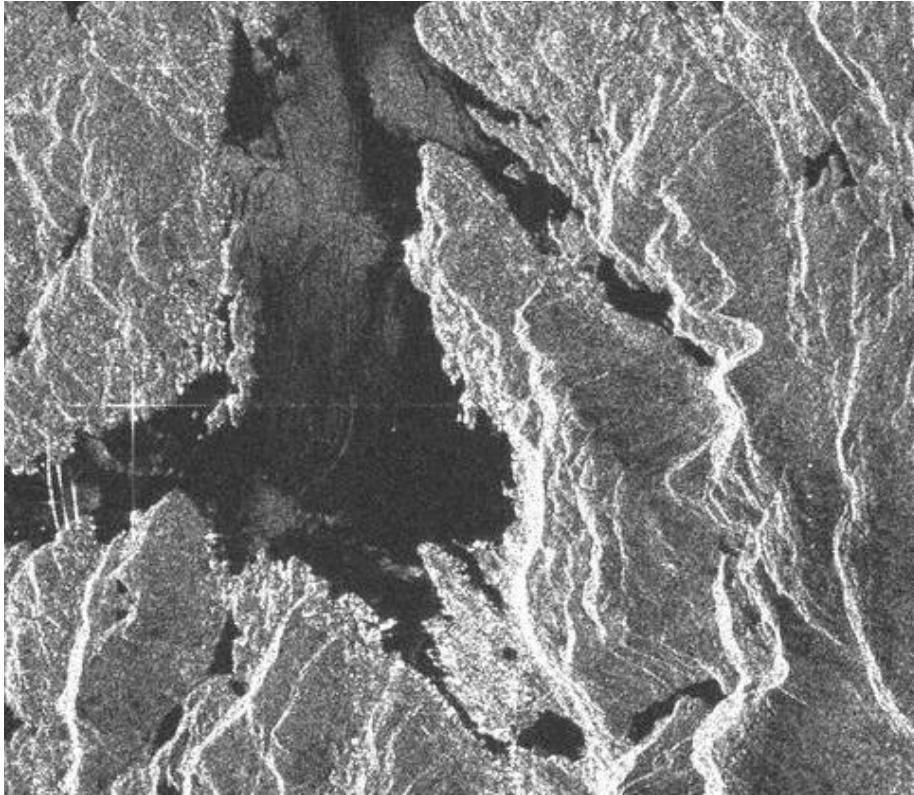


Figure 2-8. Picture of Bergen by the ERS-2 satellite. Orbit 36370 (ascending) Frame 1208, April 4, 2002 21:29 GMT[41]

For a map/remotely sensed imagery/satellite picture to be useful for a general user, a digital library must provide a certain amount of GIS functionality so the user can view labels of features and placenames within the picture, or place the picture over a reference map that contain placenames and labels of features.

Resource access and integration

What implications does geospatial resources have on digital libraries? Geospatial resources can be accessed via multiple different formats and protocols, and some formats require special interfaces to be able to view the geospatial resources. The resources often have complex structure, consisting of multiple parts, and can often be quite large. The ability to meaningfully process geospatial resources strictly within the confines of a standard web browser is limited, because the resources are closely tied to geographic information systems or other types of data exploration environments. In some cases the resources are entirely unusable outside of the appropriate analysis or visualization system.

“These characteristics of geospatial resources—their complex structure and close ties to analysis environments—call out the need for georeferenced digital libraries to be able to describe, in detail and for the benefit of both programmatic clients and human users, the components that make up a library resource and the different modes by which it may be accessed. It is insufficient to describe the content of a geospatial resource by, say, a single,

unadorned URL ("click here"). A more detailed description allows the user to make an informed decision about the best way to access the resource, and can facilitate a seamless handoff of the resource from the library to a more capable analysis or visualization tool.“[26]

2. 3 Georeferenced Digital Libraries

2. 3. 1 Georeferenced Digital Libraries

What does the term georeferenced mean in the Digital Library paradigm?

“*Georeferenced* means that, whenever possible, each item in a library is associated with one or more regions of the Earth’s surface. (We refer to these regions as the item’s footprint.)“[42]

Charles C. Cutter[43] stated in 1904 that the purpose of the catalogue is to offer the user a variety of approaches or access points to the information contained in the collection of library by enabling a person to find any work when one of the following is known:

- The author
- The title
- The subject

The catalogue also has an additional objective to assist the user in the choice of a work and to show what the library has:

- By a given author
- On given and related subjects
- In a given kind of literature

In georeferenced digital libraries the access points of digital libraries exist as postulated by Charles C. Cutter (he postulated them for traditional libraries, but the same access points are used in digital libraries), but additionally the ability to access any work based on its spatial georeferenced location is added as a point of access. Everything that happens, happens in time and in space, so being able to interpret problems through this paradigm offers new access points and new powerful ways to search for information. To be able to retrieve information about a geographical location is something new, “show me all information available about this location”.

3 State of the Art

*“Get your facts first, and then you can distort them as much as you please.”
- Mark Twain*

3.1 A fully fledged georeferenced digital library

What is a fully fledged georeferenced digital library? There exists a lot of different digital library systems today that allow various methods of georeferenced searching. A fully fledged georeferenced digital should deal with the seven issues that arise when creating a georeferenced digital library.

It should provide **discovery** of the georeferenced resources in the collections. The system should **integrate a gazetteer** in the system to aid in the discovery of resources, and **specialized ranking** of search results based on georeferencing should also be implemented. Some georeferenced resources require sophisticated mechanisms to **access** them, therefore a fully fledged georeferenced digital library should supply the mechanisms necessary to operate on them. The volumes of information that can be stored in a georeferenced library can be mind boggling, thus the **scalability** of such a system is important. **Strong data typing** should also be supported. Strong data typing and scalability are implementation issues. The georeferenced digital library must also provide **spatial context** to the users of the system, this is a critical issue in the user interface.

The concept of a georeferenced digital library is well defined in the quote below:

“A georeferenced digital library is an information system that stores georeferenced resources, and moreover provides a spatial orientation to those resources in terms of discovery, browsing, viewing, and access.”[26]

A fully fledged georeferenced digital library should support all three methods (visually, formally, and informally) described in [37] for specifying locations in geolibraries. It should also include a gazetteer service that converts the placenames to footprints. The coordinates of the footprint should be essential in the retrieval of georeferenced information objects from the collection(s) in the georeferenced digital library.

Spatial context in the user interface should also be provided to make it possible for a user to interpret, evaluate and utilize the georeferenced resources.

“...a fully developed georeferenced digital library must provide a certain amount of its own (lightweight) GIS functionality. At a minimum, a digital library must make it possible for users to evaluate georeferenced resources, and particularly geospatial resources, for query relevance.”[26]

3. 2 Comparison of georeferenced systems

A comparison of different georeferenced systems will show how georeferenced capabilities and aspects are functioning and implemented in various systems. What system supports the most geospatial features, and comes closest to the above concept of a fully fledged georeferenced digital library?

3. 2. 1 Investigated system aspects

- Purpose: For what purpose was the system built? What questions or problems is it supposed to solve?
- Distributed: Is the system based on a centralized database or does it use autonomous distributed databases and some sort of distributed search algorithms? Is it a centralized portal, or are there many searchable nodes? Does the system use distributed services?
- Geospatial search: What methods for specifying location is used? What spatial context is used?
- Gazetteer service: Does the system provide a gazetteer service? If so, what gazetteer is used?
- Data storage: What metadata standards/formats/frameworks are used for data storage? Does the data have to be stored in any special kind of database?
- Applications: What services are offered to the end user? How are queries input to the system? How was the ease of use, and its usability? Ease of use and usability of systems can vary from user to user due to individual differences, e.g. domain knowledge, computer experience and computer self-efficacy (an individuals judgement of one's capability to use new information systems).

Since the total number of geographical information systems is too great to allow a complete analysis, a small subset has been selected which we will take a closer look at. When choosing systems, the most important criterion has been that the systems must seem to offer some of, or all of the possibilities offered by a “fully fledged georeferenced digital library”. The information below has been found in published documentation of the systems, and through e-mail contact with some of the projects.

In the following subsections these systems are compared:

- Norgesglaset
- Primus
- EDINA: Go-Geo!
- Galleri Nor
- DLESE
- ADL

3. 2. 2 Norgesglaset (Norwegian map service)

Purpose:

Norgesglaset[38] is a digital map service for displaying maps on the web. It's developed by the National Land Survey (Statens Kartverk) of Norway.

According to the website information Norgesglasstet aims to:

- Enable searching for addresses and placenames (more than 450000 placenames in Norway) and to place the placenames on the map
- Navigate digital maps (zooming, etc.)
- Transform coordinate sets
- Explain the path to a place for people that are unfamiliar with that place

Norgesglasstet has almost 35000 inquiries each day, so it is a service that is popular in Norway (Norway only have about 4.5 million inhabitants).

Distributed:

Norgesglasstet is not made to be a distributed system, but to be a web service for displaying interactive online maps. Norgesglasstet uses several databases that can reside on different machines.

Geospatial search:

It is possible to perform informal searching by placenames, street names, county, municipality, and postal codes. It allows visually interaction with the onscreen map. The system supports formal searching via the interactive map. If the button for the Map coordinates is clicked, then the coordinates for what the mouse pointer hovers over are listed in real-time. Drawing a box around the location will zoom in on it. Norgesglasstet supports two different methods for specifying location in geolibraries (visually and informally). Map coordinates can be displayed, but it is not possible to enter map coordinates into any fields to define a search region.

Spatial context is supplied to the user through the interactive map listing placenames, roads boundaries, and other common map features (see figure 3-1).



Figure 3-1. Norgesglasset user interface[38]

Gazetteer service:

Retrieval services in Norgesglasset are making use of several databases/registers:

- Sentralt Stedsnavnregister (SSR) containing about 500 000 placenames
- Addresses (updated addresses from the national GAB-register)
- Historical maps
- A transformation service based on the official transformation-routines.
- Maps (of several different scales)

Norgesglasset does not reference to other data than what is available to the National Land Survey

SSR is the official Norwegian gazetteer, so Norgesglasset does indeed make use of a gazetteer service. The SSR gazetteer contains about 500 000 placenames, including approved alternative spellings and variants. Placenames are accurate within 50 meters and have been added manually from an official map series (Norge 1:50000).

Data storage:

There is no collection of documents attached to the Norgesglasset service. It was not designed to be a georeferenced digital library.

The service is based on 5 raster-based maps that cover Norway in the scales 1:2 millions, 1:1 million, 1:250 000, 1:50 000 and 1:5 000.

Applications:

Users of Norgesglasset can search for addresses and placenames by keyword searching. It is also possible to display the placenames on the map. The interactive map can also be used to find coordinates of locations (for example it can be verified that Nordkapp is not the northern most part of mainland Norway). It can be used to explain the path from point A to point B as a traditional map. Norgesglasset is easy to use when the zooming and the overview map features are understood. Use of the overview map to target what region to view and use of the zoom features to get the wanted detail level. Norgesglasset is always available on the web, so it is very accessible and free of charge.

But is it a fully fledged georeferenced digital library or even a georeferenced digital library? No, because the system can't retrieve any documents from the places found on the map. If the interface of Norgesglasset could be used to retrieve documents from georeferenced collections it would be a georeferenced digital library.

3. 2. 3 Primus (Norwegian digital library for museum collections)

Purpose:

According to the Primus[44] web page the purpose of the Primus project is to make a database system to register, manage and present museum collections.

The Primus project was started in 1996 as a cooperation between some Norwegian museums. The project was initiated for development of a complete information system for registration, management, and presentation of museum collections. Primus has been available to all Norwegian museums since april 2001. The Primus system is used by roughly 45 institutions in Norway, for example Sverresborg (The Trøndelag Folk Museum).

Key features of the Primus system are:

- Hierarchical information storage
- No data is erased, changes are stored. Registration history
- Material independent registering
- The histories of the artifacts are registered
- Each artifact has a digital image attached to it (possible to add video presentations also)

- Each artifact can have several happenings attached to it
- Each happening can be tied to one or more places/locations

Distributed:

Primus only supports one database system, Oracle. The Oracle database should run on a single server, but the Primus clients can be installed on an unlimited number of other computers in the organization's network. If the Oracle database crash, the entire system will go down, so it have the characteristics of a centralized system.

Geospatial search:

It is possible to perform informal searching for geographic locations. The system does not make use of visually searching via interactive maps, or use of the formal method of searching by coordinates. It is possible to add longitude/latitude coordinates through use of the "fact" metadata field, but there is no specific field made for such metadata. Each location record in Primus registers: country, county/municipality, area, area description, address, land no. / title no., and fact fields about the location.

Gazetteer service:

No gazetteer service is used in the Primus system.

Data storage:

Primus is based on an extension of "Feltkatalogen for Kunst- og Kulturhistoriske Museumer", a metadata format used by museums in Norway. The Primus system is limited to using Oracle databases, so all collections must be stored in Oracle database solutions.

Applications:

The users of the Primus are people working on museums that register museum items. Primus is used for registration, management and presentation of items in the collections of the museums.

Primus is not primarily a georeferenced digital library system, but it has some capabilities to perform informal searches via placenames. There is no gazetteer implemented in the system, and it was never implemented to be a georeferenced digital library. The museum collections contains georeferenced materials, each item in a collection is made or found somewhere, and it can have multiple happenings attached to it, and each happening has taken place somewhere in space and time.

3. 2. 4 EDINA: Go-Geo!

Purpose:

"...EDINA's discovery and access tools endeavour to abstract the complexities into simple point and click interfaces. The primary objective is to achieve maximum dissemination of valuable data (in terms of both cost and research potential) to the widest possible audience with the minimum of obstacle and delay."[45]

Go-Geo![46] is an online resource discovery tool that allows for the retrieval and identification of spatial metadata records.

“It demonstrates the utility of a *data locating* tool for geospatial resources and the critical role a more comprehensive *one-stop shop* geo-portal for spatial data and basic mapping services might provide.”[45]

Looking at strategies for promoting metadata creation is another core part of the Go-Geo! project, also looking at cost-effective mechanisms by which academic researchers can publish their data for use by others is another aspect of the project.

According to the FAQ page for Go-Geo the aims for the project are:

- to develop a geo-spatial portal suitable for roll out as a full service in the JISC Information Environment
- to promote the possibilities of such a service
- to identify metadata providers and also services for inclusion in any cross-searching
- to produce a report identifying the relevant stakeholders and explaining how the needs of the user groups will be met

Distributed:

Go-Geo! is based on the Z39.50 protocol (plus it uses the GEO and BIB1 profiles for Z39.50), this protocol is used to search remote databases. Go-Geo! is a distributed system with a web portal search interface that users can use to retrieve information from various collections in the UK.

The distributed nature of using the Z39.50 lead to problems with ordering of results (sorting and relevance), and searches could be slow or time out completely. Therefore an index database was created with a Z39.50 server interface which contained enough of the ‘discovery’ metadata from all the Z39.50 targets to allow searches from the portal to be carried out locally. Collections that contain geospatial datasets with coordinates and that have a consistent format (FGDC XML) can be locally indexed, other resources with no explicit geospatial content (COPAC, REGARD, GCMD) have to use standard remote Z39.50 searching.

A spider was implemented to crawl the Z39.50 targets and harvest information for the local index. Go-Geo! always retrieves a live record from a Z39.50 target when the user requests a record. Records viewed are always current metadata even if the indexes are out of date.

Go-Geo uses autonomous collections that can be searched and retrieved via the Z39.50 protocol. The metadata search engine on the web portal site undertakes a search of a set of indexes which describe the metadata held at remote Z39.50 targets. The result list from the query provides links to the resources so the records themselves can be retrieved from the Z39.50 targets.

Geospatial search:

The geographic area of interest can be defined by placename, administrative area, grid coordinates or by using a map. Content from the UNESCO thesaurus is used in Go-Geo!’s advanced search page as a controlled vocabulary, a pop-up page contains an interactive keyword thesaurus[47].

So Go-Geo! supports informal georeferenced searching via placename in the simple search interface and in the advanced search interface. Formal specifications can be made by coordinates using Easting and Northing for the National Grid of the United Kingdom in the advanced search interface. It is also possible to interact with a map to automatically get the Northing and Easting coordinate values for the advanced search, so Visual searching is supported also. After the search results are presented to the user, it is possible to select what records to highlight in a map and display them in a map view (see figure 3-2) to see the geographic coverage. Searches can be sorted by geographic or lexical relevance to the query.

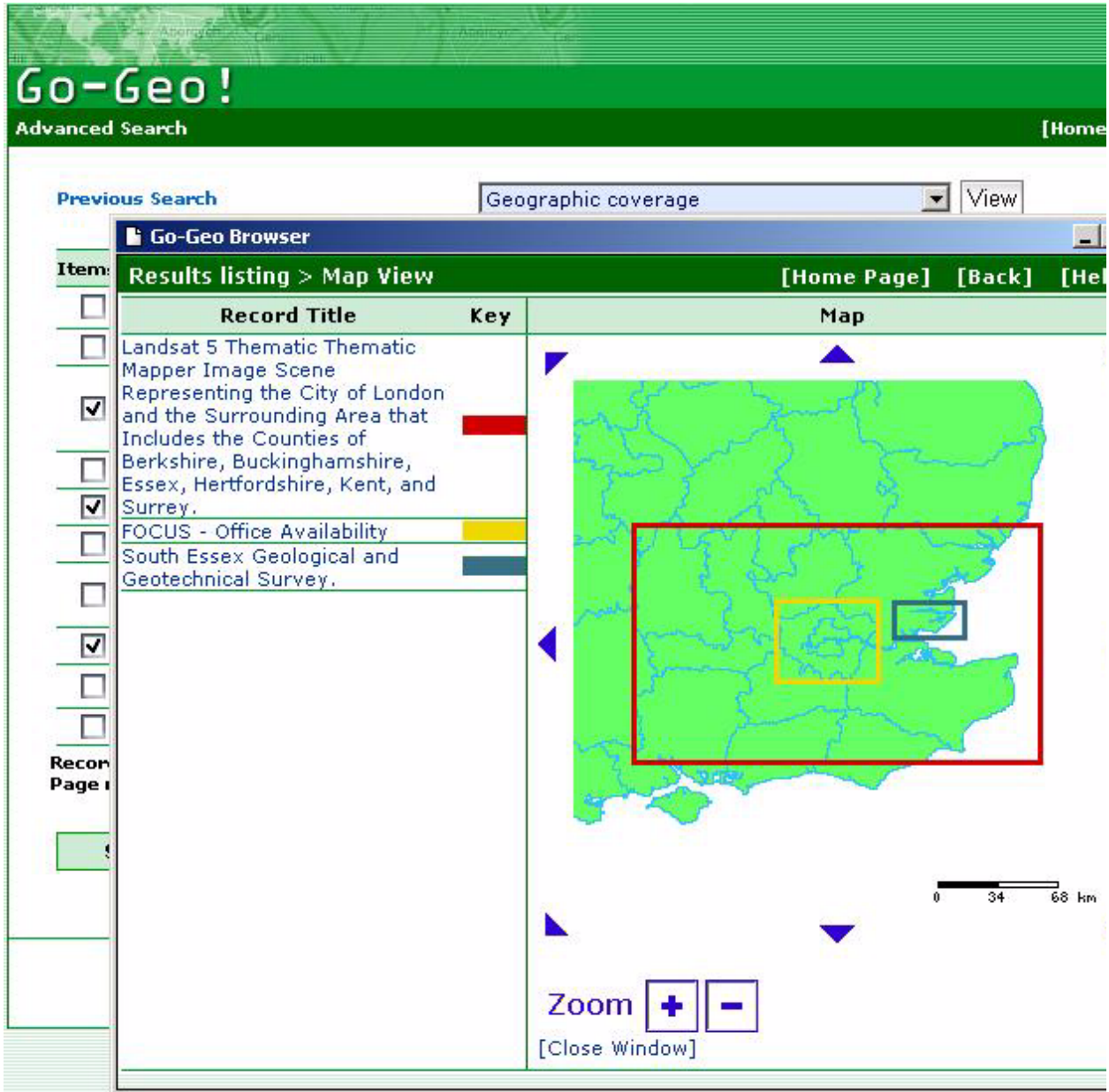


Figure 3-2. Go-Geo! result page after a simple search (placenames appear when zooming in)

Go-Geo! supports all three methods of georeferenced searching as well as being able to highlight in map the geographic coverage of the search results. Go-Geo! has an element of GIS in it, GIS is used to produce the maps which appear under the “*Where*” tab for each record in the search results. The rest of the service does not involve the use of GIS. The metadata records describe the content, quality, condition and other characteristics of geospatial datasets.

In addition to geospatial searching Go-Geo! also supports traditional topic and keyword forms of searching including support for both controlled vocabulary searching and free text searching.

The Go-Geo! tool is a UK portal and therefore Go-Geo!’s current geographic scope is the UK.

Gazetteer service:

Go-Geo! uses a gazetteer service called geoXwalk[48]. GeoXwalk is another project run by EDINA and other partners. It is a middleware server which can translate between multiple geographies (postcode <=> placename, grid ref. <=> placename etc.). GeoXwalk can be used in a two step process to convert a placename to a footprint, and then convert the footprint back to a list of placenames that can be used to expand the query.

Data storage:

To add collections to the Go-Geo! system the metadata of the collections must be written in the FGDC or ISO 19115 metadata standards. XML metadata conversion functionality has been developed to convert between different metadata formats used inside Go-Geo!. Rather than providing a layer like the bucket framework that makes heterogeneous access to collections possible, Go-Geo! converts between different metadata formats.

Applications:

Go-Geo provides a user-friendly georeferenced digital library where it is possible to do searches for georeferenced information objects from the UK. The Go-Geo! tool supports informal georeferenced searching via placenames. Formal method and visual method for specification of location are supported in the advanced search interface, all three methods are thus supported in Go-Geo!.

Currently the Go-Geo! project supports only the FGDC and ISO 19115 metadata standards and not heterogeneous metadata standards. The scope of the portal is currently limited to the UK because it’s a UK portal. It’s a very user-friendly system that is easy to understand and make use of, and it implements all of the 3 different methods for specifying geographic locations.

3. 2. 5 Galleri Nor

Purpose:

Galleri Nor[49] is a photo archive at the National Library of Norway in Mo i Rana. It is a collection containing digital versions of photos taken in the time period 1880 - 1950, each photo also has a metadata description. The photos come from various locations all over Norway. Galleri Nor will in the future contain photo-material from various central norwegian photo collections. Galleri Nor contains over 70000 photos, and is publicly available only through the web. The collection has existed since 1996 and the current new web graphical interface was created in 2004. The Archive is updated whenever new material is added or existing material is changed.

The Galleri Nor collection aims to make photo collections available to the public digitally through the web, which is the main purpose of Galleri Nor.

Distributed:

Galleri Nor is stored in an Oracle Database and uses the norwegian “Feltkatalogen” as metadata format for the photos in the collection. The web interface of Galleri Nor offers searching and retrieval of the information objects stored in the single Oracle database where the Galleri Nor collection is stored, so it is not really a distributed system even though the database and web interface can reside on different computers.

Geospatial search:

Searching can be done using search keys as: topic/motif, country/county/municipality, place-names, name of photographer, name of person in photo, name of institution, date, and numbering.

The Galleri Nor collection contains photos that are georeferenced using the placenames, the photos are from various places in Norway. The Galleri Nor system doesn’t make use of any GIS components. Galleri Nor supports the informal method of specifying location via the use of placenames, country, county and municipality.

Galleri Nor has no interactive map and there are no coordinates stored in the metadata description of each information object, so it is not possible to use of coordinates in georeferenced searching.

Gazetteer service:

Currently no gazetteer service is used for query expansion. Marit Olsen at the Norwegian University of Science and Technology has done a master thesis [50] that shows how automatic query expansion and use of a gazetteer improve the recall when searching the Galleri Nor collection. The improved recall can in turn improve the precision somewhat, but this is not implicit.

Data storage:

“Feltkatalogen” is the metadata format used for the Galleri Nor collection. The collection is stored in a Oracle database.

Applications:

The Galleri Nor web page offers free of charge access to the Galleri Nor collection at the Norwegian National Library. Galleri Nor contains more than 70.000 photos from all over Norway. The photos are from various collections all taken by many different photographers. Since the photos cover the time period 1880-1950 they can be used for history research, as well as other purposes.

It is possible to search the Galleri Nor collection using informal method for specifying location. Placenames and country/county/municipality can be used as keywords to search for. The user interface is simple, user-friendly, and easy to understand (if norwegian language is understood). Results that are retrieved are relevant, but by making use of a gazetteer it is possible to increase the recall and precision. The photos are easily accessible through a web browser.

The Galleri Nor collection is well suited for a georeferenced digital library since the photos in the collection are georeferenced by placenames. Currently it lacks longitude/latitude coordinates and only the informal method of specifying a geographic location is used.

3. 2. 6 DLESE (Digital Library for Earth System Education)

Purpose:

DLESE (Digital Library for Earth System Education)[51] is a free service funded by the National Science Foundation. It is a geoscience community resource that supports teaching and learning about the Earth system. DLESE is being built by a community of educators, students, and scientists to support the Earth system education at all levels. The use of DLESE is free of charge, and the vast majority of the resources that DLESE can discover are free as well.

Currently DLESE provides:

- Easy access to quality teaching and learning resources about the Earth as a system for a wide range of learners
- Services to help users effectively create, use, and evaluate digital learning resources
- Interfaces and tools to allow student exploration of Earth data
- A community center that fosters interaction, collaboration, and sharing

The resources DLESE provides access to are web-accessible teaching or learning materials such as: lesson plans, scientific data, visualizations, interactive computer models, and virtual field trips. The resources are organized in collections that reflect a coherent, focused theme. All resources in DLESE are relevant to the Earth System education, and they are all contributed by community members.

DLESE resources that are considered exemplary after a close examination are added to the Reviewed Collection, and this is a form of verifying quality resources. The resources added to the Reviewed collection are reviewed for scientific accuracy, pedagogical effectiveness, ease of use, clarity and completeness of documentation, ability to motivate learners, robustness, and significance of content.

DLESE also supports educators and learners through a suite of services like; peer-review systems, workshops and meetings that serve as catalysts for community action, technical training to support future digital library developments, tools and interfaces to explore Earth data, and the sharing among the DLESE community.

The DLESE portal serves as the geoscience node in the National Science Digital Library (NSDL). The NSDL serves the broader communities of science, technology, engineering, and math education. DLESE is a discipline-oriented collection in the NSDL, and most DLESE records can be discovered in the NSDL.

The community is fundamental to the DLESE vision, and DLESE is governed by an elected Steering committee who provide oversight and guidance. There is an annual meeting that provides a forum for broad community input into digital library future development for DLESE. This

is a digital library made by a community for a community where user input is essential for the use and development of the digital library.

Distributed:

DLESE can conceptually be considered a three layer architecture: interfaces, services, and collections. The digital library is built in a distributed fashion, so any particular interface, service, or collection can be located anywhere on the Internet. Services support specific tasks, such as searching, browsing, reviewing, and so on. Services interact with each other through agreed upon protocols or Application Program Interfaces (APIs).

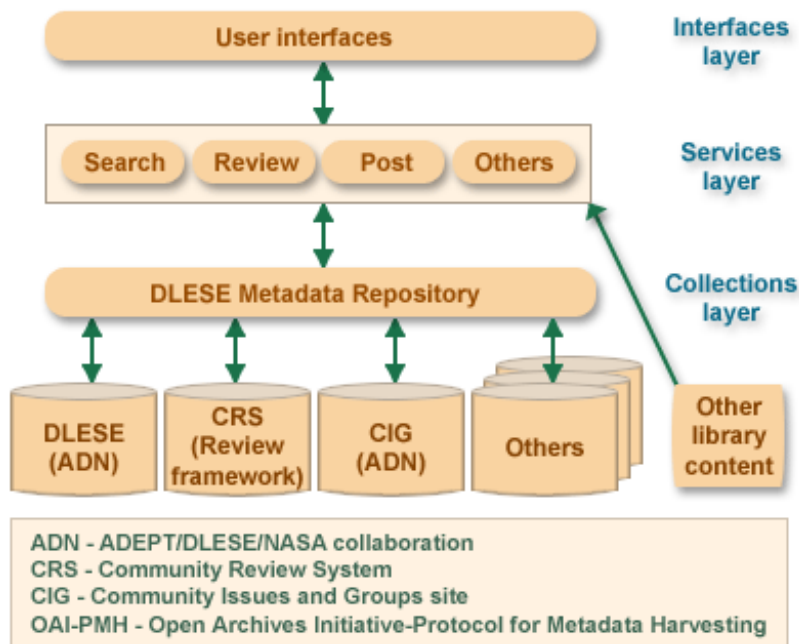


Figure 3-3. The three layer architecture of DLESE[51]

The “collections layer” encapsulates both collections with metadata, and other web-based content that don’t necessarily have metadata. Collections share metadata with the DLESE metadata repository. Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) is used to link the various DLESE collections with the DLESE metadata repository, and allow individual services to work with the repository and with other services. Since the OAI-PMH is the interoperability mechanism used by the National Science Digital Library (NSDL), this allows DLESE to be interoperable with the NSDL.

The “collections layer” shows that DLESE doesn’t host the resources, but link to autonomous distributed collections of resources. DLESE can build virtual collections from selections of collections from various collection providers. The DLESE web portal provides searching in metadata records that describe resources located in autonomous distributed collections.

DLESE has adopted digital library policies, standards, and technologies to help ensure that distributed building efforts yield a coherent and reliable end-user digital library experience. DLESE is being developed as a distributed digital library system.

Geospatial search:

The version 3.0 of DLESE that is projected for 2005-2006 will support georeferenced discovery and discovery of Earth system events. The current search interface supports informal specification of location through entering placenames as search terms. By default, the system searches all of the resource records' title, description, keyword, creator's last name, subject, and resource type fields. It also searches over the terms in the resources' URLs.

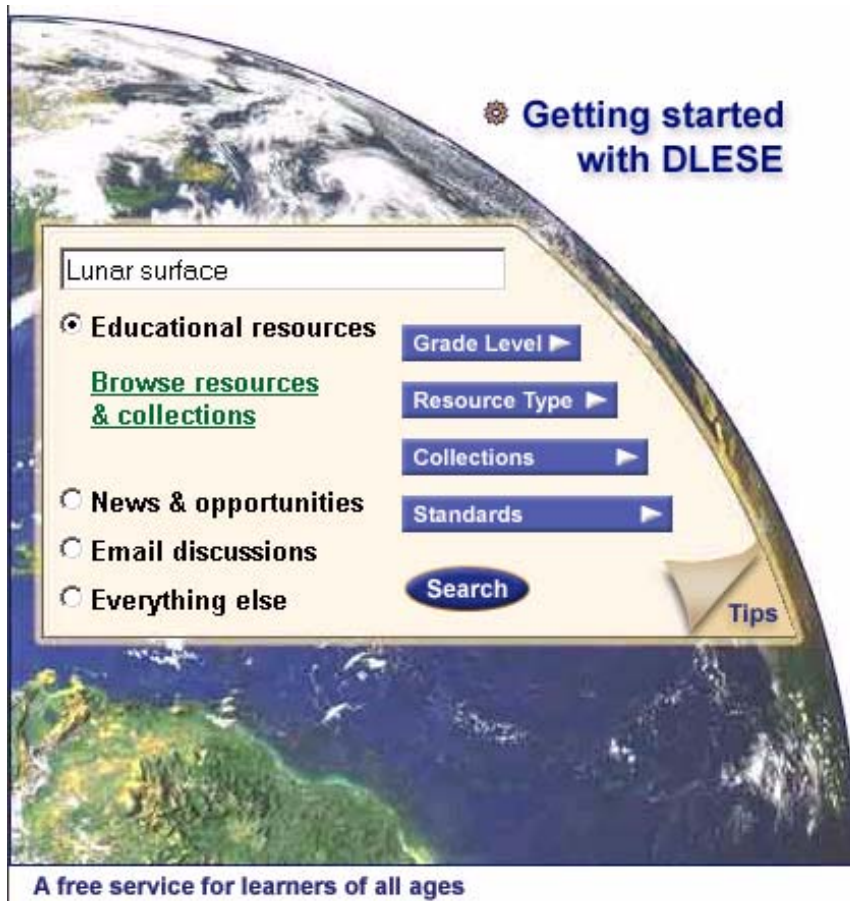


Figure 3-4. The search interface of DLESE[51]

Searches in DLESE retrieve a set of metadata records describing materials that match the criteria that is searched over. Each metadata record provide a link to the resource. DLESE doesn't host any resources, but they do check to be sure that the resources described in metadata records remain accessible and functional.

The screenshot shows the DLESE search interface. At the top, there are navigation tabs: Educational Resources, For Educators, News & Opportunities, People & Groups, For Developers, and About DLESE. Below these is a search bar with the text 'Lunar surface' and a dropdown menu set to 'Educational resources'. There are also filters for Grade Level, Resource Type, Collections, and Standards, along with a 'Clear selections' button. The search results section shows 'Results 1-7 of 7 for 'Lunar surface''. The first result is 'Lunar Phases Web Tool' with the URL <http://www.calvin.edu/~lmo/nar/moon/index.html>. The description of the tool is: 'Intended for use in introductory astronomy courses, this Java-based tutorial demonstrates lunar phases clearly and logically. It includes day, time of month, and apparent direction of the moon. The Java demonstrations are meant to complement the explanations of concept demonstration involves a stick figure human on the earth's surface, which can be interactively moved as the rotation of the planet is mov phases change relative to time and direction... Full description. See reviews, teaching tips, related resources, etc.'

Figure 3-5. Results are ordered by relevance to query, terms used in the query are highlighted

The results are ordered by what is most relevant to the query that is submitted. Resources from the DLESE Reviewed Collection are placed high on the ranked list in the result set. When version 3.0 is deployed the ordering of the retrieved result set have to take spatial similarity into account also.

DLESE has tested a beta version of their search engine that supports geospatial searches. It uses the Alexandria Digital Library Gazetteer for placename mappings. It also uses a natural language IR index to support placename searches. For geospatial searches their current emphasis is on bounding box search capabilities rather than polygons.

The beta demonstration system allows the user to perform informal specification of location through placenames or textual searching. Usability analysis done by Lian Liu at the University of Colorado, suggested that users prefer to search by name or natural language text, rather than by selecting a region interactively on a map. It was found that users were most interested in using maps to understand search results, rather than to formulate queries. This is why the beta version only support informal specification of location. A “View Map” button is displayed next to the resource title for each retrieved resource that have geospatial information. If the “View Map” button is clicked then the search result is plotted on a map displayed to the user. An icon on the map indicate where the search results reside geospatially. Users can then zoom in or out on the map or click on an icon to go to the full resource description.

Gazetteer service:

The ADL Gazetteer and a natural language IR index is used to support placename searching in the beta version of the new search engine for DLESE.

Data storage:

Metadata records are the content of DLESE. DLESE does not hold the physical files of resources, only metadata records are held. Therefore, metadata is a key structural component of the library. Technically, DLESE metadata is kept in the form of structured digital records in the eXtensible Markup Language (XML).

Since DLESE provides access to a broad spectrum of resources, different metadata frameworks have been evolved to support characteristics of the resources being described. The current metadata framework used for discovery in DLESE is the ADN (ADEPT/DLESE/NASA) framework. This framework is strongly typed and XML based, it is also the most interesting of the metadata frameworks from a georeferencing perspective (because of the coverage metadata).

The ADN (ADEPT/DLESE/NASA) metadata framework is used to describe resources typically used in learning environments (e.g. classroom activities, lesson plans, modules, visualizations, some datasets) for discovery by the Earth system education community. The coverage (spatial and temporal) metadata is part of the optional metadata (that can be registered for each resource) for the ADN framework. The ADN metadata framework makes it possible to describe geospatial coverages with longitude and latitude coordinates. This framework is not limited to the Earth but can also describe other planetary bodies that a resource is about.

The ADN metadata framework captures the following geospatial characteristics:

- Bounding boxes and detailed geometries (points, polygons, and lines) are used to define footprints. Currently the emphasis is on bounding boxes for geospatial searching
- Vertical elevation can be described in the framework
- Planet or body of our solar system where the footprints apply
- Placenames with and without coordinates. Without coordinates, placenames can only be searched as keywords
- Events with and without coordinates. Keyword search is the only way to find events without coordinates
- Objects in space, location of objects not located within our solar system described using the convention of right ascension, declination, and epoch.
- Coordinate system using geographic longitude and latitude coordinates as a frame of reference for specifying the location of an object in “space”.
- Datum, which horizontal and vertical datums that are used as the base reference in the coordinate system. Datums apply to coordinate systems.
- Projection, what projection is used to represent the 3D celestial sphere as a 2D representation.
- Longitudes and latitudes are in decimal degrees with west longitudes and south latitudes as negative values and east longitudes and north latitude as positive values.

Applications:

End users can use DLESE for easy access to quality teaching and learning resources that can be used in education. DLESE also supplies services to help users effectively create, use and evaluate digital learning resources as well as supplying interfaces and tools to aid in the use of resources.

The community is a core part of DLESE since it fosters interaction, collaboration, and sharing. DLESE is a user-friendly digital library, it is a digital library made by a community for a community that emphasizes that users are key part of the system.

It is possible to find resources in DLESE both through a very user-friendly browsing search method, and through standard searching for information by submitting a query of keywords. The

results sets retrieved contain highly relevant information (for educational use). DLESE doesn't host the resources, but the resources are easily accessible through links to the them.

Only informal method for specification of location is supported, visual maps will be available to understand search results but not for specifying query regions.

3. 2. 7 ADL (Alexandria Digital Library)

Purpose:

“ADEPT aims at providing the structure and tools for a global integration of information currently stored in a diversity of geo-referenced databases around the world, into a world-wide spanning set of independent collections, accessible from a common point of access with a common interface. Thus making all information uniformly available, yet preserving the local independency, heterogeneity and authority of information storage and structure.”[52]

What was earlier called the ADEPT (Alexandria Digital Earth ProtoType) architecture is now known as ADL (Alexandria Digital Library). Today the ADEPT project has evolved and is involved in the realm of Virtual Learning Environments (VLE). The ADEPT middleware along with an operational ADL node is used to create the ADEPT VLE.

According to [53] the two main goals for ADL are:

- Make it easy for new collections to be added to the library.
- Allow users of the library to issue a single query against multiple heterogeneous collections.

The ADL architecture is made to accept various existing collection metadata schemes to fulfill the first goal. Heterogeneous collections at ADL nodes must appear to the user to support identical searchable metadata, this will fulfill the second goal according to the ADL developers.

Distributed:

The ADL middleware is made to be a distributed system. It supports distributed searches, and nodes can have collections added to them from all over the world. There can be multiple nodes in the system, and each node can have local collections as well as remote collections. It is possible to add remote collections from other nodes to a local node through the RMI (Remote Method Invocation) support in the middleware. It is also possible to share local collections (making them available) for other nodes. The searching at each node is limited to the collections that are available at that node.

The ADL middleware is dependent on Java, Ant, and Tomcat to be installed on the server. The ADL middleware and the other software it is dependent on is constructed so they can run on a lot of different software (Windows, Linux, Unix, Solaris, and so on) and hardware platforms.

Geospatial search:

All three methods of defining geospatial locations are supported in the middleware. Informal method can be invoked using placenames, formal method is supported via textboxes where coordinates can be entered, and the interactive map can be used as a visual method for specifying location. When drawing a bounding box in the interactive map (visual method), the coordinates in the textboxes are automatically updated to reflect the current bounding box selection in the interactive map.

The ADL system makes it possible to search for fundamentally different data types in the same way because all document types are mapped to the bucket framework. The bucket framework also makes use of strong datatyping so special georeferenced ranking is made possible and strong datatyping should be used for coordinate based searching.

“A thesaurus is used to solve semantic ambiguities, for example many words for the same-phenomenon.”[54]

Spatial ranking was added as a feature to the MIL ADL node on the 7th of October 2004.

Gazetteer service:

The middleware uses the ADL gazetteer-service for translation between placenames and footprints (longitude/latitude coordinates). The gazetteer is used in ADL to achieve a less coordinate dependent client. The ADL gazetteer catalogues more than 5.9 million placenames, all of the names are associated with one or more terms from the ADL Feature Type Thesaurus and portions of the names are associated with either the gazetteer type terms from the U.S. Geological Survey or from the U.S. National Geospatial-Intelligence Agency (formerly NIMA)[55].

Data storage:

ADL defines a general architecture that makes it possible to add different types of databases and collections to the middleware. This is because of the innovative ADL bucket framework that makes it possible to add collections no matter what metadata standard or format they use. The ADL middleware also support a wide variety of databases so what type of database a collection is stored in is not a hindrance. The ADL middleware does not normally need to change the content or metadata of the database in order to make use of it. ADL is also constructed to be scalable so lots of collections can be added to the middleware.

Applications:

ADL makes it possible to interpret problems and their solutions in spatial terms, answering questions like “what information exists about this place” by simply drawing a bounding box around the spatial region and hit the “Search” button. It is also possible to add temporal constraints to the query so questions like, “what happened there then?” can be answered. Search for placenames are also supported. ADL offers ways to explore the concept of spatial literacy to the users.

ADL also supports standard digital library features like keyword search and retrieval. All three methods for specifying geographic location are supported in the ADL system. It is also possible to select what object-type of information to search for (images, software packages, textual works,

cartographic works, or data sets), and ranking scheme used to rank the retrieved set can also be selected (by date (newest first, oldest first) or by similarity to the query region).

ADL has a higher threshold for first-time users than Go-Geo! or DLESE. There is no simple search option when searching, all searching is done in an “advanced” search page. Very much information is presented all at once to the user, so users will take some time to find their way around the system. Scrolling down the search-option menu to find the search button at the bottom is a little awkward. Very much functionality is offered at once to the users, this can be mind boggling for new users that try to orientate themselves in a new search system. Once familiar with the system, it is easy to use and offers great functionality. Since ADL offers features and functionality that very few other systems can offer, users will most likely find new features and functionality they have never encountered before.

3. 2. 8 Brief overview

Following is a brief listing and comparison of georeferencing capabilities found in the different systems described previously (the three methods for specifying geographic location, gazetteers used, metadata formats/standards/frameworks supported, what type of databases the collections are stored in, and what geographic coverage the system currently supports).

Table 3-1: Georeferencing capabilities

Name:	Norges glasset	Primus	EDINA: Go-Geo	Galleri Nor	DLESE (beta)	ADL
Visually (interactive map)	Yes	No	Yes	No	No*	Yes
Informally (placenames)	Yes	Yes	Yes	Yes	Yes	Yes
Formally (longitude/latitude)	No**	No	Yes	No	No	Yes
Gazetteer	SSR	No	GeoXwalk	No	ADL + IR index	ADL
Metadata	No***	Extension of “Feltkatalogen”	FGDC or ISO 19115	“Feltkatalogen”	Any (ADN framework)	Any (bucket framework)
Supported types of databases for collections	No***	Oracle	NA	Oracle	Link to remote resources	Relational databases (JDBC)
Geographic coverage	Norway	Norway	United Kingdom	Norway	Entire solar system	Entire world

* Interactive map is only used to understand results, not to define query regions

** Can get mouse-over coordinates, but can’t define query region by filling in any coordinates

*** Online digital map-service for displaying maps, not a digital library with collections

NA - Information Not Available

As a minimum, georeferenced digital libraries must make it possible for users to evaluate georeferenced resources for query relevance. Go-Geo!, Norgesglaset, DLESE (beta version), and ADL all have GIS functionality that allow the user to evaluate a query for spatial relevance. Primus and Galleri Nor however, have no GIS capabilities that allows users to evaluate the spatial relevance of queries. Primus and Galleri Nor contain information objects that are georeferenced, but the systems do not provide a spatial orientation to the information objects in terms of discovery, browsing, viewing, or access. Norgesglaset has no collections because it is an online digital map service and not a georeferenced digital library.

Go-Geo!, DLESE (beta version), and ADL are all georeferenced digital libraries, but are they fully fledged georeferenced digital libraries? They all have gazetteer services to support translation from placenames to footprints, and they all provide spatial context through interactive maps. Longitude/latitude coordinates are essential in the retrieval of the georeferenced information objects in these systems.

The DLESE beta version does not support the visual or formal methods for specifying geographic location, the decision to only use the informal method is based on the user-tests for usability. However that means that DLESE does not fulfil all the criteria for a fully fledged georeferenced digital library. DLESE uses strong data typing in the ADN metadata framework, and the architecture used scales well since DLESE only hosts metadata descriptions and links to the resource collections that are autonomous. Since DLESE makes their own ADN metadata description of the information resources, it is possible to add any type of collection to DLESE. The ADN framework is not limited to the planet Earth, but can be used to georeference anywhere in this solar system. It'll be interesting to investigate how spatial ranking will be done when the new geospatial search interface will be launched. Tool development for use of georeferenced information objects may also need to be developed. DLESE looks like a very promising project at the moment.

Go-Geo! supports all the three methods for specifying geographic location. Geographic context is displayed as placenames and boundaries on the interactive map. The geoXwalk is the gazetteer service that is used by the system. Go-Geo! is a United Kingdom web portal and is therefore currently limited to the geographic scope of the UK. The FGDC or ISO 19115 metadata standards are supported currently, so collections must be described in these metadata standards. Scalability becomes less important when Go-Geo! is by choice limited to the UK geographic region. Nevertheless Go-Geo! has good scalability according to [47], it can fit approximately 1 million indexed metadata records on a single modern personal computer (currently the Z39.50 servers provided by the UK Geographic community is only in the order of eight thousand records). Go-Geo! supports spatial and lexical ranking.

ADL supports all three methods for specifying geographic location. ADL also implements spatial ranking of the retrieved datasets based on spatial similarity between the query region and the information objects footprint. Strong datatyping is also used in ADL, and geographic context is supplied through the interactive map. The scalability of ADL is good, and it supports any metadata scheme through the use of the bucket framework. ADL can supply the mechanisms necessary to operate on special georeferenced information objects (downloading large files as multipart

files). The ADL gazetteer is used for translation from placename to footprints in ADL. ADL fulfills all the requirements of a fully fledged georeferenced digital library.

4 ADL

4.1 ADL (Alexandria Digital Library)

ADL is a georeferenced distributed digital library architecture that is being developed at the University of California, Santa Barbara (UCSB). The architecture is a framework for building digital libraries containing georeferenced information. Distributed means that the components of the library may be spread across the Internet, as well as coexisting on a single computer. ADL is an architecture for managing, querying, and presenting geospatial information in a uniform way.

4.1.1 ADL Overview

The ADL description is based on information from [37].

The ADL project was originally motivated to create a digital library that could both reproduce and extend the functionality and content of traditional research map libraries. Maps are one of the earliest forms of sharable human knowledge. Maybe they originated as drawings in mud or sand, but by the 16th century they had evolved into sophisticated, called representations of the surface of the Earth.

Map libraries are distinct from other libraries in a number of important ways. Map libraries are **more specialized** than a general library, and because of that not every library has or can afford a large map collection. There are issues of **preservation and storage** of maps, images, and globes due to their size and cumbersome physical form. It is **notoriously difficult to catalog** maps, images, and globes in the traditional author/title/subject paradigm of classifications. The most common way for a user to search is looking for a map of *somewhere*, and thus geographic coverage is the most obvious basis for search and retrieval of maps and related objects.

The ADL project began in 1994 as an attempt to address these three problems.

“...users would be able to use the library remotely, leveraging the library’s investment by extending access to it globally; digital storage would resolve issues of preservation and the management of physical media; and an automated catalog would be capable of finding information by geographic location.”[37]

When the Web arrived and was popularized it was recognized that it was a far more effective vehicle for access, search, and disseminations so the ADL project rapidly changed to a HTTP-based approach. By distributing the prototype of ADL to potential users the developers got a stream of useful feedback and guidance on the appropriate design for the user interface. They discovered that it is not enough to present users with a map and expect them to identify an area of interest on it. Context such as scale and level of detail is of great importance for users when dealing with maps.

One of the most important discoveries of the ADL project team is that the:

“...concept of search based on geographic location could be generalized to apply not only to maps and images, but to any information objects that possessed geographic references or *footprints*, by being associated with points or areas on the surface of the Earth.”[37]

“The project coined the term *geolibrary*, for a library containing georeferenced objects and with a search mechanism based on geographic location as the primary search key.”[37]

The most prominent of the concepts pioneered by ADL is the ability to search by geographic location as represented by longitude and latitude coordinates.

“Geographic space is continuous and multidimensional, and an infinite number of possible locations can be referenced, making it impossible to create a discrete, unidimensional search key analogous to author, title, or subject. Complex relationships such as containment, overlap, and adjacency exist between locations, and while they can be deduced from formal, coordinate specifications they are mostly invisible in informal, placename-based specifications. A geolibrary can only exist in a digital world, therefore, and it remains one of the most powerful concepts to have come out of digital library research.”[37]

The ADL project developed a fully-fledged middleware that supported HTTP interfaces to multiple clients, and connections to multiple catalog databases. In the recent years the ADL middleware has been constructed to be interoperable with other collections and catalogs, to move from a single monolithic collections (the UCSB collection) to a complex of smaller, more homogeneous and distributed collections. The distributed collections can be accessed and searched transparently in a globally effective digital library. This decentralization has allowed other sites to participate in the collection-building process.

ADL was the first Digital Library project (1994-1999), and came to be the umbrella project (1999-2004). The ADEPT (Alexandria Digital Earth ProtoType) project is about integrated learning environments. A digital learning environment in ADEPT consists of an ADEPT component and an operational ADL component. The ADEPT project funding is about to expire.

The operational ADL is part of the library at UCSB, and has a permanent staff. The development of the ADL middleware will continue with a couple of follow on projects, one in the Bren School, and one in the library.

4. 1. 2 Georeferenced information

The ADL architecture is developed to manage, query, and present georeferenced information. An item in an ADL collection should have one or more footprints associated with it if possible. It's also possible to add collections without any georeferenced information to an ADL node, but this should not be done due to ADL policy:

“There is no requirement that collection servers support every bucket, but all collection servers are required (as a matter of ADL policy, not architecture) to support the *adl:geographic location bucket*.”[53]

So all collections added to a node have to implement the bucket for geographic location as a matter of policy. It makes sense that collections in a georeferenced digital library have georeferenced content. The bucket framework will be described later in this chapter.

4. 1. 3 Heterogeneous collections

It is possible to add heterogeneous collections to an ADL node, but what does heterogeneous mean?

“Heterogeneous means that a library may contain multiple types of digital information, including non-traditional items such as remotely-sensed imagery, executable models, and multimedia instructional materials.”[42]

The ADL architecture’s ability to contain heterogeneous collections and perform homogeneous searching, retrieval, and presentation of the collections is directly linked to the bucket framework of ADL.

4. 1. 4 The Bucket framework

First an overview of buckets, then we explore:

- The nine standard buckets
- Operators for searching
- Buckets in the ADL architecture

A bucket is an abstract, strongly typed metadata category that have defined semantics for searching. Source metadata is mapped to the buckets via tuples. Source metadata is the raw metadata that is unprocessed and untransformed from its native state as it is provided in the collection. The reason the term “metadata category” is used instead of “metadata” field is to emphasize that the ADL framework is not a content standard for metadata[42].

Tuples in buckets indicate the metadata value, the source of the metadata, and its appropriateness for discovery. An example of a tuple is:

Originator = {(MARC 100 [Personal Name], “Mark Twain“, searchable)}

An item may map any number of typed (field, value) pairs to a bucket. Such a way of mapping metadata enables clients of a library to choose between operation at the high level of bucket description of metadata or operate at the native metadata level. It’s called “drilling down” when operating at native metadata level. The bucket framework with its tuples provides a uniform homogeneous view of the library’s heterogeneous collections.

*“The ultimate effect of the ADL bucket framework is that clients of the library can, without relying on any *a priori* knowledge or out-of-band agreements, successfully search arbitrary collections at a uniform, high level. Furthermore, the types of search provided are far more capable than just text-based search. At the same times, the richness and semantics of the underlying native metadata are entirely preserved, and are discoverable and exploitable by clients in an entirely regular way.”*[56]

The buckets are the most special feature of ADL. Each collection have to implement at least one bucket. The buckets define information it is valuable to search for, and the abstraction means that an information seeker doesn't have to know how the data is stored to make use of it. A collection doesn't have to support all the standard buckets, but it must at least support one (geographic locations at least according to the policy). As many buckets as possible should be implemented for a collection to support as many search capabilities as possible for the collection.

The nine standard buckets

The following information is based on information from [53,57].

Table 4-1: Standard buckets

Name	Internal name	Type	Operators	Content
Geographic locations	adl:geographic-locations	spatial	contains, is-contained-in, overlaps	The item's spatial footprint, i.e., an approximation of the subset of the Earth's surface to which the item is relevant, expressed as any of several types of geometric regions defined in WGS84 longitude/latitude coordinates
Dates	adl:dates	temporal		The item's temporal footprint, i.e., the range of calendar dates to which the item is relevant
Types	adl:types	hierarchical	is-a	Terms drawn from the ADL Object Type Thesaurus identifying the meaning or content of the item
Formats	adl:formats			Terms drawn from the ADL Object Format Thesaurus identifying the form or representation of the item
Titles	adl:titles	textual	contains-all-words, contains-any-words, contains-phrase	The item's title. This bucket is a subset of the "Subject-related-text" bucket
Originators	adl:originators			Names of entities related to the origination of the item (authors, publishers, distributors, etc.)
Assigned terms	adl:assigned-terms			Subject-related terms from controlled vocabularies. This bucket is a subset of the "Subject-related-text" bucket
Subject-related text	adl:subject-related-text			Text indicative of the subject of the item, not necessarily from controlled vocabularies. This bucket is a superset of the "Titles" and "Assigned terms" buckets
Identifiers	adl:identifiers	identification	matches	Item names and codes that serve as unique identifiers

ADL has defined a set of nine standard search buckets. The buckets are defined conventionally, not architecturally. Clients can assume that all collections support all of the nine buckets listed in table 4-1. The standard buckets define the content that can be searched in each bucket, and what operators to use when searching each bucket.

Additional buckets can be defined if needed, and not all nine buckets needs to be implemented. An ADL query will fail however, if a query constraint is used that corresponds to a non implemented bucket.

Operators for searching

The spatial and temporal bucket types support the following operators:

- **Contains:** The search region (spatial/temporal) defined is contained inside an information object's spatial/temporal description (footprint/time period).
- **Overlap:** The most inclusive operator for spatial and temporal searching. As long as a fraction or more of an information object's footprint/time period description match the query region (spatial/temporal) it is considered a match.
- **Is-contained-in:** The information object's footprint/time period is totally surrounded by the query region (spatial/temporal).

A more complete description of the operators is given in chapter 2.2.8.

The hierarchical bucket types support the **is-a** operator. The is-a operator is used to express hierarchical relationships. If the "image" type is chosen in the object type thesaurus drop-down menu, the search will return information objects that are typed as images. There is inheritance in the hierarchy so the search for "images" will be expanded so that it also includes the subclasses of the "image" type, such as "remote-sensing images".

Textual bucket types support these operators:

- **Contains-all-words:** All the terms in a query must be found in the information objects metadata in order to retrieve it as a match.
- **Contains-any-words:** If one or more terms from the query are found in the information object's metadata it will be retrieved as a match. This is the most inclusive textual search operator.
- **Contains-phrase:** If phrase searching is used, all the terms have to be found in the information object's metadata. The search terms also have to occur in the same order in an information object's metadata as in the phrase search in order for the information object to be retrieved as a match.

The **matches** operator is used in the identification type bucket. Identifiers should be unique and characters like: spaces, colons, semi-colons, slashes, and multiple dashes should not be used in identifiers. The matches operator retrieves hits that are equivalent with the search term.

Buckets in the ADL architecture

The buckets are a key part of the ADL architecture, and they provide a uniform interface to one or more collections. Through that interface it is possible to retrieve the items that are stored in the collections. It is through the use of buckets it is possible to search the collections. Which buckets

are implemented for a collection to delimit the search capabilities that can be offered in the client user interface. The buckets play three specific roles in the ADL architecture, they are used to: describe items, search for items, and characterize collections.

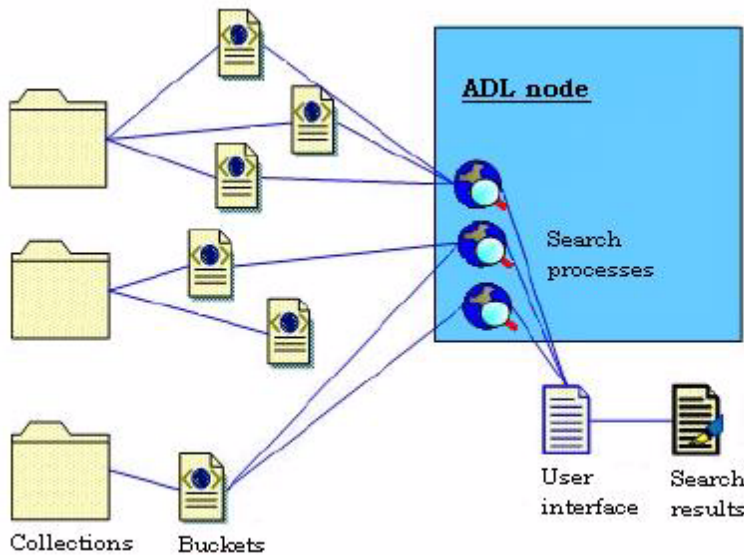


Figure 4-1. Logical overview of the ADL architecture[58]

When looking at the bigger picture, ADL buckets are really just strongly typed search indexes for categories of information. The strong data-typing allows ADL to express geographic and temporal searching over a variety of information. All searching in ADL uses the buckets, so there has to be a bucket implemented in order to search a collection.

Which buckets that are feasible to implement for a collection, depends on the structure of the data in the collection, what kind of metadata that exists for the collection, in addition to what user groups that will make use of the collection. There are no limitation regarding how many nodes a collection can have, or how many collections a node can have.

4.1.5 Technology

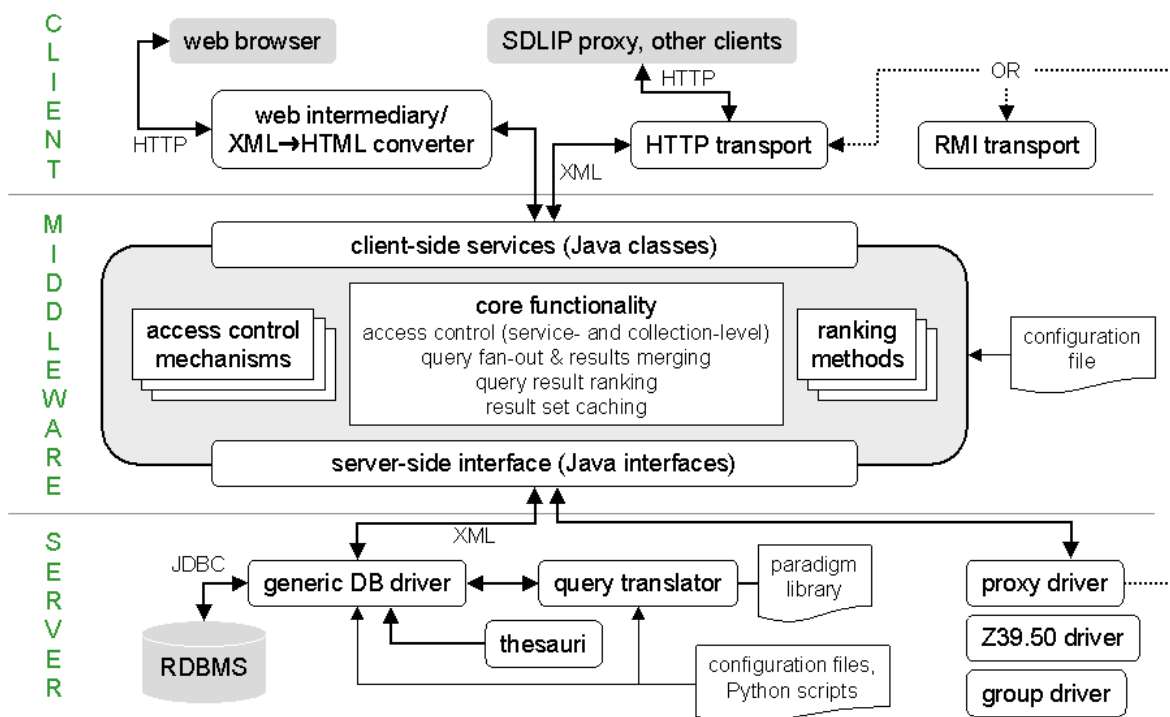


Figure 4-2. The 3-tier architecture, listing the standards and technologies used[59].

ADL is available free of charge, and is built through the use of open standards. ADL has been programmed using Java, Java servlets, and JSP (Java Server Pages). JSP and servlets are extensions of the original Java definition, and these two technologies are mostly used for java implemented solutions on the Internet. Servlets and JSP have access to the Java API (Application Programming Interface) and the JDBC classes. Services in ADL are implemented through servlets, and JSP is used to make the user interface web pages.

One of the great advantages of the Java technologies is that they can be compiled once and then run on any operating system (OS) that has java runtime environment support. ADL is also dependent on a servlet engine (Tomcat), and Ant. Tomcat and Ant can be run on various operating systems as well. Modularity is one of the key aspects of object oriented programming, and since Java based technologies are used modularity should be easy to implement. The Remote Method Invocation (RMI) interfaces are also Java based technology that ADL makes use of for making their digital library distributed. It is possible to add remote collections from other nodes to the local node through the use of RMI. Since ADL is implemented using Java based solutions, no delimits are put on choice of OS.

XML (eXtensible Markup Language) is used in ADL in two ways. Some of the configuration files in ADL are written in XML, and ADL uses XML as an information carrying format when queries and results are encoded in XML. Users enter keywords to search for, the search parameters are then sent as XML to the middleware, and when the results are retrieved they are sent as XML from the middleware to the client interface. All operating systems can read and write text files,

thus they can read XML files. Since almost any kind of information can be encoded using XML, and because of the readability, it is a good format for storing and exchanging information.

Another standard ADL makes use of is SQL (Structured Query Language). Spatial searching is supported through OpenGIS simple features for SQL. It is possible to draw a four-point bounding box to define a spatial footprint that puts a spatial constraint on searching. The ADL middleware supports “SQL exact” and “SQL sub-match” type text fields.

There are configuration files for both the middleware tier, and for the server tier in the diagram. The configuration files in the server tier are used to configure the collections of the node. An ADL node in this setting is the middleware running on a webserver, each node can include several collections that each can reside on different computers anywhere in the world. In theory it should be simple to add new collections or update the collection configuration of a node.

Figure 4-2 shows that clients that want to interact with the ADL middleware have to be able to send and receive HTTP messages. A web browser that supports XML and HTTP have all the features needed to be used as a client for an ADL node.

4.1.6 Thesaurus

*“What’s another word for thesaurus?”
- Steven Wright*

ADL makes use of controlled thesauri and hierarchical term lists (e.g. Format and Type thesauri) to improve the possibility of finding appropriate terms, so each item only needs to be tagged with one term. Searching for the “image” type, the middleware will automatically expand the search to include “remote-sensing images” also.

Each term in a thesaurus is a word or phrase for a conceptual category. Terms can be divided into two broad groups: preferred terms and non-preferred terms. Following is some characteristics of thesauri:

- Preferred terms: Words that should be used in the description of information objects.
- Nonpreferred terms: Words that should **not** be used for descriptions, the preferred term that should be used instead is usually listed.
- Broader term (BT): A term that is the super-class for the current term. Aeroplane is a BT for a fighter for example.
- Narrower term (NT): A term that is the sub-class of the current term. A fighter is the NT for an aeroplane. Broader and narrower terms must be acyclic, if A is BT of B then B must be NT for A.
- Related terms (RT): Relations between terms that are not covered by BT and NT can be for example: synonym or antonym. RT can be used to express these relationships between terms.
- Used for (UF) and Use-instead (USE): USE is used to map a non-preferred term to a preferred term, the UF lists what non-preferred term(s) a preferred term can be used instead of.
- Comments: Comments can be added to each term in a thesaurus to clarify the use of the term.

A thesaurus is used to control the vocabulary that can be used for making the descriptions of a collection’s information objects. This improves both search precision and recall, because queries

and descriptions use the same vocabulary. In the ADL user interface it is possible to select what type (image, map, software packages, textual) of information to look for by using the Object Type Thesaurus, and to define what format should be (for images formats available are: JPEG, TIFF, BIL, DOQQ, or ERDAS IMAGINE) retrieved by using the Object Format Thesaurus.

The ADL Thesaurus protocol is a protocol for accessing various thesauri. The protocol is lightweight, stateless, XML-, and HTTP-based. The protocol supports downloading, querying, and navigating thesauri. The protocol allow programmatic clients to access and utilize existing thesauri. The protocol does not support creation, maintenance, mapping between, or sharing of thesauri. More information about the protocol can be found at [60].

4. 1. 7 ADL Gazetteer

A gazetteer service is used to translate between placenames and footprints, and in the ADL middleware the ADL gazetteer is used for this translation. The ADL Gazetteer consists of more than a protocol and the possibilities of translating between footprints and placenames, but these are the most interesting features in this setting.

The ADL Feature Type Thesaurus (FTT) can be used in the ADL Gazetteer for description of feature types. FTT is a set of terms for categories of geographic places; terms to indicate the nature of a place. More about the FTT can be found at [61].

The ADL Gazetteer Content Standard (GCS) [62] is a framework for recording descriptions of named geographic places, including spatial locations, toponyms, and source referencing for pieces of information about each place. The version 3.2 of the GCS supports recording of historical data about places, and it is designed to support international and thus multilingual applications. It can be considered an archival structure. The treatment of temporal descriptions is one of the most interesting aspects of the new version of the content standard.

There are three temporal statuses that are required: current, former, or proposed. The temporal description can be used to describe: the feature, placenames, spatial location, classification (typing), relationships and data. It is also possible to specify date ranges with beginning and end dates, or associations with named time periods like the Middle Ages.

The GCS deal with the following sections[62]:

- Names - details of origin, language, and use
- Classification - typing according to a reference scheme (ADL FTT can be used)
- Codes - for example postal codes associated with the place
- Spatial location - footprints can be represented by a bounding box and detailed geometries
- Street Address
- Relationships - to other named places
- Data - for example population, and elevation
- Description
- Links - external resources about the feature
- Other - supplemental note or metadata

A feature's name, classification, and spatial location along with some supporting information are required by the GCS for each gazetteer entry. For any gazetteer entry a selection of the non-required elements can also be added.

The ADL GCS and the ADL Gazetteer Protocol are companions, but the protocol and GCS are independent structures. The protocol is a protocol for accessing general purpose gazetteer services and it provides a standard XML based query and response structure for peer-to-peer querying of distributed gazetteers. More information about the ADL Gazetteer can be found at [55].

4. 1. 8 Webclient

The webclient that comes shipped with the middleware is the standard user interface for the middleware server. The services of the middleware server receive XML queries and returns the results in XML format as well. A web browser is all that is needed to access the webclient. The webclient is a JSP page with an interactive map. ESRI ArcIMS mapserver is used because it has the best tools for use in JSP pages. The interactive map allows users to interact with an onscreen map to define a query region.

It is possible to make a personalized client (make a special client for the local library and its special user-groups), or to make use of the middleware as a geospatial service in a larger distributed digital library architecture (the DLESE beta used the middleware server as a service for geospatial services). The webclient along with the middleware can be used as a georeferenced distributed digital library, or the middleware server can be used as a service in a larger georeferenced distributed digital library.

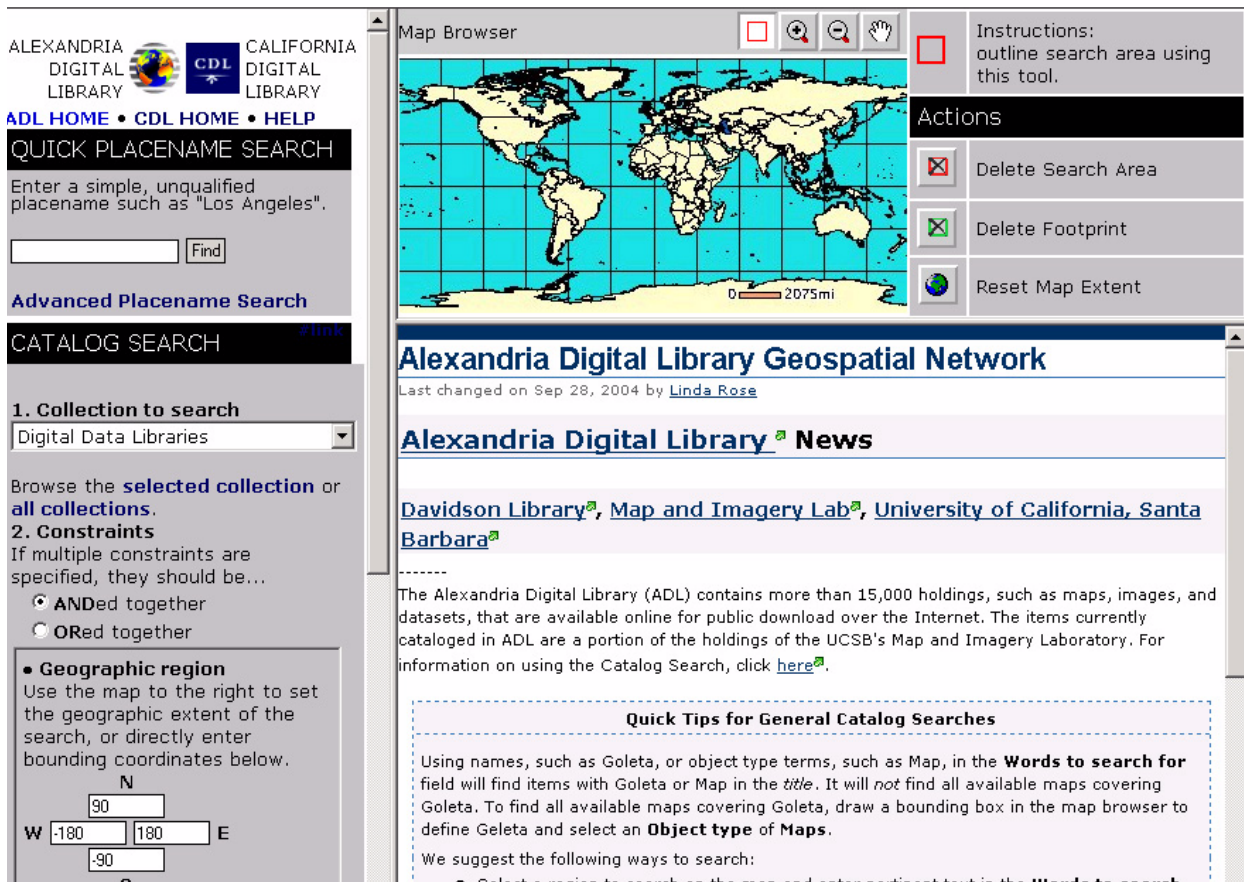


Figure 4-3. The ADL webclient[63]

Some data types like images (photographs, maps) and text are easy to display in the webclient, while data types like video or very large files can be difficult to present properly. It is possible to download large files and data types the web browser can't display properly.

4. 1. 9 Middleware services

The middleware supports ten core services for interaction between the middleware and the middleware collections. These services can be used for testing and administration of a local node. Some of the services are used through the webclient to support querying and the retrieval of results.

A user interface that can be used to access the services that the middleware offers is the "Middleware Test Forms" JSP web page that is shipped with the middleware. An example of such a web page can be found at [64]. If a default installation of a node is done, then no authorization is needed to access the middleware services through the test page for that node.

The middleware supplies these services[64]:

- **configuration() : properties** - Returns middleware configuration parameters as an XML document. Lists for example available collections at the node and supported ranking schemes.

- ***status()* : HTML document** - Returns an HTML document describing the status of the middleware. This is the place to look if errors occur in a query. Lists result set information.
- ***collection(collection-name) : metadata*** - Returns the collection-level metadata for a collection as an XML document. Input one of the collection names returned from the configuration service to list that collection's metadata.
- ***query(query) : query-id*** - Asynchronously queries one or more collections for items that match one or more constraints. Ordinarily the return is an integer that identifies both the running query and the corresponding result set.
- ***cancel(query-id) : void*** - If the query identifier (integer returned from the query service) of a currently running query is input, then it will cancel that query.
- ***results(query-id) : result-set-reference*** - Used to access a result set, can also be used to delete result sets.
- ***metadata(view, collection-name, itemID) : metadata*** - Returns a view of the metadata for a collection item. The view may be one of the three standard ADL metadata views (bucket, browse, access) or any other view supported by the item.
- ***reference(base-url, remote-collection-name [, localname]) : void*** - Creates a collection at the local node that is a proxy for a remote collection at a remote ADL node. The remote collection is specified by an RMI or HTTP base URL and a remote collection name. If a *localname* is supplied, then that name will be used for the collection at the local node.
- ***unreference(collection-name) : void*** - Removes the proxy collection specified from the local node.
- ***unload(collection-name) : void*** - Unloads the drivers for the specified collection. Any subsequent reference to the collection will cause the collection's configuration files to be reloaded.

The three next services are not really middleware services, but can be used for testing and administration:

- ***collection_availability(availability-option, collection-name) : void*** - Disables or enables a collection. If a collection is disabled it is not shown in the webclient user interface if a user access the node.
- ***rmi_control(rmi-option) : void*** - Enables (rebind) or disables (unbind) the middleware's RMI interface.
- ***bucket99_status()* : HTML document** - Returns an HTML document describing the status of all currently-loaded Bucket99 drivers.

4.2 The ADL distributed architecture

The Alexandria middleware server is a peer-to-peer based architecture for searching geospatial data, it is based on the Java and the Tomcat technologies.

4.2.1 Collections

It is possible to install a node of the middleware without any collections, however a digital library without a collection is not really a digital library since collections are part of both the library concept as well as the digital library concept. Installing the middleware without adding collections to the node is therefore pretty meaningless.

So what **requirements** are there for adding a collection to an ADL node? There is a policy in ADL that every collection is required to support the `adl:geographic` location bucket, however since it is a policy and not defined in the architecture it is possible to add a collection that doesn't implement that bucket. It is required that the metadata of a collection have to be digitally available, if not it will not be possible to map buckets to the metadata. The information objects/documents of the collection can be either analog or digital, but preferably both the collections metadata and the information objects are digitally available. A collections item-level metadata must be mapped to at least one bucket in order to be able to perform searching in that collection. Another requirement is that each information object in a collection must have a unique identifier. It is preferred but not required that collections are available in JDBC compliant databases, but it is possible to use object-oriented databases as well.

There are no limitations to what data-type the data in collections can contain, it is possible to add collections of movies. A web browser doesn't support all data-types so ADL can make tools available to support downloads of data-types that are hard to display in web browsers.

A local node can make available both local collections as well as remote collections to information seekers that use the node. A local collection in this setting is a collection that can be located anywhere in the world, but the local node accesses the collection directly. Remote collections are collection that reside on other ADL nodes that is possible to make available at a local node through using a RMI (Remote Method Invocation) connection to the other node where the collection resides.

Collection drivers

The most common format for storage of collections today is through the use of relational databases. Object-oriented databases as well as some other formats also exist, but relational databases are the most common.

“At present, ADL works best with information stored in databases so we work best with sites that track their information in a database. Many scientific data formats have header information that contains information about who created the data, when was it created, edited, and processed, where was it created. If you can read these header files, and populate a database with the information, you can use ADL.”[65]

To add a collection to the ADL middleware a collection driver for the collection to be added has to be implemented, as well as various other configuration files for each collection have to be written.

To reduce the efforts needed to implement a collection driver the **Bucket99 driver** was developed. This driver is made for relational databases. The “`bucket99.conf`” file has to be configured to make use of the Bucket99 driver. In this file the database location is stated, as well as authorization information needed for the database access. Some queries also have to be added to the configuration file in order for the correct information to be returned. The bucket99 driver was made so it would be easier to make a collection driver for relational databases, so the three drivers listed below don't have to be implemented for each collection.

Collection drivers for non relational databases can be implemented through three interfaces:

- CollectionDriver
- MetadataDriver
- QueryDriver

The Bucket99 driver is generally the most useful, since most collections are stored in relational databases.

4. 2. 2 Adding a local collection to a local node

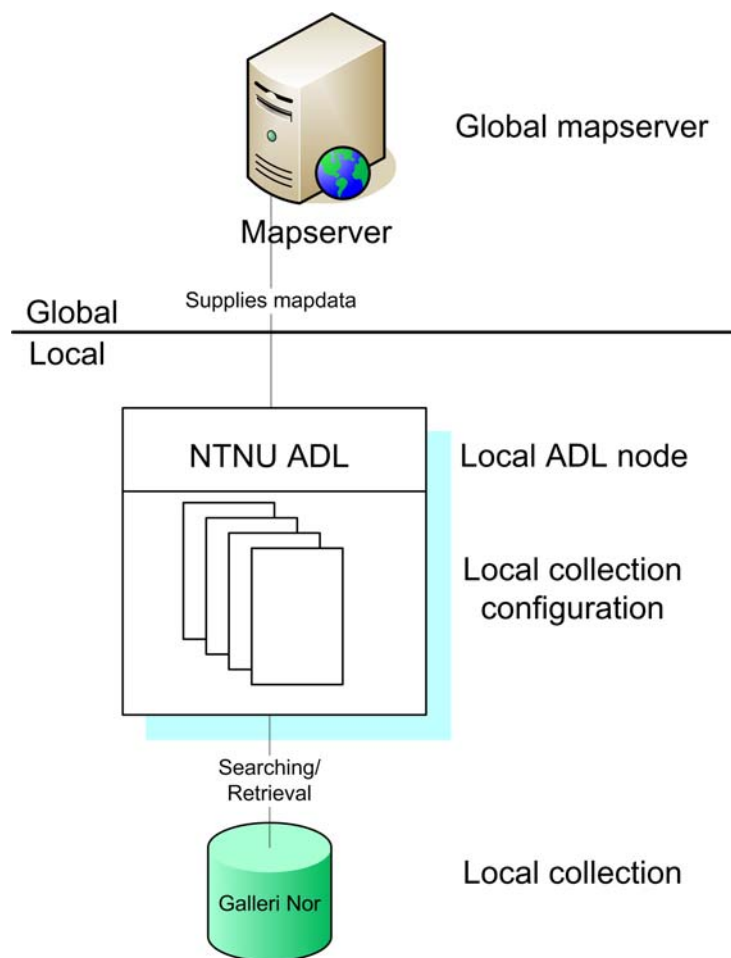


Figure 4-4. Adding a local collection to a local node

The mapserver in figure 4-4 is an ESRI ArcIMS mapserver. The mapserver delivers dynamic maps, GIS data, and services to the interactive map that can be found in all the nodes in the ADL's georeferenced distributed digital library system. The mapserver does not need to be global, but currently the clients are shipped with a configuration that uses the ESRI ArcIMS mapserver at the UCSB campus. ESRI ArcIMS is used because it has the best tools for use in JSP pages, it also has a webservice based service.

The local node contains local collections. For each collection in a node there is a directory (for example /gallerinor) that contains the configuration files for that specific collection.

“To construct a searchable collection, often you do not need to create additional tables. You will need to create configuration files and construct queries and report templates to produce the standardized ADL views.”[65]

4. 2. 3 Making a local collection available to the distributed network

Sharing of information is currently done by a Java RMI interface.

There are two levels of possible sharing:

- Use other existing ADL collections that is shared from other nodes
- Share the local collections with the rest of the distributed network

If the local collections are to be shared with other ADL nodes, then the RMI (Remote Method Invocation) server at the local node must be enabled. The RMI server is not enabled by default when the middleware is shipped. This is done because the RMI server controls will throw error messages that can be confusing when initially setting up a node. The “rmi-server.conf” file is the configuration file for setting up the RMI server.

Once the RMI server is enabled the developers at UCSB must be alerted by e-mail that new collections are made available to the distributed network. In the future there might be an ADL discovery service that will query the RMI server at the local nodes to get an overview of what collections are currently available to the distributed network.

When the RMI server is enabled all the local collections at the local node are then suddenly made available to the entire distributed network. Currently there is no configuration file to list what collections to make available to the entire network, and what collections should remain private at the local node.

Sharing needs to be used carefully. It has the potential to cause a loop, because currently there are no way of knowing whether a collection is shared from another server or it is an original collection from the configuration.

4. 2. 4 Making use of collections from other ADL nodes

There are two ways to make use of collections from other ADL nodes:

- Configuring an RMI collection
- Dynamically accessing the collection

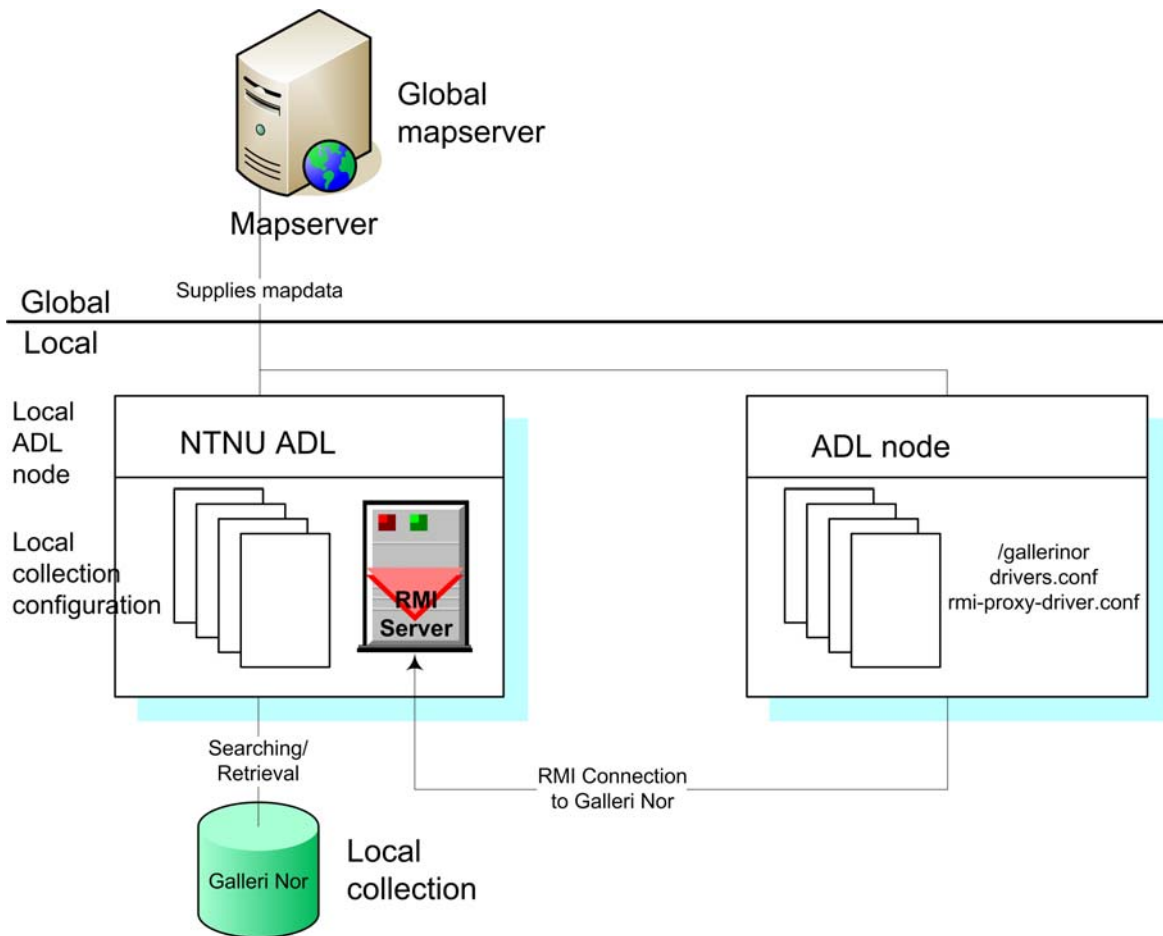


Figure 4-5. Another ADL node have configured a RMI collection from GalleriNor.

To make a collection from another ADL node available for searching and retrieval at a local node it is possible to **configure an RMI collection** in the local node for that collection. Make a new directory for the RMI collection in the local node (for example: /gallerinor), and then add the files “drivers.conf” and “rmi-proxy-driver.conf” to that directory and implement them for the chosen collection. The RMI service accesses the remote interfaces. Only the RMI port needs to be known. Based on experience, connections between the machines need to be allowed if there is a set of firewall rules.

It is also possible to search and retrieve information by **dynamically** connecting to the shared RMI collections. The reference servlet/service on the “*test-form.jsp*” web page creates a collection that is a proxy to a remote collection. The remote collection is specified by an RMI or HTTP base URL and a remote collection name. The collection is referred to locally as *localname* if supplied.

- RMI server: RMI or HTTP base url to the server that contains the collection
- Collection name: The remote collection name can be different from the local collection name. This means that a collection can have a different name in the local node than it has in the original node.

- Local collection name: What the collection is referred to locally on the local node that contains it.

4. 2. 5 Glimpse of the future

In the future there will be an ADL discovery service that will contain a list of RMI collections that are available to the distributed network. When signing up to the discovery service it will query the local RMI server at set intervals to get an overview of what collections are made available to the network.

When a user wants to add an RMI collection to a local node he/she only has to select an RMI collection from the list of available RMI collections, and download and save the “drivers.conf” and “rmi-proxy-driver.conf” files for that collection in a directory (/collections/[nameOfRmiCollection]) on the local node to make the RMI collection available to the local node. No such global list exists today, but it will be maintained by the global discovery service when it becomes available. Supposedly the configuration files for the RMI collections will be uploaded to the global list from the node that originally makes the collection available, so it can be maintained at one location. Versioning of the configuration files when changes occur and notifying the nodes in the distributed network in some way is also important. All other nodes that want to add a remote collection to their node should download the files needed from one place (the global list).

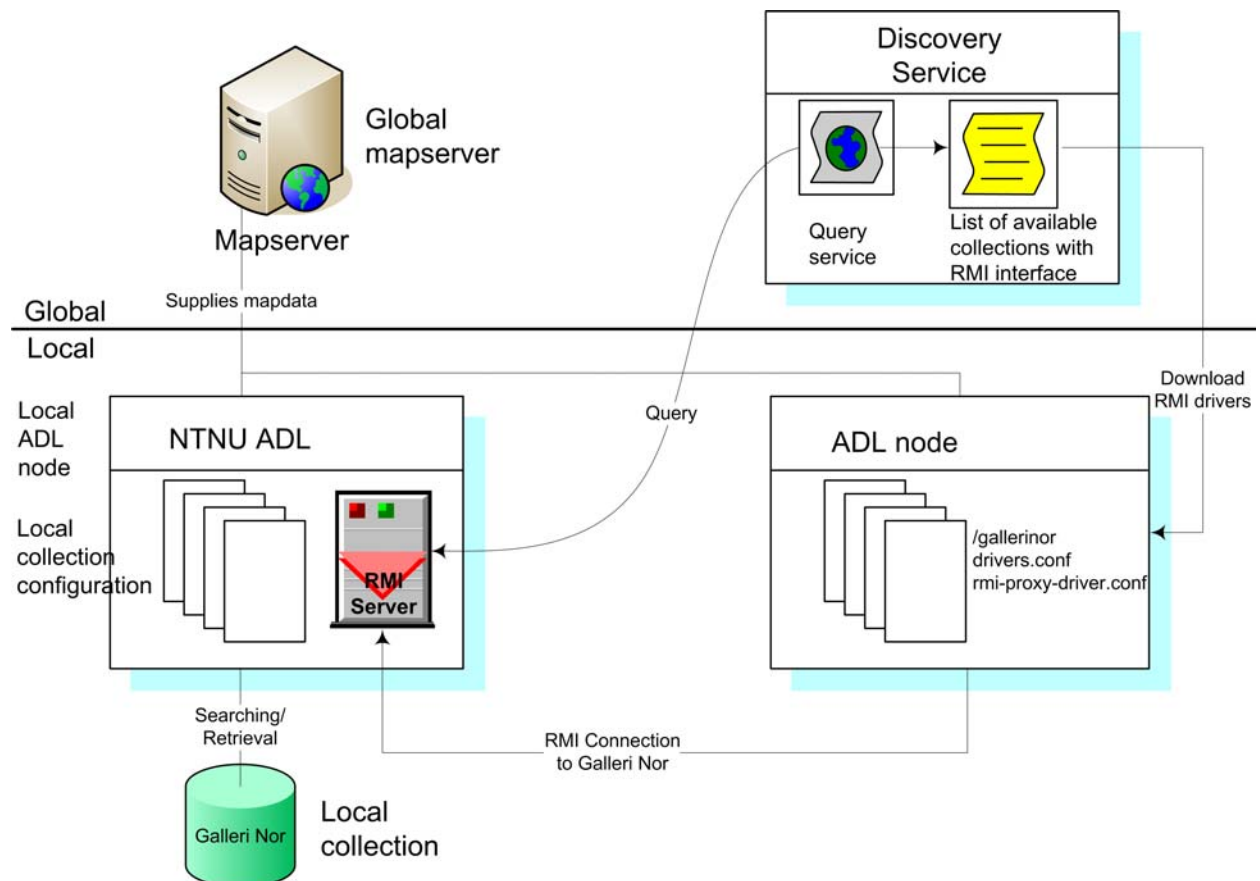


Figure 4-6. A global ADL discovery service, querying RMI servers, supplying a list of available RMI collections and configuration files to other nodes.

4. 3 Operational ADL nodes

4. 3. 1 Map and Imagery Laboratory (MIL)

The Map and Imagery Laboratory [66] ADL node at University of California, Santa Barbara (UCSB) can be viewed as the main ADL node. It is the most frequently updated node and it is the “display” node for the Alexandria Digital Library (ADL). The MIL node contains more than 15,000 holdings such as maps, images, and datasets. The holdings are available online for public download over the Internet. The items currently cataloged in ADL are a portion of the holdings of the UCSB's Map and Imagery Laboratory.

Collections at the MIL node

The MIL node contains various collections, the super class collections are listed below:

- Aerial Photography
- Educational Materials (DLESE materials)
- Projects (Maja Forest GIS)
- Digital Data Libraries (MIL Online, ADL Catalog, Air Photo Flights)

This node uses the standard webclient as its user interface. Since this node is always kept up to date with the most recent developments, this is the place to try out the latest functionality of ADL. For example spatial ranking was made available at this node the 7.th of October 2004.

4. 3. 2 Environmental Information Lab (EIL)

The Environmental Information Lab (EIL)[67] ADL node is used to find and access earth science data. The EIL has made its earth science data collections accessible through the ADL node. EIL is located at the University of California, Santa Barbara (UCSB).

Collections at the EIL node

The local collections at the EIL contain satellite photographs from various satellites, for example the NOAA-12 and NOAA14 satellites.

The EIL AVHRR (NE Pacific) collection is a collection of satellite photographs taken by Advanced Very-High Resolution Radiometer (AVHRR) sensors on board the National Oceanic and Atmospheric Administration (NOAA) polar-orbiting satellites, and the collection is about the North East Pacific. There is a lot of different satellite photographs in various collections at the EIL ADL node.

MODIS (Moderate Resolution Imaging Spectroradiometer) and DAAC (Distributed Active Archive Center) Data Pools are other collections of satellite photographs at the EIL node. MODIS is a key instrument aboard the Terra (EOS AM) and Aqua (EOS PM) satellites.

EIL uses the standard webclient with no modifications.

4. 3. 3 New Zealand ADL (NZADL)

NZADL[68] is the New Zealand implementation of the Alexandria Digital Library. The aim of NZADL is to provide collections of geographically referenced material in New Zealand that can be accessed free of charge. SERL (Software Engineering Research Laboratory) at the Auckland University is the founder of the NZADL service.

The project leader for NZADL is Sergei Gulyaev, e-mail: sergei.gulyaev@aut.ac.nz

Collections at the NZADL node

An important focus for ADL's collection is on information supporting basic science, including the Earth and Social Sciences.

The local collections in the NZADL node are:

- Scanned Aerial Photographs
- Scanned Satellite Photographs

These two collections contain aerial/satellite photographs from New Zealand. NZADL have also added several remote collections from the EIL and MIL ADL nodes.

NZADL has customized the webclient graphical user interface and removed the “Quick Place-name Search” part of the GUI in the standard ADL webclient interface. Only the “Catalog Search” part and interactive map is displayed in the search interface. This makes the GUI easier to understand since there are fewer options to choose from, and therefore less complexity for novice users.

The latest news on the web portal was added 13. May 2003, when the photos of the workshop was published on the site.

4. 4 ADL - a service or a total system

4. 4. 1 ADL as a total system

The ADL Middleware in combination with the webclient can be used as a georeferenced distributed digital library, and thus can be viewed as a total system. Various technologies and protocols are used in the middleware and the webclient, and each of the parts that makes up the whole can be viewed as components.

A total system is often chosen because the system covers the requirements of the organization by offering the most useful functionality and services, as well as being available at a reasonable price. ADL comes free of charge and offers a lot of useful services and technology for georeferenced digital libraries.

4. 4. 2 The middleware as a service

The ADL middleware can be used as a service in a digital library. The middleware can be used for implementing geospatial functionality in an already existing digital library. The beta version of DLESE uses the ADL middleware as a service that offered geospatial capabilities to the DLESE system.

The ADL middleware along with the webclient can be used as a total system, or the ADL middleware can be used as a service that offers geospatial functionality and services.

5 Installation of the NTNU pilot system

*“Obstacles are those frightful things you see
when you take your eyes off your goal.”
- Henry Ford*

This chapter describes the final installation and configuration of the NTNU ADL node.

5.1 Components of the node

There are three essential parts of a complete ADL node:

- A webservice
- The ADL middleware
- A collection

A webservice is needed to host the ADL middleware. The collections for the middleware can reside on remote servers or on the same server as the ADL middleware. A digital library has to have collections in order to make it interesting for users. There is no point in searching for information in a digital library that has no collection available for searching.

When autonomous collections are linked to an ADL node, and ADL nodes are linked together in a global information infrastructure, then the ADL can be used as a foundation for a global distributed digital library.

5.1.1 Webservice

*“Even if you're on the right track, you'll
get run over if you just sit there.”
- Will Rogers*

The Fenris Solaris server was chosen to host the ADL middleware. The Fenris server is a server used and managed by members of the information management group at NTNU. Java, Ant, and Tomcat had to be installed on the Fenris server before the installation of the ADL middleware.

The ADL middleware is dependent on Java and a servlet engine (Tomcat). Tomcat is dependent on Ant and Java. Thus I had to start with the installation of Java, Ant, and Tomcat on the Fenris server before I could install the ADL middleware.

Installations

The following software have been downloaded, installed and configured on the Fenris server:

- Java 1.4.2_04-b05 - Stable release
- Ant 1.5.2 - Stable release
- Tomcat 5.0.25 - Stable release

Both the Tomcat servlet engine and the ADL middleware needs Java, Tomcat also needs Ant for unpacking and installing WAR files. In Tomcat I configured a realm containing a username/password along with administrator/manager role assigned to that user. This user was then used for online management of Tomcat.

5.1.2 The ADL middleware

A prerequisite for my master thesis was to base my solution on the ADL architecture. The ADL middleware along with the webclient can be used as a georeferenced distributed digital library architecture. I decided to install an ADL node at the Norwegian University of Science and Technology (NTNU) and use the webclient as the default user interface.

When Java, Ant, and Tomcat were installed and configured on the Fenris webserver, then the installation of the ADL node could finally begin.

Installation and configuration of the ADL middleware

I downloaded the latest version of the ADL middleware (2.2.03) and since I used Tomcat as the servlet engine, I extracted/unzipped the installation file into the “/webapps” folder in Tomcat. When logged in as a manager on the Tomcat web pages it is possible to upload WAR files and install them through the web page, they will then be stored in the “/webapps” folder.

A text editor is used to edit the two files (domain.js and webclient.conf) needed for the configuration of the node. In these two files I specified the domain and the URL for the middleware, **document.domain="fenris.idi.ntnu.no"**; and **WEBAPP_URL: http://fenris.idi.ntnu.no:8080/adlmw/** were the values edited. When these files were edited and saved I restarted the Tomcat servlet engine for the changes to take effect.

The JDBC drivers for the databases that host the collections have to be placed in the “/common/lib” directory of Tomcat, or in “/adlmw/WEB-INF/lib” in the ADL middleware. If it is placed in “/common/lib” directory then all web applications on the Tomcat servlet engine can use them. If the drivers are placed inside the library of a web application, then only that web application can use them.

When the middleware was up and running I changed the default page from “welcome.jsp” to “index.jsp” by editing the “web.xml” file, and restarted the middleware web application for the change to take effect.

I later edited the “frame_blank.html” web page, this is the default information page in the webclient. I edited it and added simple instructions for searching a collection so new users can easier understand the webclient.

The “collection_opml.xml” is used to define the list of collections displayed in the drop-down menu in the webclient. When a collection is added it must also be added to the “collection_opml.xml” file so it is possible to select the new collection from the drop-down menu in the webclient. The “frame_blank.html” file should be edited to customize the information in the default webclient page to suit the site.

One or more JDBC drivers for the databases the collections are stored in also need to be added. When it was decided to use a MySQL database for the Galleri Nor metadata then the **mysql-connector-java-3.0.14-production-bin.jar** JDBC driver were added to the Tomcat “/common/lib” directory, so a MySQL JDBC driver became available for the ADL middleware. After a restart of the Tomcat servlet engine the ADL node was ready for the configuration of the collections.

Organization of the configuration files for the middleware.

Basedir: \$CATALINA_HOME/webapps/adlmw

Configuration files for the **middleware**:

- frame_blank.html
- /hierarchies/collection_opml.xml
- /javascript/domain.js
- /WEB-INF/config/webclient.conf
- /WEB-INF/web.xml

5. 1. 3 A local collection

The other prerequisite for my master thesis was to base my solution on “A collection from the National Library of Norway that contains georeferenced items“. Galleri Nor was selected as the collection of choice.

The Galleri Nor database at the Norwegian National Library has more than 70.000 photos from the time period 1880 to 1950. This collection contains georeferenced digital objects, and was therefore a good first choice for appending to the ADL node at NTNU. The digital photos in Galleri Nor are georeferenced with placenames.

The Galleri Nor collection is located in an Oracle database at the National Library of Norway. I got *read only* access to the collection. I was given a VPN client for Windows that was used for connecting to the National Libraries network.

Longitude/Latitude coordinates

The Galleri Nor metadata in the Oracle database at the National Library did not contain any coordinates that could be used for formally or visually specification of location. It was concluded that adding coordinates to the metadata description of the information objects in the Galleri Nor collection would be feasible in order to make it possible to explore the visual and formal methods of specifying location when searching Galleri Nor via the ADL middleware.

Since Galleri Nor is a database at the National Library of Norway, gaining access to edit the original database and adding longitude/latitude coordinates would not be likely. It was therefore a better solution to make a local copy of the needed metadata for searching and retrieval from the Galleri Nor collection and then add footprints (longitude/latitude coordinates) to the local metadata copy. The ADL node will use the local copy of the metadata expanded with longitude/latitude coordinates for searching, and the photos from Galleri Nor at the National Library will be retrieved when the resultset is presented to a user. This way it is possible to modify the metadata

of the local copy without any danger of damaging the original metadata in the database at the National Library

The local metadata copy also solves the problems associated with VPN, these problems are described in chapter 6. The local ADL node search the local metadata copy and then the photos are retrieved from Galleri Nor collection at the National Library by using URLs to access them.

Rudolf Nottrott and Håvar Valeur at UCSB (University of California, Santa Barbara) made two scripts that they ran for automatically extracting the longitude and latitude coordinates for the placenames in Galleri Nor using the ADL gazetteer at UCSB, and they also included norwegian letters (æ, ø, å) in the ADL gazetteer. The longitude/latitude coordinates were added to the Galleri Nor metadata-copy so each placename now was associated with a footprint that enabled visual and formal searching of the Galleri Nor collection. The local metadata copy was installed in a MySQL database.



Figure 5-1. A photo from Galleri Nor[49]

Figure 5-1 display a photo from the Galleri Nor collection at the National Library. The original metadata from Galleri Nor is listed below together with the coordinate-extended metadata the NTNU ADL node uses when searching. Some of the metadata fields from the original Galleri Nor metadata that contain *null* values have been omitted. Read chapter 4.1.4 for more information about the ADL bucket framework used for the ADL metadata description.

Table 5-1: Original metadata

Galleri Nor metadata	
Owner (Eier)	Norsk Folkemuseum
Previous owner (Tidligere eier)	Hansen, Knut Vigar
Photograph (Fotograf)	Lindahl, Axel
Dates (Datering)	1880 - 1890
Titles (Tittel)	
Search terms (Søkeord)	jernbanestasjon, stasjonsbygning, båter, havn
Supplementary information (Utfyllende opplysninger)	Prot: Trondheim Jernbanestasjonen
Name (Navn)	Trondheim jernbanestasjon - (avbildet)
Placename (Sted)	71 Trondheim 71 Sentrum Fosenkaia
Collection (Samling)	
Inventory number (Tilvekstnr)	NF.WL 01847
Internal number (Internnr)	NBR9312:02330

Table 5-2: ADL metadata

ADL Catalog Record	
Originators	Lindahl, Axel, fotograf Norsk Folkemuseum Hansen, Knut Vigar Trondheim jernbanestasjon
Dates	Begin:1880-07-01 End:1890-07-01
Titles	Trondheim. Motivs: jernbanestasjon, stasjonsbygning, båter, havn. By fotografer: Lindahl, Axel
Geographic-locations	Geographic Point: Latitude=63.4160 Longitude=10.416700
Types	photographs
Formats	JPEG
Subject-related-text	[Galleri Nor] MOTIV jernbanestasjon, stasjonsbygning, båter, havn [Galleri Nor] PRESISERING Trondheim, Sentrum
Identifiers	GALNOR Photo ID: 46134 GALNOR Inventory Number: NF.WL 01847

Adding a local collection to the local ADL node

David Valentine is currently the ADL developer to contact at UCSB for getting help with the configuration of ADL nodes and adding collections. When I installed the 2.2.03 version of the middleware I did the configuration of the ADL middleware, and David Valentine supplied new improved configuration files for adding the Galleri Nor as a local collection to the NTNU ADL node.

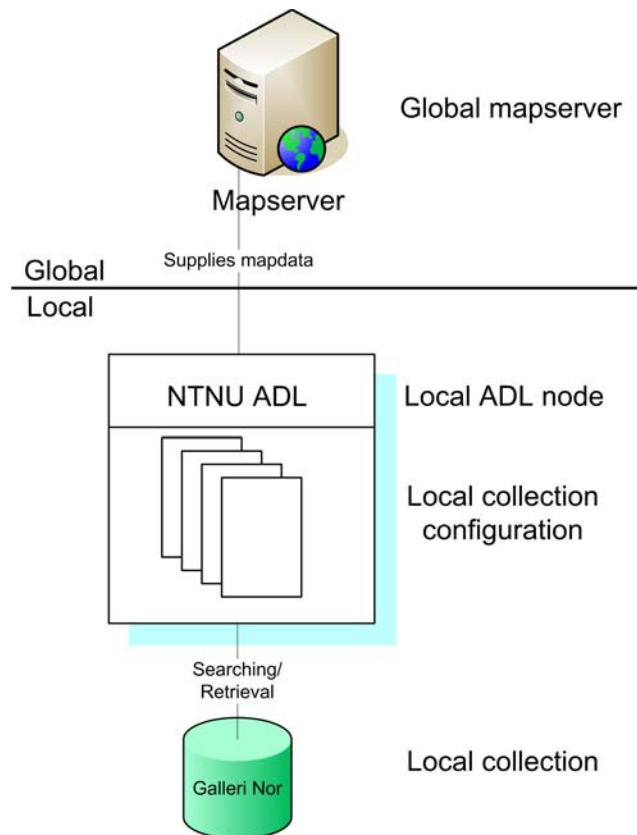


Figure 5-2. Adding a local collection to a local node

Following is a list of configuration files for each local collection.

Basedir: \$CATALINA_HOME/webapps/adlmw/WEB-INF/config/collections

Configuration files for a **local collection** in the middleware:

- /galerinor/access-report-template.xml
- /galerinor/browse-report-template.xml
- /galerinor/bucket-report-template.xml
- /galerinor/bucket99.conf
- /galerinor/database.properties
- /galerinor/drivers.conf
- /galerinor/metadata.xml
- /galerinor/query-translator.py
- /galerinor/CollTest.properties

Each collection appended to the ADL middleware will have its own catalog, for example “/galerinor”, where all the configuration files for that collection is stored. Of the nine configuration files listed above, the “CollTest.properties” file is the only nonstandard configuration file. More templates can be added to a collection, but the access, browse, and bucket views are the standard report templates for collections.

Making the collection available to the distributed network

In order to make the collections at the local node available to the distributed network the RMI (Remote Method Invocation) server at the local node must be started, and an e-mail must be sent to the developers at UCSB to inform them that new collections are available.

I started the RMI server, set the RMI register to run on port 6755, and restarted the Tomcat server for the changes to take effect. Since the ADL system currently doesn't have a global collection discovery service the only way to make the local collection “visible” for the distributed network is by sending an e-mail to the developers at UCSB. I sent an e-mail with the information to the developers at UCSB.

Once the RMI server is enabled all collections on the local ADL node will become accessible for other ADL nodes as long as they know the URL to the node, the collection name and the port the RMI register is running on. An ADL node can include local collections made available at other local nodes as remote collections in that node.

An ADL node can only search the local and remote collections in their node. By enabling the RMI server I made it possible for other nodes to include my local collection as a remote collection in their node, thus my local collection is searchable from the whole system. The sharing of collections are done as a peer-to-peer based distributed system.

5. 2 The NTNU ADL node prototype

*“The obscure we see eventually.
The completely apparent takes a little longer.”
- Edward R. Munon*

On the 7th of September 2004 the 2.2.03 version of the ADL middleware was installed on the Fenris server. Galleri Nor was added as a local collection to the NTNU ADL node on the 27th of September after solving some issues with the collection configuration. More details about installation history and issues are described in chapter 5.2.3.

5. 2. 1 Searching the local collection

Figure 5-3 below shows a logical view of how the searching of the local collection is done at the NTNU ADL node.

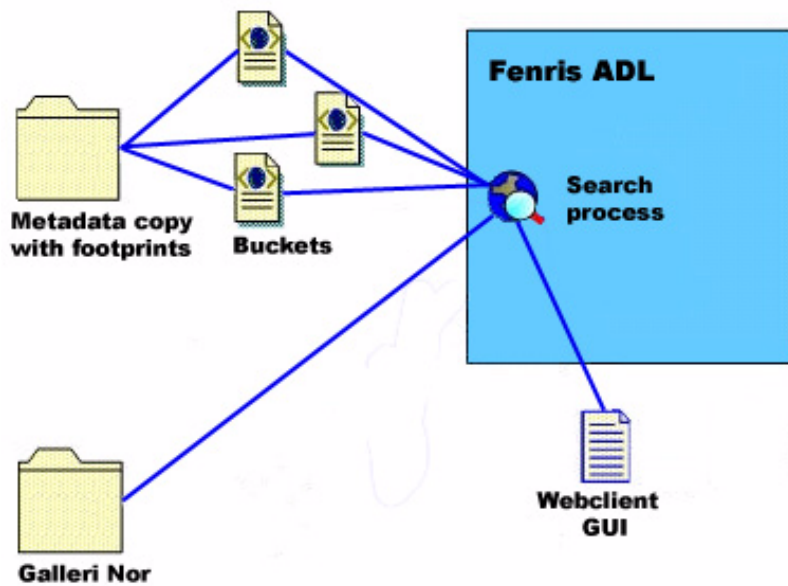


Figure 5-3. Logical view of how the Galleri Nor collection is searched.

1. A user sends a query from the webclient GUI (Graphical User Interface). 2. An asynchronous search process performs a search via the buckets that is mapped to the local metadata-copy. 3. A resultset is returned to the search process. 4. The search process retrieves the photos from the Galleri Nor collection at the National Library via URLs. 5. The retrieved results are displayed to the user in the webclient GUI.

All photos displayed to the information seekers are retrieved from the Galleri Nor collection at the National Library. The local metadata-copy is used for constructing a resultset to be retrieved from the National Library.

What are the pros and cons of this solution?

Pros:

- Can change/add footprints or other metadata
- Searching is done with minimum strain on the National Library system (only retrieve photos from the resultset).
- Avoid the need for a VPN client and thus VPN time-outs

Cons:

- Increased network latency and delays

5. 2. 2 New access points for the Galleri Nor collection

The metadata-copy that the NTNU ADL node uses for searching was expanded with footprints for each information object. The footprints were added to enable visual and formal searching of the Galleri Nor collection. Placename (informal method of specifying location) searching was already partially supported in the Galleri Nor web interface that the National Library supply. Gal-

leri Nor does not use a gazetteer, this is feasible however according to the findings of Marit Olsen [50].

The NTNU ADL node supports all three methods (formal, informal, and visual) of specifying location in geolibraries. It is now possible draw a bounding box on an interactive map to find photos from specific locations without knowing the placenames or coordinates that is used in the metadata description of those photos. If the coordinates for a feature is known, it is possible to enter the coordinates to formally search a region without knowing where on the interactive map that feature is located, or informal searching can still be done through the use of placenames.

The ability to access any information object in Galleri Nor based on it's spatial georeferenced location is added as a point of access for the collection. Everything that happens, happens in time and in space so being able to interpret problems through this paradigm offers new access points and new powerful ways to search for information. To be able to find photos from locations where placenames are unknown is a new way to access the information objects in Galleri Nor. The ADL node can "show all information available about this location", so it is possible to solve problem in a spatial context.

5. 2. 3 A brief installation history

Following is a short installation history showing some of the problems and solutions that have occurred during the last year. This list contains only major events in the history of the NTNU ADL node at the Fenris webserver.

- **23. Mars 2004:** Accessed the Galleri Nor collection at the National Library for the first time.
- **19. April 2004:** The port 1521(Oracle port) is opened on the camilla.nb.no server for Fenris IP address.
- **20. April 2004:** First ADL node up and running on the Fenris server. Had problems getting this to work so installation of a new middleware node was started.
- **14. June 2004:** ADL middleware-2.2.01 installed and configured on the Fenris server.
- **23. June 2004:** Local Galleri Nor collection installed at the Uranus server at the NTNU campus.
- **29. June 2004:** Local Galleri Nor collection made searchable in the local ADL node. Middleware version 2.2.01. User and password for the database with read access to the database at Uranus. The MySQL user is configured so it can be used from the machine where the ADL middleware is running ('DBusername', '%.ntnu.no'). Database names and table names are case sensitive in MySQL Server on operating systems that have case-sensitive filenames (such as most Unix systems), Windows systems have case insensitive table names and database names. I had to change to the correct case in configuration files where the database and tables names where used. The column names in the tables are case insensitive though, regardless of operating system.
- **Middle of August 2004:** Interactive map suddenly stopped working because the map server in USA was updated to the ArcIMS 9 standard, this broke some old versions of the middleware. Version 2.2.03 of the middleware were launched to fix this problem.
- **07. September 2004:** Installed the new middleware 2.2.03 but still the interactive map area did not work.

- **09. September 2004:** Interactive map is now working on both mw-2.2.01 and mw-2.2.03 on Fenris. I helped the team at UCSB locate the problem by testing what ports I could not connect to. They found it was a firewall on their side that blocked access to one of their servers.
- **14. September 2004:** Local Galleri Nor collection made searchable by the new version 2.2.03 of the middleware. The IT department updated the indexes of the MySQL database where the Galleri Nor copy is located and this greatly improved search speed.
- **15. September 2004:** Received new configuration files from David Valentine, followed instructions for updating my configuration of the middleware and Galleri Nor collection configuration. When I updated the “middleware-2.2.0.jar” file as I was told to do, this resulted in the search servlet for the web page not loading correctly, showed a “File Not Found” error message.
- **16. September 2004:** Updated configuration of Galleri Nor, but after the updates I still got no results when searching. Fixed the search servlet bug by adding 3 values to the “middleware.conf” file as Greg Janée told me. David Valentine sent me a new “middleware-2.2.0.jar” file that fixed it too, but Greg’s solution of the problem was chosen.
- **20. September 2004:** Installed new configuration for Galleri Nor, it returned no results when searching.
- **22. September 2004:** Found out that the problem with the current configuration is an SQL problem. We are running MySQL 3.23.53 at Uranus and they tested the configuration on MySQL 4.1.3. Never got asked what version we were running.
- **27. September 2004:** Tested new configuration using the galnor database at the UCSB campus to test the configuration and see it working, currently the NTNU ADL node searches the UCSB database.
- **28. September 2004:** “Found” a security flaw that makes it possible to disable and enable collections, and to upload new RMI collections dynamically to a node. Can do this at MIL, EIL and the NTNU ADL node without any authorization requested at all. It should not be possible to make use of a system’s test pages without the authorization needed for doing so.
- **19. November 2004:** Edited “frame_blank.html” and added *How to search Galleri Nor* on the default web page for searching.
- **24. November 2004:** Enabled RMI server on port 6755

The help of the developers at UCSB has been necessary in order to get the NTNU ADL node up and running with a searchable collection. The installation of the ADL node is easy in the latest versions of the middleware, the hard part is to add and configure a collection.

6 Discussion

*“The trouble with the world is that the stupid are
cocksure, and the intelligent are full of doubt.”
- Bertrand Russell*

6. 1 Problems and solutions in the installation phase

What problems, if any, did arise during the different installations? Was it possible to solve the problems, and if so, how were they solved?

The three essential parts of the ADL node will serve as a framework for this discussion:

- A webserver
- The ADL middleware
- A local collection

6. 1. 1 A webserver

The installation and configuration of Java, Tomcat and Ant on the Fenris server went without any big problems. The online documentation of the installation processes were easy to comprehend and execute. It was an interesting observation that *gnutar* that is used on most Linux and Unix systems is incompatible with the *tar* version on Solaris systems. On Solaris systems *gtar* is used (to start *gnutar*) to correctly untar tar files made by *gnutar*, unless *gtar* is set up to be used as the default tar version for the tar command.

6. 1. 2 The ADL middleware

The installation and configuration of the ADL middleware is relatively easy as the documentation is quite good. The documentation of the installation of an ADL node has improved greatly [69], so the installation process is much better documented now than in earlier versions of the middleware. The online documentation for the installation and configuration process is easy to follow step by step, however the default file for presenting information in the webclient search page is not mentioned anywhere in the documentation. The “frame_blank.html” file should be mentioned in the installation documentation.

In the latest version of the installation documentation web page it is possible to add comments at the bottom of the web page, so problems encountered can be posted there. If problems are encountered during the installation process, there should be hyperlinks between the installation documentation web page and pages that can be used to solve the problem, such as a FAQ (Frequently Asked Questions) or a troubleshooting page.

The configuration of the middleware is done by editing five different configuration text files that are stored in different folders in the file hierarchy:

- `frame_blank.html`
- `/hierarchies/collection_opml.xml`
- `/javascript/domain.js`
- `/WEB-INF/config/webclient.conf`
- `/WEB-INF/web.xml`

The last three configuration files listed are those that need to be edited to get the middleware up and running. The “`collection_opml.xml`” file only needs to be updated each time a change in the collections available for information seekers needs to be addressed. The “`frame_blank.html`” can list news events for the local node, and/or tips for searching the collections at the local node.

Java, Tomcat, or Ant may need to be updated when a new version of ADL is installed, since the ADL node is dependent on Java and a servlet engine (Tomcat, that in turn is dependent on Ant). I did not have to upgrade any of the required software before the last installation of the ADL node since I had updated the required software before the installation of the previous version of the ADL middleware.

The interactive map did not load as it was supposed to in the beginning, but this was due to a firewall at UCSB that was blocking access to the mapserver. This was solved after I helped the developers at UCSB locate the problem by identifying what ports I could not connect to.

6. 1. 3 Adding a local collection to the NTNU node

The discussion below is about the two different problem domains related to collections:

- Gaining access to collections from a collection provider
- Configuring and adding a new collection to a local ADL node

Gaining access to collections from a collection provider

In september 2003 at the ECDL2003 in Trondheim a meeting was arranged between the involved parties (NTNU, ADL developers from UCSB, and the National Library of Norway), and it was agreed that the National Library would grant us *read only* access to the Galleri Nor collection.

The technical solution used for gaining access to a collection has an impact on what problems will be encountered. If a firewall blocks access to the collection there are several legal ways that can be used to allow traffic between the computers that need to communicate. VPN (Virtual Private Network), SSH tunneling, or simply letting requests from a certain IP address pass through the firewall unhindered. Opening up for IP addresses or using SSH tunneling works fine across different operation systems (Windows, Solaris, Unix, and so on), however VPN have some limitations.

Limitations associated with VPN:

- VPN clients are not operating system independent, a VPN client for Windows can't be used on a Solaris server.
- VPN can be set to time out after an interval of no activity
- VPN clients can only be part of one VPN (only the most expensive VPN clients from Cisco support multiple VPN connections simultaneously)
- A computer that connects to a VPN becomes part of the new network and loses the services it had available in the original network. This can also be a security risk if multiple persons use a shared computer and that computer becomes part of a new VPN, since all the persons using that shared computer will have equal access to the new VPN.

The main problem is that VPN does not stack (unless the most expensive VPN client from Cisco is used) so VPN can only be used for one collection provider. When VPN is used the local server will no longer be part of the local network, but belong to the VPN of the collection provider. The time out after a small period of inactivity also makes the VPN solution a bad solution for communication across a firewall. Allowing a certain IP address to pass the firewall or using SSH tunneling are better solutions for communication between the server that hosts the ADL node and the different collections.

Configuring and adding a new collection to a local node

The installation of the ADL middleware is relatively easy, but the configuration and adding of new collections to an ADL node is not trivial.

Eight different configuration files need to be implemented for each collection. They are written in a variety of different standards and formats. Some of them are short and easy to understand like the "database.properties" file that only contain the database user-name and password, other configuration files are several pages long and hard to comprehend.

Problems related to collection configuration

- There are few comments in some of the configuration files, making the understanding and comprehension of the information in them hard.
- There are no URLs in the configuration files to online examples of how the files should be implemented.
- What script/programming language that is used in a configuration file is listed in the web description of the file, the "query-translator.py" file is for example written in Jython not Python. File endings can be used to guess some languages (.js = javascript, .xml, and .conf and properties are java-property files).
- Some of the configuration files have dependencies to other configuration files (aka export and import values from other configuration files) without this being well documented in any way. Low coupling and high cohesion is a design principal that is important to follow in order to make large complex systems easier to understand.
- The purpose of the configuration file is listed on web for most of the files but not all. So it is hard to know the purpose of some of the configuration files for novice users of the system.
- If the configuration file need a DTD it is only available in some of the online documentation (and not available for all xml files for example "collection_opml.xml"). Maybe DTDs should be listed with an URI or example in the file that need it

- There are lots of different values that can be edited in the configuration files, but what values are required to change in order to make the system work? Most values in some files are only default values that should not be touched.
- Which formats should the values be written in, and which standards are used in the configuration files?
- Some of the features of ADL are unique (for example the bucket framework), and may need a short descriptive explanation so new users become aware of the concept in order to use it. Online documentation exists, but the documentation can be hard to understand for new users of the ADL system.
- If there are special notes or comments about a file they should be listed as comments in the configuration file (this only exists in the “opdl_collection.xml” file currently).

All these factors add up, and thus adding new collections to an ADL node become a rather difficult process, at least for persons that don't share a developers complete understanding of the whole ADL system. Currently it is rather difficult to configure and add new collections to an ADL node.

“Flexibility comes at the expense of simplicity-it is not easy to configure you first few collections. Those who have worked with metadata may remember that even the first experiences with simplest metadata standards can be frustrating process.”[65]

The quote above comes from a note in the FAQ (Frequently Asked Questions) for version 2 of the ADL middleware, it states that simplicity is sacrificed for the sake of flexibility (supporting heterogeneous collections).

In order to configure and add a new collection to an ADL node it is important to understand the concept of the ADL bucket framework and how to implement it. **A variety of different standards and formats are used** in the configuration files, and a new user have to know how to make use of all these different standards and formats on top of understanding the concept of mapping metadata to the bucket framework in order to add a collection to an ADL node. Understanding the concept of the bucket framework is not hard, implementing it though is another story.

“Any distributed solution, however abstract and implementation independent it initially is intended to be, is bound to be influenced by the underlying middleware it is built upon. Categories of middleware solution appear quite different in their abstraction of the distributed system, and different middleware solutions are often not interoperable. A main problem for digital library interoperability is the heterogeneous usage of middleware in current digital library architectures.”[1]

The configuration of the ADL middleware and the configuration needed for adding collections to an ADL node are influenced by the underlying standards and formats used to implement the ADL middleware. The Python code that needs to be written in the “query-translator.py” file is a good example of how the underlying standards and formats add complexity to the configuration. A brief introduction to the query translation file can be found in [54].

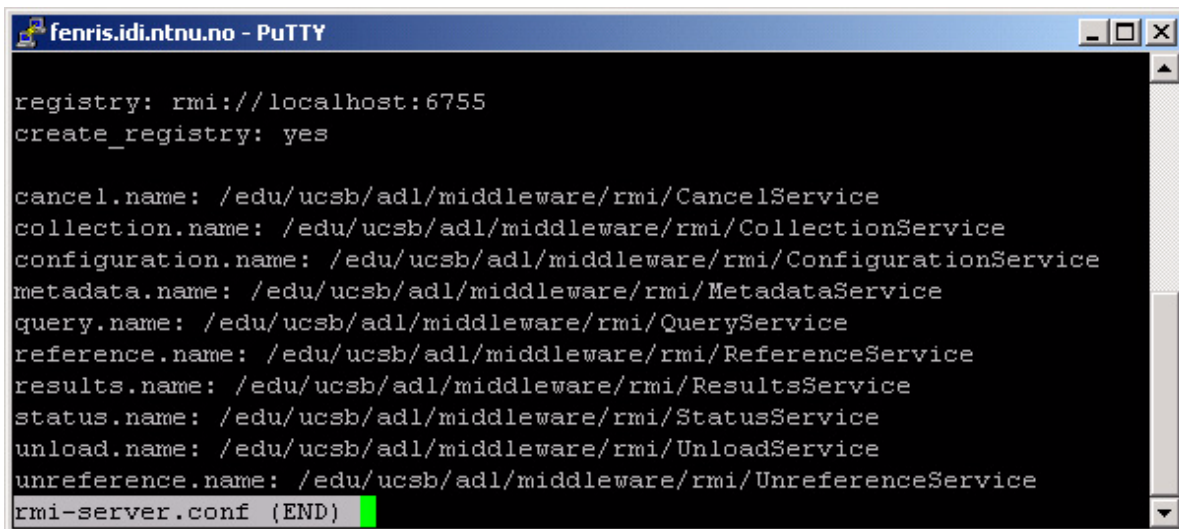
The documentation for adding new collections to an ADL node may be hard to comprehend for a novice user, as he usually does not have detailed knowledge of the system. The documentation for the installation and configuration of the middleware node is far easier to comprehend and to follow as it is a “recipe” that describe how to get the middleware up and running. The installation procedures and installation documentation have gotten better as the ADL project has moved forward.

David Valentine is at the time of this writing assigned to help making the configuration files for new ADL nodes and adding new collections. Having a person assigned to writing configuration files for others that want to make use of the ADL system might be an indicator that the system is currently hard to configure for novice users of the ADL.

Thanks to the team at UCSB the Galleri Nor collection got geographic coordinates added to NTNU’s metadata copy. The configuration written by David Valentine was developed for the metadata-copy extended with footprints stored in the MySQL database on the Uranus server. The collection configuration made use of set operators that are only supported in MySQL 4.x or newer, since the MySQL server on Uranus is running version 3.23.53 the new configuration did not work with this database. Currently the NTNU ADL node searches a metadata-copy at UCSB stored in a MySQL 4.1.3 server. The previous Galleri Nor configuration written by Rudolf Notrott was compatible with MySQL 3.23.53. Lack of communication about software version may have lead to the development of an incompatible configuration. An optimal database layout for Galleri Nor that would have simplified the configuration can be found in appendix D.

6. 1. 4 Making local collections available to the distributed network

It was an easy process to make all the collections at the local node available to the distributed network. All that needed to be done was to set the port number the RMI register would run on, and set the create_registry value to yes.



```
fenris.idi.ntnu.no - PuTTY
registry: rmi://localhost:6755
create_registry: yes

cancel.name: /edu/ucsb/adl/middleware/rmi/CancelService
collection.name: /edu/ucsb/adl/middleware/rmi/CollectionService
configuration.name: /edu/ucsb/adl/middleware/rmi/ConfigurationService
metadata.name: /edu/ucsb/adl/middleware/rmi/MetadataService
query.name: /edu/ucsb/adl/middleware/rmi/QueryService
reference.name: /edu/ucsb/adl/middleware/rmi/ReferenceService
results.name: /edu/ucsb/adl/middleware/rmi/ResultsService
status.name: /edu/ucsb/adl/middleware/rmi/StatusService
unload.name: /edu/ucsb/adl/middleware/rmi/UnloadService
unreference.name: /edu/ucsb/adl/middleware/rmi/UnreferenceService
rmi-server.conf (END)
```

Figure 6-1. Screenshot of the “rmi-server.conf” configuration file

However there is one feature I feel that is missing. When the RMI server is enabled then all the collections at the local node are made available, it is either all or none. It should be possible to decide what collections to share, and what collections should remain private for the local node.

Figure 6-1 can also be used to illustrate another problem. The figure shows the “rmi-server.conf” configuration file. There are twelve lines of configuration values in this file, and of these only the last part of the first line (the RMI register port number) and the boolean value of the `create_registry` line need to be edited. The ten next lines are just default values used by the middleware, these lines are irrelevant for a local administrator at a local node and only increase the complexity of understanding the system.

6. 1. 5 Other problems encountered

In the middle of august 2004 the global map server at UCSB was upgraded to the ESRI ArcIMS 9 standard and the interactive map suddenly disappeared from all ADL nodes except the MIL node. First I tried to install a new file that was supposed to fix the interactive map on the NTNU ADL node. That didn't work and I was told to upgrade from middleware 2.2.01 to 2.2.03. I then installed the 2.2.03 versions on the middleware, but that didn't fix the interactive map either. We later discovered that a firewall at UCSB was blocking access to the mapserver for nodes that weren't located at the UCSB campus. Not all old ADL nodes are compatible with the new ArcIMS standard for the interactive map, but the interactive map in the 2.2.01 middleware is compatible with it. Currently the 2.2.03 version of the ADL middleware is running on Fenris.

The installation of a file that was supposed to fix a problem resulted in other parts of the system malfunctioning.

6. 2 ADL as a research project

The ADL project is a research based project seeking to find solutions to problems that are related to distributed georeferenced digital libraries. Their main focus therefore is research problems and not making a commercial flawless system that will be widely spread and made use of, nonetheless they are interested in spreading their system to the public. A research project is to a higher degree a moving target (in configuration management) than commercial products that have a more defined purpose and limitations. Research projects are on the frontier of the research-field and are therefore more prone to changes as solutions and implementations for solving problems change more rapidly than in more stabile environments. Research projects are not finished systems, but systems that are continually evolving as the research is ever-moving forward in one or more directions, this evolution makes the research project a moving target.

6. 2. 1 The georeferencing paradigm

The georeferencing paradigm where geospatial knowledge can be used as an access point to information stored in a collection is one of the unique features of digital libraries that can't be easily replicated by traditional libraries. This unique feature will make georeferenced digital libraries increasingly popular if people learn to solve problems within the spatial paradigm of thinking. Everything happens in time and space. There will be a golden age for georeferenced digital librar-

ies if people learn to solve problems in the spatial paradigm, and are aided by georeferenced digital libraries and georeferenced collections that are publicly available.

Making the process of installation and adding of collections to georeferenced digital libraries easier is therefore very important to further the distribution and thus the availability of georeferenced digital libraries. It is important to make the technology available to information seekers and to make the georeferenced digital libraries easy to comprehend and use. The usefulness and ease of use of a system from the users viewpoint are the key factors of what leads to user acceptance of digital libraries.

6. 2. 2 Cooperation with the ADL developers

The cooperation and communication with the developers of the Alexandria Digital Library have been a necessary part of bringing the NTNU ADL node up and running with the a searchable collection. Rudolf Nottrott, David Valentine, Greg Janée, and Håvar Valeur are the persons I have communicated and cooperated with at UCSB. Without their help the ADL node at NTNU would not have had a searchable collection. Håvar and Rudolf added the footprints to the copy of the Galleri Nor metadata, and Rudolf wrote the first basic configuration for adding Galleri Nor to the NTNU ADL node.

David Valentine wrote the refinement and new configuration for the Galleri Nor collection at the NTNU ADL node. The new configuration had some added features like *highlight in map*, and links to the National Library web pages for descriptions of each of the photos retrieved.

I have helped solve problems like the firewall problem that caused the interactive map to disappear from all ADL nodes outside of the UCSB campus, and given feedback to beta-versions of the new webclient interface for presentation of search results. Documentation and interaction with the ADL middleware have changed a lot since the first version I installed the 20th of April 2004, compared to the ADL node that is now up and running. I hope my feedback along the way have been useful, at least I have communicated some of the concerns that a new user of the system would have. The changes along the way have improved the ease of use and the documentation of the ADL greatly.

7 Possible improvements and recommendations

*“Most good judgement comes from experience.
Most experience comes from bad judgement.”
-Unknown*

The recommendations have been divided into two categories:

- General recommendations
- Specific recommendations

The general recommendations are recommendations for both the ADL and other digital libraries, discussing topics of a general character. Specific recommendations are proposed improvements that are focused on the ADL system.

7.1 General recommendations

*“I don't want any yes-men around me.
I want everybody to tell me the truth
even if it costs them their jobs.”
- Samuel Goldwyn*

Digital libraries are characterized by the use of many different standards, formats, and technologies. Should a local administrator need to be an expert in all the different technologies, formats, and standards used in order to install and configure a distributed digital library node or add a collection to it? I think the key to answering this question is linked to the identification and understanding of the role of the different actors/user groups that make use of a digital library.

7.1.1 Documentation

When designing software it is important to know for whom the software solution is designed. Identification of the different user groups is essential to the design of the software solution, it is important to know **for whom** the design is made so that the terminology and complexity in both software and documentation of the software match that of the user group in question.

For digital libraries, including ADL, there are 3 primary user groups:

- Software developers
- Local administrators (install, configure, and maintain the local digital library node. Update and add new local collections)
- Information seekers (users using the local nodes for searching georeferenced information)

Having identified three different core user groups of a system it is obvious that we need to document the system at different levels that satisfies the knowledge and “need to know” for each user group.

Software developers document the system for other developers. This documentation will be technical and detailed as developers must have a complete understanding of the system. In research projects and commercial enterprise projects the system documentation will most likely not be publicly available. A research project usually publish its findings for the research community.

The role of the **local administrator** is the role I will focus on since I believe herein lies the key to understand the nature of the problems related to the configuration and adding of new collections to ADL nodes. What is this role of the local administrator then?

A local administrator should be able to read and follow instructions for configuration and adding collections without needing to know a lot of different standards and formats used in the middleware of the digital library, or how to implement the different search buckets.

Local administrators are users that are computer literate. It is important that the documentation for this user group is written so that it is easy to understand how to install and configure collections the first time around. The documentation should provide a structured description of how to install and configure the middleware and collections, like a checklist to go through to make sure that every task is done. Local administrators generally want anything that can aid in configuration, and since they often have a lot to do, things need to go smooth and with few errors. Documentation should be to the point (what needs to be done to get things to work) with brief descriptions to get an overview of the system and understanding what it is supposed to do, just like a *cookbook*.

It is important to describe the information local administrators need at their level. Providing information on how to test the configuration (with examples), where configuration files are located, and what programming language is used in a certain configuration file (Jython, XML, Javascript, Java properties). The system should also provide understandable error messages to the local administrator when configuration errors or malfunctions in the system do occur. It should be unnecessary to rely on sending error messages to the developers in order to understand what is wrong and how to correct the error.

There should be a FAQ page where a local administrator can search for error messages in order to understand these messages and how to solve the problem. An understandable error message that also has a link to a FAQ page where it is listed how to solve the problem, or possible to post the problem so the developers can have a look at it, and other searching for the same problem can see it has occurred before (with responses on how to solve it), would maybe be an improvement.

In ADL the *test-forms.jsp* web page is useful for checking when there are errors in the configuration of the middleware and collections. If the status button is clicked, then a page will list the status for the last queries. The error messages at the *test-forms.jsp* page is more understandable than getting no results when trying to search for a place (some errors are not reported to the user interface when searching).

In the installation documentation [70] for the middleware there should be links to documentation for troubleshooting. There is a troubleshooting section in the documentation for ADL, but a more complete coverage of troubles and links to that documentation should be easier to find. Linking between the installation documentation (and from error messages) and the troubleshooting documentation would make this information easier to find.

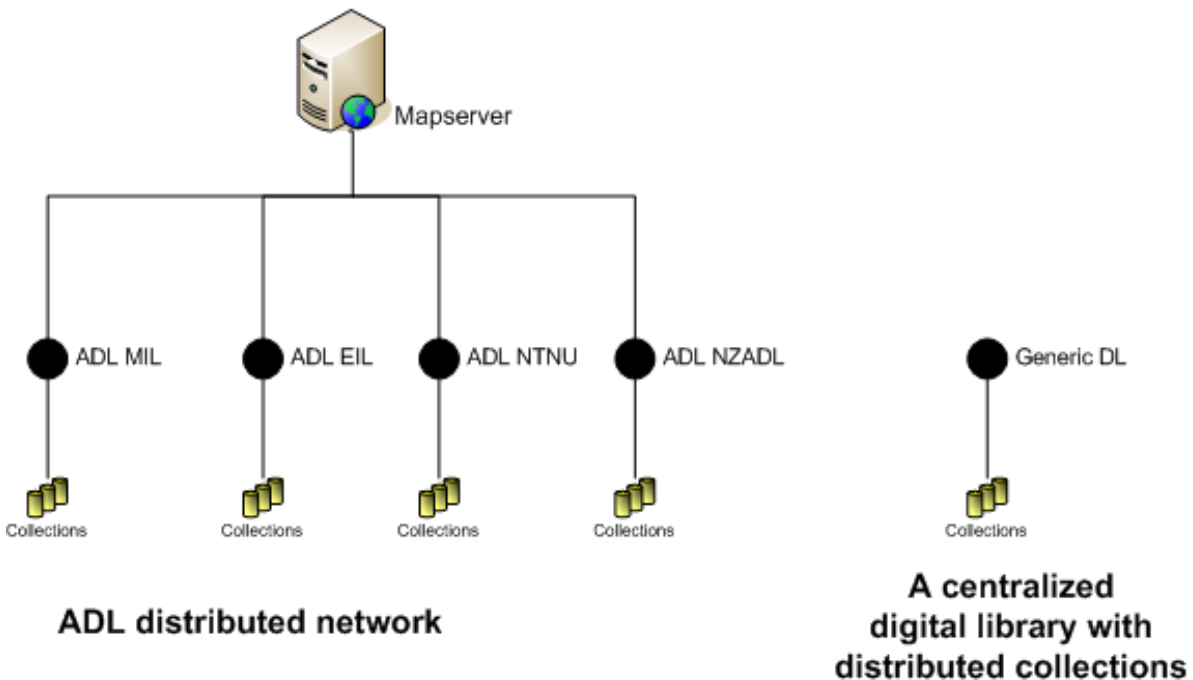


Figure 7-1. The division between the local administrator and software developer user groups

In digital libraries, where the nodes are not distributed, the roles of the developer and local administrator are often overlapping. After the development and deployment of a digital library, the developers will often assume the role of the local administrator through maintenance and adding of collections. This way the expert knowledge of the digital library is still available within the organization, and further development can also be done in small expansions.

In a global distributed digital library network with distributed nodes, there will be a more apparent division between the role of the local administrator and the role of the software developers. The developers of the digital library may assume the local administrator role in the main node in the network, but at the other nodes the local administrators will not be former developers with expert knowledge of the system. When the roles of the local administrator and the software developer are clearly separated, then the importance of user-friendly design and documentation for the local administrator user group becomes apparent.

The documentation for **information seekers** must be easy to comprehend, hiding unnecessary complexity, and using terminology that fits this user group. Links to the documentation should be easy to find in the interface of a digital library if help is needed. First time users need special attention, and a tutorial can be of great help. Some of the factors that lead to user acceptance of digital libraries are given in the figure 7-2 below.

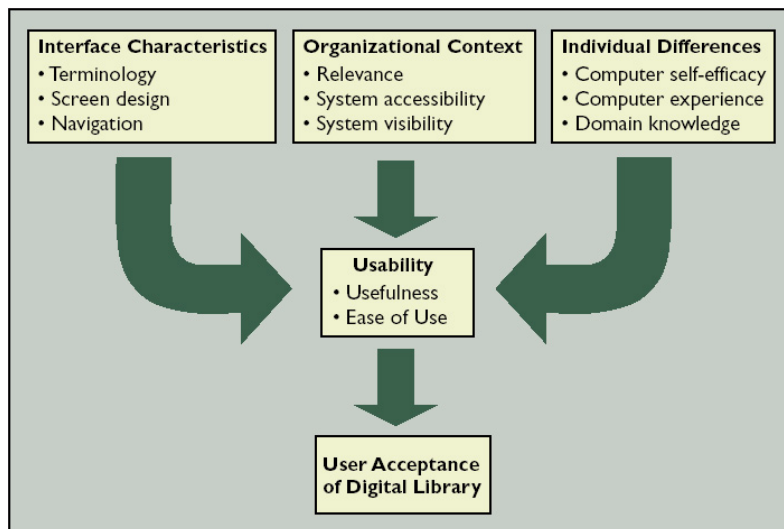


Figure 7-2. Model of user acceptance of a digital library. [71]

The factors that lead to user acceptance are most important for the information seekers user group, but do also apply to the other user groups. Developers will be least affected since they are the ones that develop a system. More information about user acceptance of digital libraries can be found in the ACM article [71].

To summarize the key points above:

- Identify the user groups of the system
- Hide complexity by only showing the information needed by the specified user group
- Documentation must be written with a complexity and a terminology that can be understood easily by the user groups for that system

7. 1. 2 Configuration Management

*“I think that's the last bug”
- Deluded programmer*

Configuration management is an umbrella activity that should be applied throughout the software process. Modifications invariably occur while software is being developed and after it is released to a customer, the five software configuration management (SCM) tasks; identification, version control, change control, configuration auditing, and reporting, are applied to bring order to the chaos change in large software projects can lead to.

Research projects like ADL, where the frontiers are pushed forward, are projects where change will occur more frequently than in normal software development processes. It is therefore important that SCM is an integrated activity in the development process so change can be anticipated and managed.

All information produced as part of the software process become part of the software configuration. The configuration is organized in a manner that enables orderly control of change. The development environment that is used to create software can also be placed under configuration

control (this can be useful when the software developed depend on other software to function correctly). ADL is dependent on a lot of other software/technologies to function correctly (Ant, Tomcat, tar/zip, JDBC compliant database, Java, TCP/IP), and changes that occur in these software programs can break the ADL software.

The scheme used for **identification** of software objects must recognize that objects evolve throughout the software process. Each object can have an evolution graph that describes the change history of the object. A software object 1.0 undergoes revision and becomes 1.1. Minor corrections or changes result in versions 1.1.1 and 1.1.2, a major update will update the version number to 1.2. A major modification to an object results in a new evolutionary path 2.0. Changes can be made to any of the versions, but not necessarily all the versions. The use of a proper identification scheme can make the understanding of an evolving system easier to understand.

Version control in the context of SCM described by [72]:

“Configuration management allows a user to specify alternative configurations of the software system through the selection of appropriate versions. This is supported by associating attributes with each software version, and then allowing a configuration to be specified [and constructed] by describing the set of desired attributes.”

For each version of the software there is a collection of software configuration items (data, documents, source code), and each version may be composed of different variants. One variant of a version of ADL can be shipped with the Remote Method Invocation component as part of that version, while another variant of that version can be shipped without the RMI component.

The SCM **change control** task is a procedural activity that ensures quality and consistency as changes are made to a configuration object. The change control activity always starts with a request for change, the request for change leads to a decision to make or reject the change, and culminates with a controlled update of the software configuration item that is to be changed.

The **configuration audit** is a software quality assurance activity that helps to ensure that quality is maintained as changes are made. The testing done in this SCM task ensures that changes done to the software will not break or cause problems with other parts of the software. Changes are tested before they are shipped to customers to make sure the changes work as intended and don't break other parts of the software.

Configuration **status reporting** provides information about each change to those with a need to know. Reporting is a task that should answer the following questions: What happened? Who did it? When did it happen? What else will be affected?

Reporting plays a vital role in the success of large software development projects. It is important that “*the left hand knows what the right hand is doing*”, and reporting plays this role of relaying information and improving communication among all people involved. Consistency is important both when communicating with customers and in the documentation of a system. The reporting task can be a boost to the communication and thus affect consistency in a positive way.

Why should configuration management and configuration tools be used in distributed georeferenced digital libraries?

Effective configuration management delivers:

- Increased reliability
- Improved quality
- Increased productivity
- Improved management control

7. 1. 3 Network communication through firewalls

The use of VPN solutions (described in chapter 6.1.3) should be avoided even if they look good on paper and are easy to implement. Time-outs after short periods of inactivity, the non stacking issue, and the problems associated with the server becoming part of the new VPN and no longer a part of the local network are major limitations with the VPN approach.

Allowing a certain IP address to pass the firewall or using SSH tunneling are better solutions to secure communication between a digital library and its collections.

In our case it was easier to make a local copy of the needed metadata for searching and retrieval of the photos from Galleri Nor via URLs since this access method is not blocked by the firewall.

7. 1. 4 Communication in distributed organizations

*“If one morning I walked on top of the water
across the Potomac River, the headline that
afternoon would read, ‘President Can’t Swim.’ “
- Lyndon Johnson*

When a distributed system is used on a global scale the ability to communicate within the distributed organization will be affected.

EMS instant messaging	Chat Instant messaging Video conferencing EMS Application sharing	Simultaneous (synchronous)
E-mail Voice mail Discussion boards Scheduling Calendaring	E-mail Voice mail Discussion boards Scheduling Calendaring	Different timing (asynchronous)

Interaction timing

Interaction locale

Same place Different place

Figure 7-3. Groupware classification by degree of separation in space and time.[73]

Groupware are software suites designed to be used as tools to enhance group work. In the broadest sense this mean that groupware is any computer software product intended to enhance the ability of coworkers to work together. Groupware is an implementation of the concepts and methods of Computer Supported Cooperative Work (CSCW), which is the study computerized systems that enhance the work of groups of people.

If the offices in a distributed organization are located on the opposite sides of the earth, the time difference will make it hard to communicate directly since they will not be at work at the same time if they work normal hours. The tools used for communication will then need to support asynchronous exchange of information.

In a distributed organization having the offices distributed around the globe can also be a great advantage. If an organization had an office in England (UTC 0), one in California (UTC -8) and one somewhere in western Australia (UTC +8) then work could be done 24 hours a day if each office worked on the same paper for 8 hours each. In this kind of setting asynchronous working hours in a distributed organization works out perfectly.

In this project's setting however, where we are communicating about configuration issues, synchronous information exchange would be preferable. In a setting where short questions and answers need to be communicated back and forth, the lack of synchronous/simultaneous communication is a huge drawback. When I arrived at work I would have the answers to the questions I asked yesterday waiting for me. If the solution didn't work, or another problem becomes apparent I then have to ask for help/input and wait till the next day before I can get an answer. When configuring a system answers should be found as soon as possible, because in this setting tomorrow morning seems ages away.

Normal working hours are 08.00 to 16.00, but since Norway is in the UTC +1 time zone and California in USA is in the UTC -8 time zone, we will never be at work at the same time during normal working hours. This means we have to make use of tools that support asynchronous communication, and since we are not at the same location on the face of the Earth, the tools are limited to those listed in figure 7-3 as tools for asynchronous/different place communication.

E-mail was the tool used for communication, this tool has some advantages and some shortcomings.

Pros:

- Enables asynchronous communication (e-mail waiting on me when I log on in the morning)
- Possible to send carbon copies to people related to the project
- Can send attachments (configuration files, documentation, models)
- Text is good when representing user names, passwords and other configuration values (less error prone to see textual values than trying to describe basic values in a voice mail)

Cons:

- Reply will not come till next morning at best, or it might not come at all. The receiver doesn't have to answer at once (or at all), he/she can delay it to a later time when it is more convenient or maybe forget to reply.
- If a message is lost in transmission the sender will not be notified at once (but after 3 days) that the message could not be transmitted to the receiver.
- If a message is not fully understood then there is no way of asking for clarification instantly. Since text doesn't contain body language or the tones of a voice, it is easier to misunderstand what is being communicated.
- The person that responds may not always reply to all the questions in an e-mail, only some.
- Passive form of communication
- Insecure form of communication (unless the message is encrypted)

In the beginning I made use of e-mail when communicating with the National Library of Norway, but I found that it was much faster to get an answer if I used the telephone. When I used telephone I was often told if the person I wanted to contact was away (on a conference or vacation), or ill that day. The telephone is an active form of communication that demands attention and answers right away, but this is not an option when only asynchronous communication can be used.

E-mails have also been used in the communication with the local IDI IT department. A quick answer is much easier and faster achieved by just paying them a visit and talking directly. Making use of the strong resources that dwell in the local IT department when solving problems proved to be illuminating and time saving.

The configuration files for the collections are currently being written by the developers at the UCSB, therefore communication is a vital part of getting a collection configuration up and running. The communication needs to be a distributed cooperation between the developers that are going to write the configuration files and the local administrator that wants to add a collection to a local node. Before the developers start to write the configuration for a collection they should have information about what software and software-versions that are used at the local node, so that they

don't make use of functions/methods that are not supported by the versions of the software that is used at the local node.

In a global distributed digital library network it is important to notify all local nodes when major changes are done to the network that will affect the local nodes. Communication within a distributed system should be proactive (inform nodes that an update that may affect them is upcoming) and not reactive (answer why suddenly things don't work anymore).

Another important part of communication is response times, and therefore an organization should have a policy for their help desk that sets a maximum time limit for responses. In configuration and other issues people often depend on the help desk of an organization in order to advance in their work, so answers should perhaps be given within 24 hours of when it was received (if it was received on a normal workday, or monday if the questions were received on friday).

7. 1. 5 Standardization of configuration file documentation

In large software systems that make use of several different configuration files it is important that the documentation of the configuration is tidy and easy to understand. All relevant information regarding a software system's configuration files should be made available at one location, piecing together information from various sites should not be necessary.

In large software systems it is feasible to make a standard for describing the configuration files, and to adhere to it. A standardization can make it easier for a new user of the system to understand how the configuration of the system is done (what file is used to configure what functionality).

Table 7-1: Configuration file documentation

Metadata for a configuration file	
Purpose	Short description of what the file configures
Standards/Formats	List different standard and formats that are used in the configuration file
Required	Is file required or optional in the configuration of the system? (yes no)
Resources	- URI to online examples of the configuration file - URI to DTD for XML files
Dependencies	Is the values in this file used in other configuration files? (Exports:filename) Does the file import values from other configuration files? (Imports:filename)
Notes	Special notes for configuration file
File URI	where the file is located in the file hierarchy

Table 7-1 shows an example of metadata that can be used to give a conceptual understanding and documentation of a configuration file. An example of how the ADL configuration files can be described in this format can be found in appendix C.

7. 1. 6 High cohesion and low coupling in configuration files

High cohesion and low coupling are design principals used in object oriented software design, but I think it is important to use these ideas on the design of configuration files also. This could make it easier to maintain and understand configuration files. If only the values that needed to be edited and changed were displayed to the local administrator, then the understanding and maintenance would be easier.

Using the principals of high cohesion and low coupling, the configuration files should be made minimal or with no dependencies between any of the configuration files. A configuration file should not import values from or export values to other configuration files optionally.

7. 1. 7 A graphical user interface (GUI)

A configuration tool with a graphical user interface (GUI) can be used to add a **layer of abstraction** between the configuration files of the digital library and the local administrator. A local administrator does not need to know what standards or formats the configuration files are using as long as he enters the values needed for the configuration, the configuration tool can add the values to the configuration files in the correct standards and formats. This layer of abstraction also hides the organization of the different configuration files in the file hierarchy. An administrator does not need to know where the values are stored as long as he is able to configure the middleware correctly.

If a GUI is used for configuration management it is possible to handle dependencies between configuration files more elegantly. **A value should only be added once** and automatically updated in every file that use it. If a GUI is used, then the administrator doing the configuration of the ADL node doesn't need to be aware of the dependencies between the configuration files. When configuration program with a GUI can be used for the configuration of the ADL middleware, then all the configuration can be done in **one place** without knowing where in the file hierarchy the different configuration files are stored.

In some of the configuration files, all that needs to be edited is one or two values, while the configuration file is several pages long because it also contains other default values used by the middleware. **Only the values that need to be edited should be displayed** to the local administrator, this will make the system less complex and easier to understand. A GUI can be used to only display the needed values for the configuration of the system. KISS (Keep It Simple Stupid) is a rule of thumb in the designer world, keep it simple and don't add unneeded complexity. It is important that both the design of the graphical user interface and the help system of digital libraries are designed for the correct user groups. Hiding unneeded complexity and providing the functionality needed without confusing the users are important.

A configuration program with a GUI can provide both a **simple and advanced mode** for configuration. The simple mode only displays and makes it possible to edit the values that are required, hiding unneeded complexity makes the system easier to comprehend. An advanced mode for expert users can make it possible to edit all configuration values. The GUI should aid a user in making the decisions needed to correctly configure the middleware by providing information and examples of how configuration can be done.

A GUI for configuration of the ADL node should make the necessary information available to the local administrator so he/she can make the right decisions when doing the configuration of the ADL node. A well developed help system should be part of the GUI configuration program, so help can be provided. As part of the help system there could be a tool that uploads error messages from the middleware and sends the error messages, along with which software versions (Ant, Tomcat, Java and middleware) that are installed on that ADL node, as an e-mail to the developers at UCSB. The help system should also provide examples of implementations of the different configuration files, or links to online documentation.

7. 2 Special recommendations

*“I don't have a solution
but I admire the problem.”
- Unknown*

7. 2. 1 Possible solution for configuration of the ADL middleware

I made a Java based mockup in figure 7-4 to display how the configuration process can be made easier. The value *fenris.idi.ntnu.no* is needed for two configuration files but only needs to be entered once. The user doesn't need to know which formats and standards are used in the configuration files, only the values needing to be entered. Then the configuration program can place the values at the right places and in the right standards and formats into the correct configuration files.

The values collected in the GUI would have been stored in three different configuration files (domain.js, webclient.conf, and web.xml) that use three different formats (javascript file, java property file, xml file).

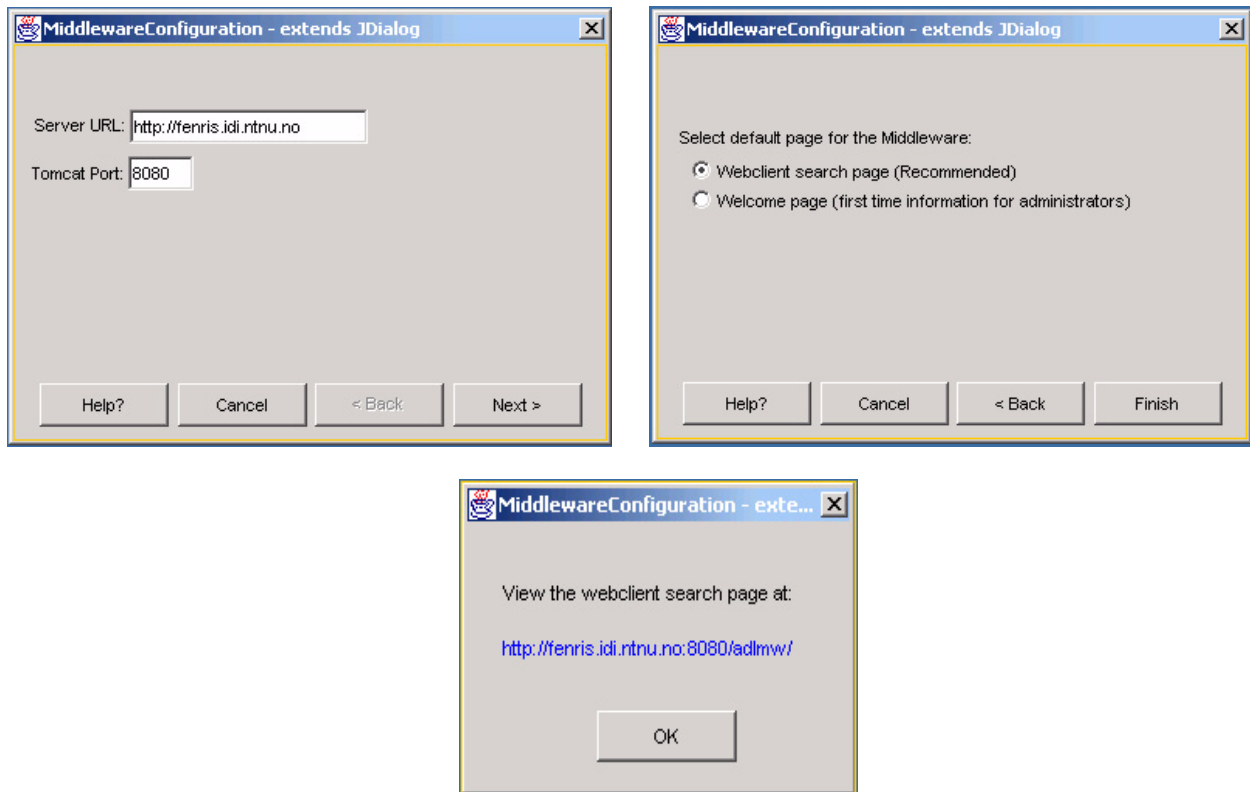


Figure 7-4. A mockup showing how configuration of the middleware can be done in a GUI

In the mockup in figure 7-4 above only the minimum information needed must be entered to get the ADL node up and running. The configuration program should handle the storing of duplicate values into different files, so a user only has to enter a value once. If a graphical user interface (GUI) is used for the configuration of the middleware, then the administrator does not need to know the location of the different configuration files, or in what files the different configuration values are stored (the configuration program will deal with that part). Both the “web.xml” and “webclient.conf” files are huge configuration files where only one value needs to be entered or edited. The mockup GUI hides all the default values in those files, only the values that need to be edited are displayed to the administrator.

The server URL, Tomcat port number and *adlmw* is concatenated into **http://fenris.idi.ntnu.no:8080/adlmw/** and this value is stored in the “webclient.conf” file.

http:// and *www.* is stripped from the server URL value and **fenris.idi.ntnu.no** is stored as domain in the “domain.js” file.

A radio box can be used to select which web page that should be displayed when going to the URL of the middleware webclient. A radio box select is much easier and less error prone than searching for and editing a value in a large xml file directly. The correct value will be stored in the “web.xml” file. When “*Finish*” is pressed, then the middleware servlet must be restarted at the Tomcat servlet engine for changes to take effect.

It is possible to make GUIs for the other configuration files of the ADL middleware as well, these should maybe be part of the configuration GUI for collections since they are more linked to that aspect of the configuration. Each time a new collection is added then the “collection_opml.xml” file also needs to be updated so the collection becomes available in the webclient interface. The news about a new collection becoming available can be entered in the “frame_blank.html” file so it will be displayed in the webclient.

7. 2. 2 Possible solution for adding local collections to a local node

Today the configuration and adding of collections to a local ADL node is far from trivial, mainly because the documentation is not written with the local administrator user group in mind, and that the local administrator has to be an expert in the different formats and standards that occur in the configuration files.

A new transparent layer of abstraction needs to be added so that only the values, logic, and concepts need to be configured by a local administrator. Once the bucket framework concept is understood by a local administrator it should be possible for him to add new local collections.

A configuration tool with a GUI can be used to add this layer of abstraction to make the configuration and adding of local collections simpler. The configuration should be done at a conceptual level instead of having to write programming code in Jython to implement the buckets and do the configuration. A local administrator should only need to understand the bucket framework concept, it should not be required that he needs to implement the buckets in Jython code.

The implementation is done to map the ADL buckets to metadata in a database, so basically a solution is needed that can map between a database and the ADL buckets, and auto-generate the code needed for the configuration files. There already exists software that can do mapping between databases, XML, and flat files and generate code in different programming languages to support the mapping between the different standards. **MapForce™** developed by Altova is such a mapping tool, more information about MapForce™ can be found at [74]. The mapping between different formats is done visually and the code needed for the mapping between the different formats is auto-generated.

MapForce™ 2005 supports mapping of XML, database, flat file, and Electronic Data Interchange (EDI) sources to XML, databases, and flat files, enabling the programmatic data transactions.

Developing a solution that resembles the MapForce™ program for configuration of collections would make it possible to work visually with mapping between the metadata and the ADL buckets, and thus only the values and logic need to be filled in to do the configuration. A local administrator would only need to understand the concept of the bucket framework to do the mapping between the database and the buckets. The implementation and coding needed in the configuration files would be handled by the configuration tool so the local administrator would not need to work in code.

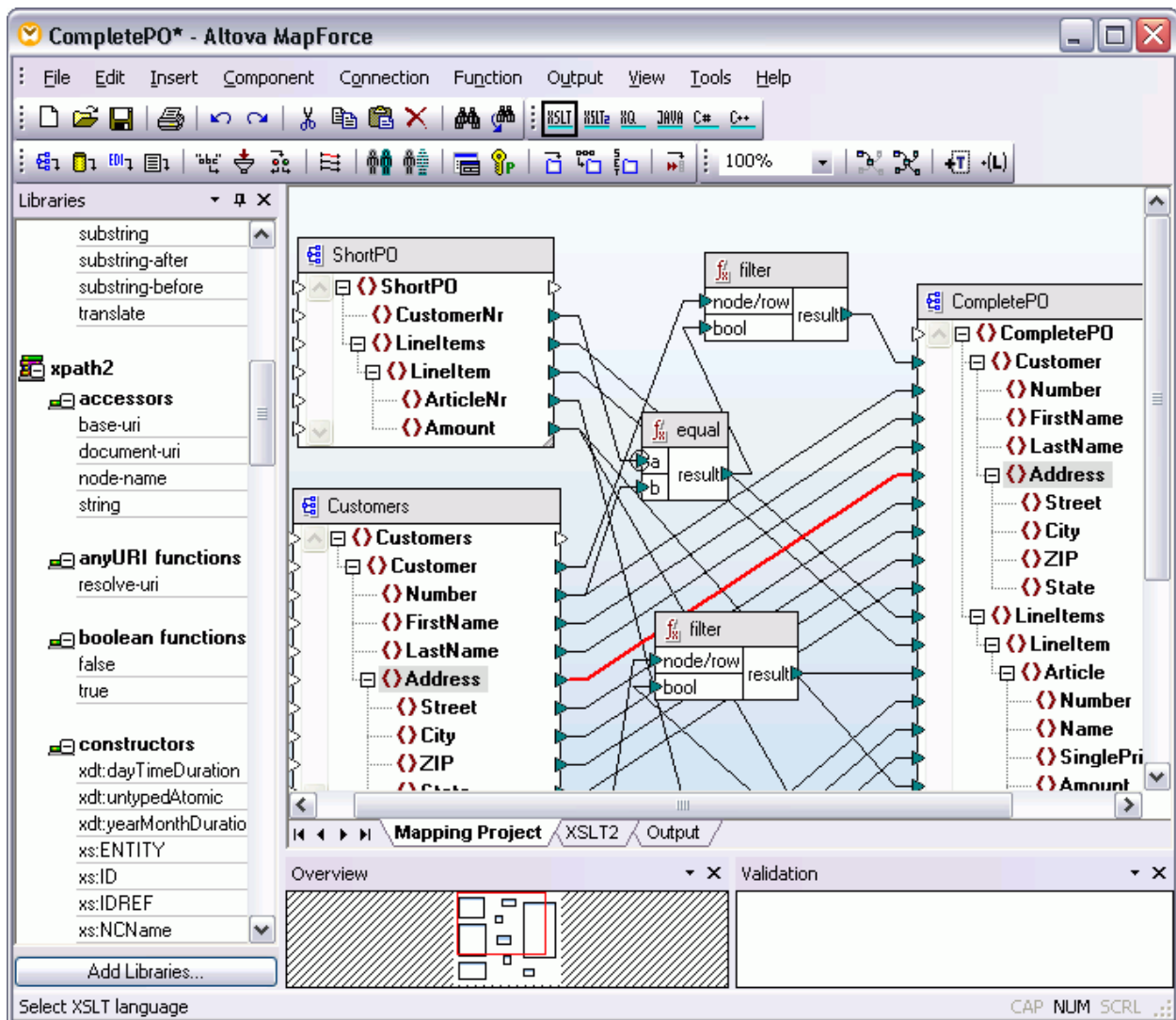


Figure 7-5. MapForce™ - Mapping between a database and XML [74]

Figure 7-5 shows how a mapping between a database and xml can be done. It should be possible to do the same mapping between a database and the nine standard buckets in ADL. The configuration program can then auto-generate the code needed for the configuration files in order to do the configuration and adding of a collection to an ADL node.

It is important to hide the complexity of coding and all the various standards and formats used in the configuration files in order to work at a conceptual level of configuration. A local administrator should not need to share a developers complete understanding of a system in order to install a node in a distributed digital library and add a collection to it. Adding a configuration tool with a GUI à la MapForce™ will add a transparent layer of abstraction between the local administrator and the configuration files and hide both the complexity of coding and the various formats and standards. The terminology and information presented to the local administrator in the configuration tool also have to be written with this user group in mind.

If the development of a configuration tool leads to local administrators being able to do the configuration all by themselves, there will no longer be any human limiting factor for distribution of the ADL and adding of new collections. And it will become easier to install and utilize the digital library.

7. 2. 3 Possible solution for the sharing of local collections



Figure 7-6. A mockup showing an alternative way to configure the RMI server

Figure 7-6 shows a mockup I made to illustrate a few points. The only two values in the “rmi-server.conf” file that need to be edited are the boolean value of create_registry and the port number the RMI register will be assigned to, the rest of the file contains default values that aren’t important for an administrator at a local ADL node.

The configuration program can also validate that the port number is an integer value between 1024-65535 (port numbers between 0-1023 are reserved). The local administrator doesn’t have to type yes/no/YES/NO/Yes/No but can click an option for activation of the RMI server. This is a less error prone method and more user-friendly way to do the configuration.

To the mockup I also added the possibility to select which collections to share with the distributed network, though this is not currently supported. There can be collections at a local ADL node that an administrator wishes should remain only locally available, while he wishes to share the rest of the collections with the distributed network.

7. 2. 4 Summary of the different solutions

Documentation and software configuration management umbrella activity should be an integrated part of every large software system development. Digital libraries are large complex software systems so I will not stress the importance of documentation and SCM here since it should be obvious.

Different tools for communication within distributed organization can have an impact on the communication in that organization, as each tool (for example e-mail) has different strengths and limitations. Interaction locale and interaction timing will affect what tools can be used for communication in a distributed organization. Defined rules for maximum response-time within a distributed organization can improve the communication. In a configuration paradigm, synchronous communication is preferred.

Different solutions of communication across firewalls can have different problems associated with them, in example VPN (Virtual Private Network) solutions should be avoided since this technology doesn't scale well. SSH (Secure SHell) tunneling and opening for communication between two IP addresses scale well. SSH tunneling is the most secure method of those two.

Three different solutions that can improve the ADL from a local administrator's point of view are:

- Standardization of configuration files
- High cohesion and low coupling in configuration files
- Making a GUI (Graphical User Interface) based tool for configuration

Each solution addresses one or more problem areas, but which solution is best and why?

Standardization of configuration files

Gathering the information needed to understand a large software system in one location, and describing it in a standardized way can make it easier to comprehend for new users. It can become easier to document a system when a standard is used for the documentation, because it is then known what features that should be documented.

A standardized description of the configuration files can make the documentation of a software system more understandable, but the configuration files will still have to be edited by hand. This solution will not hide the complexity of the file hierarchy or the default values in the configuration files. This solution offers no validation of entered values nor possibilities to test the configuration. The local administrator will still have to implement all the configuration files. The local administrator will thus need to be an expert in the different standards and formats used, and have a very good understanding of the system in order to make a working configuration.

High cohesion and low coupling in configuration files

A reorganization of the configuration files would make it possible to only display values that need to be edited in configuration files, and maybe make a more logical organization of the configuration files where all can be located at one location in the file hierarchy.

In ADL, a reorganization of the configuration files would most likely require a rewriting of large parts of the middleware, and hence be costly and time consuming. On the other side, this solution can make the configuration of a system easier to understand by having all configuration files in one location and only displaying values that need to be edited. Lower coupling between configuration files will remove dependencies that make a system hard to understand. Files with high cohesion should logically be easier to understand than files with low cohesion.

This solution offers no validation of entered values nor possibilities to test the configuration. A local administrator will still have to edit the configuration files by hand and this can be an error prone and not very user-friendly process. Also this solution suffers from the problems associated with having to implement the configuration files (expert knowledge of standards/formats and the system is needed in order to make a working configuration).

Making a GUI (Graphical User Interface) based tool for configuration

A GUI based configuration tool will render the standardization of configuration files and high cohesion/low coupling solutions obsolete.

A GUI can hide all default values that are of no interest to a local administrator and only display values that need to be entered or edited. This can make it easier to understand the system by making it less complex. When a GUI is used, the location of the different configuration files in the file hierarchy becomes irrelevant for the local administrator, and as long as all configuration can be done through a GUI based configuration tool, the local administrator does not need to know what happens behind the curtains (where values are stored or in which formats/standards).

What makes the GUI solution the **preferred solution** is the fact that a tool can act as a layer of abstraction between the local administrator and the complexities of standards, formats, and the ADL system. A local administrator will still need to understand the concepts of the system (the concept of the bucket framework is essential in order to map between database schemes and the bucket framework) in order to do the configuration, but the local administrator will no longer need to implement the configuration files by hand. Configuration can be done at a conceptual level without having to work in code or with all the different standards. The code will be auto-generated and stored in the appropriate configuration files by the configuration tool, the local administrator will only add the values needed for the configuration.

A GUI can provide interactive help features through documentation, examples, validation of entered values (check if all needed values are entered and correctly formatted), and test features to validate if a configuration is working properly.

The ADL node is already dependent on Java, so developing a configuration tool with a GUI in Java will add no new dependencies for the ADL middleware. If the configuration tool is Java based it can run on any platform that has the Java runtime environment installed without needing to be recompiled for new platforms, just compile once and run it anywhere.

Test programs and routines that can verify and test the configuration of the different collections along with the ADL node can also be implemented in a configuration program. If that is done then it will become easier for local administrators to test and validate their configuration.

The configuration tool can make the configuration of ADL and collections easier to understand, more efficient, less error prone (validation of entered values and default values can't be touched), lower the requirements of skills needed to make a working configuration (no longer need to be an expert in all the different standards/formats, but can work at a conceptual level), and save money (through less errors). If all configuration can be done through a configuration tool then the local administrator doesn't need all detailed documentation about location of the different configura-

tion files, which formats/standards are used in them, or which values need to be edited and what are the default values. The documentation for the local administrator needs to address the configuration tool and needed knowledge for conceptual understanding of the system. It can be more like a *cookbook* for doing the configuration of the middleware and management of collections.

7.3 Further work

*“You can't wait for inspiration.
You have to go after it with a club.”
- Jack London*

It is obvious that hiding complexity and documenting for the proper user groups will increase the user-friendliness and thus the usability of distributed digital libraries. Lowering the requirement for skills needed in order to install, configure, and deploy a local node with local collections in a distributed digital library network can increase the number of nodes in the distributed digital library.

Making it easier to employ digital library technology can in turn contribute to increase the spread of digital libraries. If a digital library is very user-friendly (easy to install, configure, and deploy) then maybe more organizations and individuals will use them for publishing and maintaining (personal) collections and share the collections with the masses.

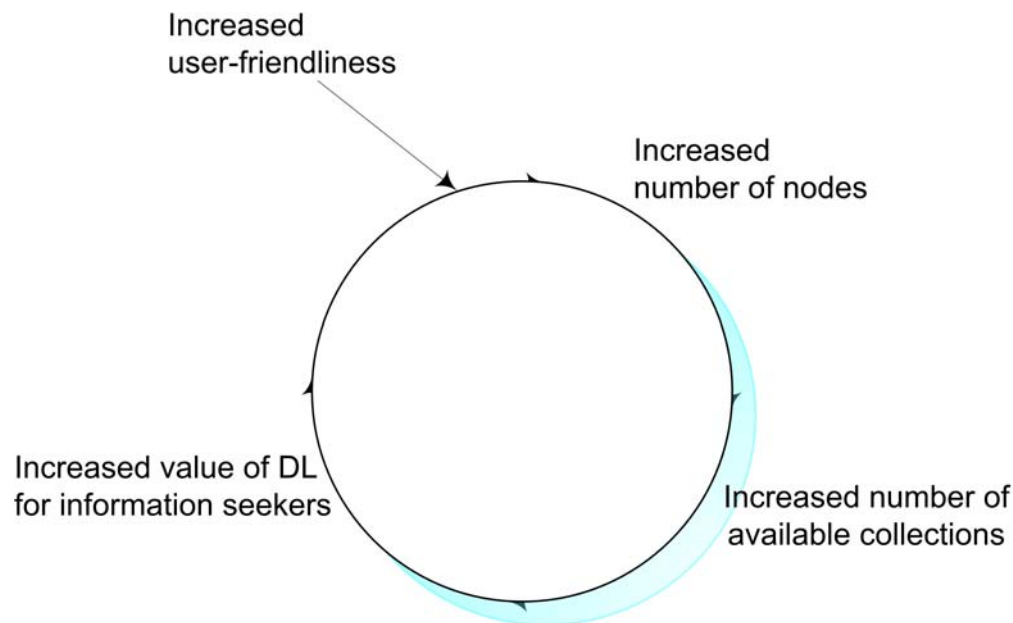


Figure 7-7. Distributed digital library growth

Increased user-friendliness can increase the spread of a distributed digital library system, and more local nodes will be added to the distributed digital library as more people manage to do the configuration of the node and adding of local collections. The number of available collections can increase the value of a distributed digital library for information seekers. The bigger diversity

(both in depth and breadth) a global distributed digital library has, the more likely it is that a person can satisfy his or her information need by using that global distributed digital library network. As more people and organizations become aware of the digital library, maybe they will make new nodes.

Of the different digital libraries that were compared in chapter 3, only ADL is a distributed digital library that is actively trying to get more organizations and private persons to embrace their technology and enlarge their distributed digital library network by adding more digital library nodes. In order to make ADL a global distributed digital library, other organizations have to embrace the ADL technology and install local nodes with local or remote collections. No single organization should control all the ADL nodes and the collections, it should be a distributed responsibility where each organization is responsible for their own local ADL node. The ADL developers are developing a framework that makes such a vision possible. Increasing the user-friendliness of ADL can be a stepstone making it possible for more organizations to set up an ADL node.

The easiest way to increase the user-friendliness of ADL will be to develop a configuration tool for configuring the middleware and management (adding, removing, and maintaining) of collections. It should be possible for local administrators to work at a conceptual level of configuration where only logic and needed values need to be entered instead of having to implement the entire bucket framework.

7.3.1 Making the GUI based configuration tool

It is feasible to develop a configuration tool to aid in the configuration of the ADL middleware and management of collections for an ADL node, since it will make the configuration process more user-friendly and easier to understand through hiding unneeded complexity. A configuration tool will also make the configuration less error prone since a local administrator can use a configuration tool to verify that all needed values are added and that they are in the correct format. Making the configuration less error prone and more user-friendly will increase the effectivity of the local administrators; fewer mistakes will be made so less time is needed for debugging, and hiding complexity means a local administrator doesn't have to spend so much time trying to understand what the system does.

Who should develop the configuration tool? The development of ADL at UCSB is ever-moving forward at a rapid pace, and to keep up with the development of the digital library the configuration tool should be developed at UCSB by the development team. If another organization is going to develop the configuration tool, then very good communication will be needed between the ADL developers and that organization. Every change that is made that would affect the configuration in some way, will need to be communicated so the configuration tool doesn't become obsolete before it is completed. The ADL developers have the complete understanding of the system needed to make a configuration tool, and they know first hand when changes occur and how that will affect the configuration tool.

Ideas for a configuration tool for ADL are listed below.

Develop a configuration tool that checks which versions of the needed software are installed on the machine that is going to run the middleware (Java, Ant, Tomcat, JDBC drivers, databases):

- List compatible software versions of the needed software in green colour
- List missing or outdated software versions of the needed software in red colour
- Aid in finding the correct version of the needed software (pointers to web pages)
- If a local administrator needs help with the configuration, a list of the installed software available to the middleware can be sent along with the help request to the developers, so the developers know which versions of software exists on that machine.

Other important aspects with a configuration tool are:

- A configuration tool should be able to show help and documentation (how entered values should be formatted, one or more online examples of configuration, help-files with terms to search for), to check if the entered values are correctly formatted, and if all needed values are entered.
- The mockups in “Special recommendations” can serve as templates when implementing a configuration tool for ADL

A possible use case that can be used for the development of a configuration tool is listed in appendix A.

“A use case is a collection of related success and failure scenarios that describe actors using a system to support a goal.” [75]

It is important to focus on how using the system can provide observable value to the user, or fulfill their goals. It will be important to develop and **write documentation for the configuration tool with the local administrator user group in mind**. Testing will be important to remove as many bugs as possible and make a user-friendly GUI. The configuration tool should make it possible for local administrators to become completely independent in the configuration process, not having to rely on others in order to do the configuration.

7.3.2 Bulletin board/Discussion board

Bulletin boards can become an important part of the documentation of a system and can be a media of communication between the developers and the user groups. If problems that are reoccurring are posted on a bulletin board then users can be directed there when they encounter that problem. This will save the staff valuable time that they can spend on other issues than to answer the same questions over and over. The bulletin board will become a resource for problem-solving that users can make use of when they encounter trouble. The bulletin board site can function as a help desk and be used for problem-solving and communication between the developers and the community using the system. Customer support is an important part of a system, it also is a good source of information for learning about bugs and what new features customers want the system to support.

In online documentation it is important to make hyperlinks from the installation and configuration guide to the resources that can be used to solve problems that can occur. If the documentation of features or problems can't be found, then it is not useful at all. An undocumented feature might as

well not exist to most users because *no one* knows about it. A user has to learn about a feature from somewhere to learn how to use it (although it is possible to discover features by accidents, these “features” are sometimes referred to as bugs).

7.3.3 A collection discovery service for ADL

A collection discovery service should be developed for the Alexandria Digital Library. The discovery service should query nodes in the ADL distributed network, and list the collections made available from each node along with metadata (for example description and copyright) for each collection.

The configuration parameters needed to add a remote collection to a node should be easily found in the collection discovery service. This would make it easier for a local administrator to know what collections are available in the distributed network, and find information about how to set them up as remote collections.

The collection discovery service should query each node in the distributed network at intervals and add shared collections along with the collection description/metadata automatically to its index.

Information seekers can also use a collection discovery service to see descriptions of shared collections and where those collections can be found. Private collections should also have short metadata descriptions, making it possible to use the discovery service as a description and linking service to find nodes to search for information. Automatic cataloguing of different themes is a possible way to order the collection descriptions.

8 Conclusion

*“A conclusion is simply the place
where you got tired of thinking.”
- Unknown*

The conclusions of this thesis have been reached through an investigation of theory concerning distributed georeferenced digital libraries and an investigation of problems and solutions associated with the pilot installation of the NTNU ADL node with Galleri Nor as the local collection.

In chapter 1.1 a summary of the problem statement was made through four questions that this thesis sought to address and find answers to. The main lessons from this thesis are the following.

1. *What characterizes a georeferenced digital library compared to other digital libraries?*

One main difference between georeferenced digital libraries and digital libraries in general, is that georeferenced digital libraries provide new access points to collections where spatial literacy can be used to solve problems and satisfy information needs.

The spatial literacy paradigm provides new ways of problem solving, and information objects in collections need geospatial metadata (longitude/latitude coordinates and placenames) in order to support the new access methods.

Gazetteers need to be an important part of a georeferenced digital library to support informal specification of location through the use of placenames.

2. *When adding a local node of a georeferenced distributed digital library with local collections, how do the local collections become visible and searchable for the whole system?*

A local collection at a local node becomes visible for the distributed digital library network by sharing it. By enabling the RMI (Remote Method Invocation) server at the local Alexandria Digital Library node, all local collections are shared with the rest of the distributed network. At present there is no operational collection discovery service for the ADL distributed network, so the ADL developers have to be notified by e-mail when new collections have been made available.

There are two ways to search collections from other ADL nodes, either configuring an RMI collection, or dynamically accessing the collection.

The development of a collection discovery service would make it easier to find potential collections an ADL node can add as remote collections. Information seekers could also use a collection discovery service to find nodes that might host relevant collections. The development of a collection discovery service would increase the visibility of shared collections in the ADL

distributed network, making it easier to make shared collections visible to the rest of the network.

3. *What problems arise when configuring georeferenced distributed digital libraries, and how can these problems be solved in an efficient and user-friendly way?*

The problems that did arise were not specific to georeferenced systems, but applies to all digital library systems with distributed nodes.

Digital libraries are characterized by the use of many different standards, formats, and technologies. A local administrator should not need to be an expert in all the different technologies, formats, and standards used in order to install and configure a distributed digital library node or add a collection to it. Several possible solutions are discussed in chapter 7. The development of a GUI (Graphical User Interface) based configuration tool seems best to address the problems related to the configuration process.

When developing digital library systems and making the documentation, it is crucial to identify the user groups of the system. Hiding complexity by only showing the information needed by the specified user group is also important. Documentation and screen design must be made with a complexity and a terminology that can easily be understood by the user groups of that system.

Just as the bucket framework in ADL adds a layer of abstraction between the middleware and heterogeneous collections, a configuration tool can add a layer of abstraction between the middleware and a local administrator. The bucket framework makes all the heterogeneous collections appear as homogeneous collections since the middleware searches the buckets instead of the various metadata standards used in the different collections. A configuration tool can add a layer of abstraction that hides the complexity of all the different standards and formats used in the configuration files, making it possible to work at a conceptual level of configuration instead of needing to be an expert in all the various formats in order to implement buckets for making a configuration.

4. *Is it feasible to develop tools to aid in the configuration of digital library systems? What can be achieved with operational tools, when available?*

It is feasible to develop configuration tools, and configuration tools should be developed to address the complexities associated with the configuration of digital libraries. The Java programming language can be used to develop a GUI based configuration tool for ADL, since Java is already a prerequisite for an ADL node.

A GUI based configuration tool should be developed to aid in the spread of the ADL as a global distributed digital library system. A configuration tool will hide unnecessary complexity and thus make digital libraries easier to configure, making them more understandable, less error prone, and thus more efficient. More efficient configuration, less bugs and errors will increase the spread of digital libraries, which in turn will save time and money for the organi-

zations that install digital library nodes. All these factors make it beneficial and feasible to develop a configuration tool.

The roles of the local administrator and the software developer are overlapping in centralized digital libraries. When the nodes are distributed the division between the two different roles becomes apparent.

The Alexandria Digital Library is a possible framework for a global georeferenced distributed digital library. Of the digital libraries compared in the *State of the Art* chapter, the ADL architecture is the only one that is built to become a global georeferenced distributed digital library with distributed digital library nodes with local collections that are interconnected.

In hindsight the prerequisites for my work have been a blessing. Using a collection from a collection provider instead of a local mockup collection made for testing purposes has illuminated problems related to the relationship between a digital library node and collection providers. These problems would have remained hidden if a local mockup had been used. ADL makes it possible to install a local node that becomes part of the distributed digital library network, while most other digital libraries only allow local collections to be added to their digital library. The value of a configuration tool becomes apparent in a global distributed digital library architecture to increase the number of nodes in the distributed network. Usability (ease of use and usefulness) for the local administrator user group has an impact on the spread of a global distributed digital library. Just like a *State of the Art* space ship will never lift off the ground if no one can operate it, a global distributed digital library network will not expand if no one manages to add new nodes.

9 References

"Come up with a really great quote,
and your name will live forever."
- Anonymous

- [1] Trond Aalberg . *Supporting Relationships in Digital Libraries*. Ph. D. Thesis. Department of Computer and Information Science. Norwegian University of Science and Technology. 2003. IDI-rapport 3/03. URN:NBN:no-3403.
- [2] Christine L. Borgman, M.J. Bates, M.V. Cloonan, E.N. Efthimiadis, A. Gilliland-Swetland, Y. Kafai, G.L. Leazer, and A. Maddox. *Social Aspects of Digital Libraries*. Final Report to the National Science Foundation, 1996.
- [3] IDI . Last checked 13. October 2004. <http://www.idi.ntnu.no/grupper/if/research.html#interests>
- [4] Digital Libraries : *Future Research Directions for a European Research Programme*. Workshop report 02/W02. ERCIM. 2002.
- [5] Christine L. Borgman. *From Gutenberg to the Global Information Infrastructure : Access to Information in the Networked World*. Digital Libraries and Electronic Publishing. The MIT Press. 2000.
- [6] Online Dictionary for Library and Information Science. Last checked on 14. October 2004. <http://lu.com/odlis/>
- [7] Arne Sølvsberg, and David Chenho Kung. *Information System Engineering. An Intorduction*. Springer-Verlag. ISBN 3-540-56310-5
- [8] Norbert Fuhr, Preben Hansen, Michael Mabe, Andras Micsik, and Ingeborg Sølvsberg. Digital Libraries: *A generic classification and evaluation scheme. Proc. 5th European Conference on Research and Advanced Technology for Digital Libraries (ECDL2001)*. Lecture Notes in Computer Science Vol.2163. Springer Verlag. 2001. pp.187-199. <http://www.springerlink.de/link.asp?id=w22qmkk09ncy59av>
- [9] Adam Nabil R., and Yesha Yelena. *Introduction*. International Journal on Digital Libraries, 1(1). 1997.
- [10] Stephen M. Griffin. *Taking the Initiative for Digital Libraries*. The Electronic Library. 16(1):24–27. February 1998. Interview.
- [11] American National Standards Institute. *Serial Item and Contribution Identifier(SICI)*. ANSI/NISO standard Z39.56-1996 (Version 2). August 1996.

- [12] M. Mealling, P. Leach, and R. Salz. *A UUID URN Namespace*. Internet draft, Internet Engineering Task Force. October 2002. Work in progress.
- [13] American National Standards Institute. *Syntax for the Digital Object Identifier*. ANSI/NISO standard Z39.84-2000. May 2000.
- [14] T. Berners-Lee, R. Fielding, U.C. Irvine, and L. Masinter. *Uniform Resource Identifiers (URI) : Generic Syntax*. RFC 2396. The Internet Engineering Task Force. August 1998.
- [15] Øyvind Vestavik. *REAP Et system for rettighetsstyring i digitale bibliotek*. Cand scient thesis. Institutt for datateknikk og informasjonsvitenskap. NTNU. Sept. 2002. <http://www.idi.ntnu.no/~oyvindve/MasterThesis.pdf>
- [16] Lorcan Dempsey, and Rachel Heery. *A Review of Metadata : A Survey of Current Resource Description Formats*. 1997. Available online at <http://www.ukoln.ac.uk/metadata/DESIRE/overview/>
- [17] MARC . Last checked 26. October 2004 . <http://www.loc.gov/marc/>
- [18] Dublin Core . Last checked: 26. October 2004 . <http://dublincore.org/documents/dces/>
- [19] Anne J. Gilliland-Swetland. *Defining Metadata. Introduction to Metadata: Pathways to Digital Information*. Los Angeles. Getty Information Institute. 1998. Also available at: http://www.getty.edu/research/conducting_research/standards/intrometadata/2_articles/index.html
- [20] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems : Concepts and Design*. Addison-Wesley, 1998.
- [21] Carnegie Mellon Software Engineering Institute. *Software Technology Review : Middleware*, January 1997. (search for “Carnegie Mellon Middleware”) Last Checked 28. October 2004 <http://www.sei.cmu.edu/str/>
- [22] F. Cairncorss. *The Death of Distance: How the Communication Revolution is Changing Our Lives*. Boston. MA: Harvard Business School Press. 1997.
- [23] Economist. *The revenge of geography*. Economist March 15. 2003.
- [24] Linda Hill. *Introduction to Georeferencing in Digital Libraries: Tutorial Document*. Santa Barbara: University of California, Santa Barbara. August 2003. Contact the author at lhill@alexandria.ucsb.edu
- [25] Greg Janée, and Linda Hill. *Gazetteer Protocol*. Alexandria Digital Library Project. 2002, April 24, 2003. Available: <http://www.alexandria.ucsb.edu/gazatteer/protocol/> [2003, May 2].

- [26] Greg Janée. *Issues in Georeferenced Digital Libraries*. Alexandria Digital Library Project. May, 2004 D-Lib Magazin Volume 10 Number 5. ISSN: 1082-9873 . Last checked 24. August 2004. <http://www.dlib.org/dlib/may04/janee/05janee.html>
- [27] C. Marley. *The changing profile of the map user*. In R. B. Parry & C. R. Perkins (Eds.), *The Map Library in the New Millennium*(pp.12-27). Chicago: American Library Association. (2001).
- [28] Nikon D2X . Last checked 5. October 2004. <http://www.europe-nikon.com/details.aspx?countryId=20&languageId=22&prodId=945&catId=91>
- [29] GPS Cord MC-35 . Last checked 5. October 2004. <http://www.europe-nikon.com/details.aspx?countryId=20&languageId=22&prodId=968&catId=92>
- [30] Coordinate Universal Time - Wikipedia, the free encyclopedia . Last checked 5. October 2004. <http://en.wikipedia.org/wiki/UTC>
- [31] K. E. Foote, and D. J. Huebner. *Database Concepts*. Geographer's Craft Project, University of Colorado, Boulder. (2000). Available: <http://www.colorado.edu/geography/gcraft/notes/datacon/datacon.html>
- [32] Columbia Electronic Encyclopedia. (6th edition)(2003). Available: Accessed through GURUNET: [http://www.gurunet.com/\[2003, May 17\]](http://www.gurunet.com/[2003, May 17]).
- [33] Peter H. Dana. *Geodic Datum Overview*. Geographer's Craft Project, University of Colorado at Boulder. (1999). Available: <http://www.colorado.edu/geography/gcraft/notes/datum/datum.html>
- [34] National Research Council Mapping Science Committee. *Distributed Geolibraries : Spatial Information Resources. Summary of a Workshop held June 15-16, 1998*. Washington, DC: National Academy Press. (1999).
- [35] GeoRef . Last checked: 1. November 2004. <http://library.dialog.com/bluesheets/html/bl0089.html>
- [36] Ray R. Larsen. *Geographic Information Retrieval and Spatial Browsing*. University of California, Berkeley. (1998). Available: http://sherlock.berkeley.edu/geo_ir/PART1.html
- [37] Michael F. Goodchild. *The Alexandria Digital Library Project : Review, Assessment, and Prospects*. D-Lib Magazine May 2004 Volume10 Number 5. ISSN 1082-9873
- [38] Statens Kartverk. Norgesglaset. Available at: <http://ngis2.statkart.no/norgesglaset/default.html>
- [39] Informedia, Available: <http://www.informedia.cs.cmu.edu/>

- [40] Harstad Municipality's map of Grytøya. Available at: <http://www.harstad.kommune.no/kart/kart-g.htm>
- [41] ESA . Available at: http://earth.esa.int/ers/ers_action/_norway_details.html
- [42] Greg Janée, and James Frew. *The ADEPT Digital Library Architecture*, ACM+IEEE Joint Conference on Digital Libraries. July 18th, 2002. Portland, Oregon, USA. Available: <http://alexandria.sdc.ucsb.edu/~gjane/archives/2002/jcdl-adept.doc>
- [43] Charles A. Cutter. *Rules for a Dictionary Catalogue : Special Report on Public Libraries*. Government Printing Office. 1904.
- [44] Primus . Last checked 20 August. 2004. <http://www.mdt.no/primus/>
- [45] James S. Reid, Chris Higgins, David Medyckyj-Scott, and Andrew Robson. *Spatial Data Infrastructures and Digital Libraries*. Paths of Convergence. D-Lib Magazine May 2004. Volume 10 Number 5 . ISSN 1082-9873. Last checked 25 August 2004. <http://www.dlib.org/dlib/may04/reid/05reid.html>
- [46] Go-Geo! . Last checked 20. August. 2004. <http://www.gogeo.ac.uk/>
- [47] Go-Geo! architecture. Last checked: 9. November 2004. <http://go-geo.data-archive.ac.uk/Architecture.doc>
- [48] GeoXwalk . Last checked 20. August. 2004. <http://hds.essex.ac.uk/geo-X-walk/>
- [49] Galleri Nor . Last checked 23 August. 2004. <http://www.nb.no/gallerinor/>
- [50] Marit Olsen. *Integrasjon og bruk av gazetteers og tesauri i digitale bibliotek Søk og gjenfinning via geografisk referert informasjon*. Cand scient thesis. Institutt for datateknikk og informasjonsvitenskap. NTNU. August. 2004. Available at: <http://www.idi.ntnu.no/grupper/if/publikasjoner/Hovedoppgave.Marit.pdf>
- [51] DLESE . Last checked: 3. November 2004. <http://www.dlese.org/dds/index.jsp>
- [52] Eirik Gjesteland, Håkon Heuch, Kjell Omdal Erichsen, Jens Olbergsveen, and Tore D. Saue. *ADEPT-NTNU*. NTNU. 2001. Available at: <http://www.idi.ntnu.no/grupper/if/publikasjoner/ADEPT-NTNU.pdf>
- [53] ADL Search Buckets: Description & Implementation. Last checked: 11. November 2004. <http://www.alexandria.ucsb.edu/research/docs/ADLDevGuide/Output/wwhelp/wwhimpl/java/html/wwhelp.htm?href=HTML-02-1.html>
- [54] Introduction to the Query Translation. Last checked 7 December 2004. <http://piru.alexandria.ucsb.edu:8888/opadl/95>

- [55] ADL Gazetteer. Last checked: 8. November 2004. <http://alexandria.sdc.ucsb.edu/gazetteer/>
- [56] Greg Janée, James Frew, Linda Hill, and Terence. R. Smith. *The ADL Bucket Framework*. (2001) Available at: <http://www.alexandria.ucsb.edu/~gjanee/archive/2001/delos-abstract.pdf>
- [57] Middleware Client Interfaces (for ADEPT). Last checked: 11. November 2004. <http://alexandria.ucsb.edu/middleware/doc/client-interfaces.html>
- [58] Jeanine Lilleng. *ADEPT NTNU, hvordan operasjonalisere node i digitalt bibliotek*. Cand scient thesis. Institutt for datateknikk og informasjonsvitenskap. NTNU. Sept. 2002. Available: <http://www.idi.ntnu.no/grupper/if/publikasjoner/Hovedoppgave.Jeanine.pdf>
- [59] Architectural diagram of ADEPT . Last checked: 15. November 2004. <http://alexandria.sdc.ucsb.edu/~gjanee/archive/2001/architecture-diagram.gif>
- [60] ADL Thesaurus protocol. Last checked: 17. November 2004. <http://www.alexandria.ucsb.edu/thesaurus/>
- [61] Metadata for the ADL Feature Type Thesaurus . Last checked: 19. November 2004. http://www.alexandria.ucsb.edu/gazetteer/FeatureTypes/FTT_metadata.htm
- [62] Guide to the ADL Gazetteer Content Standard version 3.2 . Last checked: 19. November 2004. <http://www.alexandria.ucsb.edu/gazetteer/ContentStandard/version3.2/GCS3.2-guide.htm>
- [63] ADL webclient. Last checked: 18. November 2004. <http://webclient.alexandria.ucsb.edu/mw/index.jsp>
- [64] Middleware Test Forms. Last checked: 18. November 2004. <http://webclient.alexandria.ucsb.edu/mw/test-form.jsp>
- [65] ADL FAQ. Last checked: 17. November 2004. <http://www.alexandria.ucsb.edu/research/docs/ADLDevGuide/Output/wwhelp/wwhimpl/java/html/wwhelp.htm?href=HTML-13-1.html>
- [66] MIL . Last checked: 22. November 2004. <http://webclient.alexandria.ucsb.edu/mw/index.jsp>
- [67] EIL . Last checked: 22. November 2004. http://eil.bren.ucsb.edu:8080/mw2_wc/example_aims.html
- [68] NZADL . Last checked: 22. November 2004. http://www.nzadl.org/mw_webclient/
- [69] ADL installation documentation. Last checked: 25. November 2004. <http://www.alexandria.ucsb.edu/confluence/display/adldoc/Install+Documentation+for+the+ADL+webclient>
- [70] Installation Documentation for the ADL middleware. Last checked 29. September 2004. <http://www.alexandria.ucsb.edu/research/docs/ADLDevGuide/Output/wwhelp/wwhimpl/common/html/wwhelp.htm?context=HTML&file=HTML-04-1.html>

[71] James Y.L. Thong, Weiyin Hong, and Kar Yan Tam. *What leads to acceptance of digital libraries?*. Communications of the ACM archive. Volume 47 , Issue 11 (November 2004) Pages: 78 - 83 . 2004 . ISSN:0001-0782

[72] G. M.Clemm. "Replacing Version Control with Job Control," Proc. 2nd Intl. Workshop on Software Configuration Management. ACM. Princeton. NJ. pp. 162-169. October 1989.

[73] Richard Bellaver, and John Lusa. *Knowledge management strategy and technology*. Artech House . ISBN 1580531059

[74] Mapforce - Altova. Last checked: 14. December 2004: http://www.altova.com/products_mapforce.html

[75] Craig Larman. *Applying UML and Patterns. An introduction to Object-Oriented Analysis and Design and the Unified Process*. Second Edition. ISBN 0-13-092569-1

Appendix A

A. 1 Use Case description for a configuration tool

A. 1. 1 Use Case UC1: Configuration of the ADL Middleware

Primary Actor: Local administrator (person that does the configuration)

Stakeholders and Interests:

- Local administrator: Wants fast, correct and accurate configuration of the ADL middleware, with help offered when problems arise. Help can be information made available.
- ADL system developers: Wants their system to become widely distributed and made use of.

Preconditions: The local administrator have started the configuration program.

Success Guarantee (Postconditions): The configuration of the Middleware is saved.

Main Success Scenario (or Basic Flow):

1. Local administrator enters configuration values for the Middleware
2. System verifies format of values and check that all required values are entered
3. Local administrator tells the system to store configuration values
4. System updates the configuration files with the new values, creating the files if they do not already exist.
5. System presents success message when all configuration values are stored.
6. Local administrator tells system to terminate
7. System shuts down with a goodbye message

Extensions (or Alternative Flows):

*a At any time the local administrator can tell the system to load default values

1. System restores default values and present them to the local administrator

*b At any time the system can encounter an error

1. System displays error message to local administrator (if possible)
2. Local administrator decides next action

1a. Local administrator do not know what values to enter and requests help from system

1. System presents the requested information (format, example file) to the local administrator
2. Local administrator enters the configuration value

2a. System detects that some (1 or more) of the required values are not entered

1. System displays error message to the local administrator and requests that the missing value(s) must be entered before proceeding
2. Local administrator enters the missing value(s)

3a. Local administrator aborts/cancels the configuration (this can happen at any time before step 3

1. Local administrator tells system to abort
2. System resets all values to what they were when program was started, and

display them to the local administrator

Special Requirements:

- The ADL middleware for the configuration files already exists and the files generated must be generated so they are compatible with it.

A. 1. 2 Use Case UC2: Configuring/adding and Management of Collections

Primary Actor: Local administrator (person that does the configuration)

Stakeholders and Interests:

- Local administrator: Wants fast, correct and accurate management (add, remove and update) of collections, with help offered when problems arise. Help can be information made available.
- ADL system developers: Wants their system to become widely distributed and made use of.
- Users seeking georeferenced information: Wants the node to become as functional and operational as possible.

Preconditions: The local administrator have started the configuration program.

Success Guarantee (Postconditions): The collection configuration is saved/updated.

Main Success Scenario (or Basic Flow):

1. Local administrator selects that he wants to add a collection
2. System displays options and the fields required to be filled out
3. Local administrator enters configuration values for the collection management
4. System validates formats of values and check if all required values are entered
5. Local administrator tells system to store the values
6. System updates the configuration files with the new values, creating the files if they do not already exist.
7. System presents success message when all configuration values are stored.
repeat 1-7 till wanted collections are all added
8. Local administrator tells system to terminate
9. System shuts down with a goodbye message

Extensions (or Alternative Flows):

*a At any time the system can encounter an error

1. System displays error message to the local administrator (if possible)
 2. Local administrator decides next action
- 1a. Local administrator selects to remove a collection
 1. System displays list of collections currently available to the system
 2. Local administrator selects one of the collections and tells the system to delete it
 3. System presents a message to local administrator asking him to confirm if he wants to remove

the selected collection

4. Local administrator confirms to the system that he wants to delete it
5. System deletes selected collection and display deleted message to the local administrator, then presents a new list of currently available collections to the local administrator

repeat 1-5 till done with removing the collection(s)

1b. Local administrator selects to update a collection

1. System presents list of currently available collections to the system for the local administrator
2. Local administrator selects a collection
3. System loads values for selected collection and displays them to the local administrator

Special Requirements:

- The ADL middleware for the configuration files already exists and the files generated must be generated so they are compatible with it.

Appendix B

B. 1 Suggestion for DTD for the collection_opml.xml file

```
<?xml version="1.0" ?>
<!DOCTYPE opml [

<!ELEMENT opml(head,body)>
<!ELEMENT head (title)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT body (outline)>
<!ELEMENT outline (outline*|EMPTY)>

<!ATTLIST outline text CDATA "">
<!ATTLIST outline type (group|collection) "">
<!ATTLIST outline coll CDATA "">
<!ATTLIST outline metahtml CDATA #IMPLIED]>
]>
<!-- Below is an example of the layout of the file (attributes have not been added) -->
<opml>
  <head>
    <title></title>
  </head>
  <body>
    <outline><!-- All collections wrapper -->
      <outline> <!-- Wrapper for group of collection -->
        <outline/> <!-- Wrapper for single collection -->
      </outline>

      <outline>
      </outline>
    </outline>
  </body>
</opml>
```


Appendix C

C. 1 Configuration files, suggestion for description format for ADL

Table C-1: Configuration file documentation

Metadata for a configuration file	
Purpose	Short description of what the file configures
Standards/Formats	List different standard and formats that are used in the configuration file
Required	Is file required or optional in the configuration of the system? (yes no)
Resources	- URI to online examples of the configuration file - URI to DTD for XML files
Dependencies	Is the values in this file used in other configuration files? (Exports:filename) Does the file import values from other configuration files? (Imports:filename)
Notes	Special notes for configuration file
File URI	where the file is located in the file hierarchy

Table 10-1 lists the format used to give a description of some of the configuration files in ADL. MW_BASEDIR is the directory where the middleware is installed.

Description of ADL configuration files

The following tables is just to give a brief view of how a standard format for description of configuration files can be done. Not all information about the configuration files are complete, some may be in error, but it is just an example of how it can be done.

Table C-2: Configuration file documentation

collection_opml.xml	
Purpose	Configures the drop-down menu for selecting collections in the webclient
Standards/Formats	XML
Required	yes
Resources	DTD: none (suggested DTD in appendix B)
Dependencies	None
Notes	-File can't contain standalone amperstands, use & -Duplicate collection names may cause errors
File URI	MW_BASEDIR/hierarchies/collection_opml.xml

Table C-3: Configuration file documentation

domain.js	
Purpose	To set Domain-name of the server (for example: fenris.idi.ntnu.no)
Standards/Formats	Javascript
Required	Yes
Resources	None
Dependencies	None
Notes	Only contain the line document.domain="Domain-NameOfServer";
File URI	MW_BASEDIR/javasript/domain.js

Table C-4: Configuration file documentation

webclient.conf	
Purpose	Used to set the WEBAPP_URL (for example: http://fenris.idi.ntnu.no:8080/adlmw/)
Standards/Formats	Java property file
Required	Yes
Resources	None
Dependencies	None
Notes	-WEBAPP_URL is what is most important to set in this file
File URI	MW_BASEDIR/WEB_INF/config/webclient.conf

Table C-5: Configuration file documentation

access-report-template.xml	
Purpose	Template used to make access reports, used to view results from queries
Standards/Formats	XML
Required	Yes
Resources	DTD: http://www.alexandria.ucsb.edu/middleware/dtds/ADL-access-report.dtd Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/access-report-template.html
Dependencies	None
Notes	-Use & for ampersands -Contains a DTD for an XML description of access report
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/access-report-template.xml

Table C-6: Configuration file documentation

browse-report-template.xml	
Purpose	Template used for browse reports
Standards/Formats	XML
Required	Yes
Resources	DTD: http://www.alexandria.ucsb.edu/middleware/dtds/ADL-browse-report.dtd Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/browse-report-template.html
Dependencies	None
Notes	None
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/ browse-report-template.xml

Table C-7: Configuration file documentation

bucket-report-template.xml	
Purpose	Template for bucket report
Standards/Formats	XML
Required	Yes
Resources	DTD: http://www.alexandria.ucsb.edu/middleware/dtds/ADL-bucket-report.dtd Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/bucket-report-template.html
Dependencies	None
Notes	None
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/ bucket-report-template.xml

Table C-8: Configuration file documentation

bucket99.conf	
Purpose	Configuration file for the bucket99 driver, database URL, queries, logging, collection driver, metadata driver, query translation location and vocabularies in this file
Standards/Formats	Java property file, SQL
Required	Yes
Resources	Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/bucket99.conf-annotated/
Dependencies	Import: database.properties Import: metadata.xml
Notes	WARNING: file contains password and username for database in plain text
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/bucket99.conf

Table C-9: Configuration file documentation

database.properties	
Purpose	Contains username and password for database
Standards/Formats	Java property file
Required	Optional
Resources	Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/username-password-database
Dependencies	Export: bucket99.conf
Notes	-WARNING: file contains password and username in plain text
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/database.properties

Table C-10: Configuration file documentation

drivers.conf	
Purpose	Where you set access control (gatekeepers) and drivers for the collection
Standards/Formats	Java property file
Required	Yes
Resources	Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/bucket99-x-drivers.conf-annotated/
Dependencies	None
Notes	Only contain default values.
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/drivers.conf

Table C-11: Configuration file documentation

metadata.xml	
Purpose	Contains the metadata used for describing the collection.
Standards/Formats	XML
Required	Yes
Resources	DTD: http://www.alexandria.ucsb.edu/middleware/dtds/ADL-collection-metadata.dtd
Dependencies	Export: bucket99.conf
Notes	None
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/metadata.xml

Table C-12: Configuration file documentation

query-translator.py	
Purpose	To define/structure the bucket definitions so it is possible to transform them to SQL queries, the buckets are implemented in this file.
Standards/Formats	Jython, SQL
Required	Yes
Resources	Online example: http://alexandria.ucsb.edu/middleware/sample-config-files/query-translator.py http://www.alexandria.ucsb.edu/research/docs/ADLDevGuide/Output/HTML-05-1.html#wp189905 http://piru.alexandria.ucsb.edu:8888/opadl/uploads/124/query-translator.py_new_style.txt
Dependencies	None
Notes	Can use 2 different coding styles
File URI	MW_BASEDIR/WEB-INF/config/collections/[collname]/query-translator.py

Table C-13: Configuration file documentation

web.xml	
Purpose	Used to set default start page for the webclient
Standards/Formats	xml, java servlets
Required	Yes
Resources	None
Dependencies	None
Notes	None
File URI	MW_BASEDIR/WEB-INF/web.xml

Table C-14: Configuration file documentation

frame_blank.html	
Purpose	Used to list new event on the default page of the web-client
Standards/Formats	html
Required	Yes
Resources	None
Dependencies	None
Notes	None
File URI	MW_BASEDIR/frame_blank.html

Appendix D

D. 1 Optimal Database layout for the Galleri Nor metadata

The optional layout of the database would be that the terms for title, subject related text and assigned terms would be combines in one table. According to David Valentine at UCSB it is best to create a single table with the terms combined. They used a similar technique in the original ADL bucket schema. The text table for Galleri Nor would be:

foto_text (fotoid, title, assigned-terms, subject-related-text)

With text indexes

```
CREATE FULLTEXT INDEX title_ix ON foto_text (title);
CREATE FULLTEXT INDEX subject_ix ON foto_text (subject-related-text);
CREATE FULLTEXT INDEX assigned_ix ON foto_text (assigned-terms);
```

- title: title for the object "PLACE. SUBJECTS. By Fotografer"
- assigned-terms: includes the motiv, and location (navn).
- subject-related-text: includes, title, assigned-terms, plus any other text you might want to include.

How the bucket would be implemented as dictionaries in jython in the query-translator.py configuration file:

```
# query-translator.py
bucket["adl:assigned-terms"]=
UT.Bucket(
    "textual",
    UT.standardTextualOperators,
    P.Textual_MySQLFulltext (
        "foto_text",
        "FOTOID",
        "assigned-terms",
        UT.Cardinality("1")
    )
)

bucket["adl:subject-related-text"]=
UT.Bucket(
    "textual",
    UT.standardTextualOperators,
    P.Textual_MySQLFulltext (
        "foto_text",
        "FOTOID",
```

```
        "subject-related-text",
        UT.Cardinality("1")
    )
)

bucket["adl:titles"]=
UT.Bucket(
    "textual",
    UT.standardTextualOperators,
    P.Textual_MySQLFulltext (
        "foto_text",
        "FOTOID",
        "title",
        UT.Cardinality("1")
    )
)
```

Appendix E

E. 1 Configuration files for the ADL and the Galleri Nor collections

User names and passwords have been replaced with `xxxxxx` where necessary.

E. 1. 1 collection_opml.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- USE & YOU CANNOT HAVE ANY STANDALONE AMPERSTANDS -->
<!-- having a coll= not found error? -->
<!-- you may have a duplicate collection name -->
<!-- you may be reading the wrong collection_opml.xml file -->
<!-- outline -->
<!-- attr: text - menu title -->
<!-- attr: type (group|collection) -->
<!-- attr: coll = for type=collection, middleware collection name.
If it is a group, it is the name that is passed to the query_page servlet -->
<!-- attr: metahtml = if a collection is not at the standard location (http://collections../[coll]
use this as areference to the html metadata url (to be deprecated, in the near future) -->
<!-- attr: querypage == future concept... querypage?parameters. ability to different querypages for different
collections -->
<!-- attr: basemwurl == future concept... ability to use different middleware servers -->

<opml>
  <head>
    <title>Operational ADL</title>
  </head>
  <body>
    <outline text="All Collections" type="group" coll="Catalog">
      <outline text="Digital Data Libraries" type="group" coll="dlib_all">
        <outline text="..Galleri NOR"
          type="collection"
          coll="galerinor"
          metahtml="http://www.alexandria.ucsb.edu/adl/docs_public/ADLMet
adata_ingest/ADL_CLM_HTMLGenerator.php?collection=sio"/>
        <outline text="..Sample"
          type="collection"
          coll="sample-collection"
          metahtml="http://www.alexandria.ucsb.edu/adl/docs_public/ADLMet
adata_ingest/ADL_CLM_HTMLGenerator.php?collection=sio"/>
      </outline>
    </outline>
  </body>
</opml>
```

E. 1. 2 domain.js

```
document.domain="fenris.idi.ntnu.no";
```

E. 1. 3 webclient.conf

```
# THIS FILE USES VARIABLE SUBSTITUTION AT BUILD TIME
# (at)name(at) are variable substitutions
# variables starting with wc are in default_webclient.properties
# Others:
#   mw.machine app.path build.running_collections
#   configured in build.properties
```

```
# Minimal Site configuration. (Now done at build time)
# WEBAPP_URL -- REQUIRED for functionality.
# DEFAULT_STYLE -- setup for testing
# query.running_collections -- set to adl_catalog
```

```
# WEBAPP_URL: http://localhost:8080/webclient/
# mw.machine&app.path configured in build.properties
WEBAPP_URL: http://fenris.idi.ntnu.no:8080//adlmw/
```

```
# set to false for map frames not supporting highlight in map
# highlight_in_map
# set whether highlight in map is show in ResultsServlet
highlight_in_map: true
# bp.wc.showResultsMap
# set whether a results map is shown at the bottom of the page
show_results_map: false
```

```
permQueryPageReference: index.jsp
```

```
DEFAULT_STYLE: query_noframe
DEFAULT_STYLE_FRAME: query_frame
DEFAULT_HIERARCHY_FRAME: DEFAULTFRAME
DEFAULT_COLLECTION: dlib_all
DEFAULT_OPML: /hierarchies/collection_opml.xml
# setup for default install
```

```
BASE_MW_URL: {WEBAPP_URL}
```

```
### base_middleware_url only used by CollectionMetadataServlet
#### avoid it's use
base_middleware_url: {BASE_MW_URL}
```

```
wimp_cancel_url: /wimp_cancel
refresh_rate: 2
results_page: 20
max_results: 100
```

```
wc.view.adl.bucket=/views/adl/ADL-bucket-report.xsl
```

```
wc.view.adl.browse=/views/adl/browse.xsl
wc.view.adl.access=/views/adl/ADL-access-report.xsl
wc.view.adl.resultList=/views/adl/resultList_download.xsl
```

```
query.default_collection: {DEFAULT_COLLECTION}
query.default_template: {DEFAULT_STYLE}
query.default_gazetteer: adl_gazetteer
```

```
# where queries get submitted to
query.mw_query_path: wimp_query
```

```
# query.page_base_url query.collection_base_url used by QueryPageServlet
query.page_base_path: query_page?&style={DEFAULT_STYLE}&coll=
query.collection_base_path: display_info?action=info&
```

```
# HTML formatted Collection Metadata Properties
# this is where the collection metadata files will be found by default
# for opADL this is http://collections.alexandria.ucsb.edu/[collection]/metadata.html
# for opADL this is collection_base_metadata_url[collection]collection_metadata_html_file
# this can be changed in the collection hierarchy
# edit the collection_opml.xml, and add an attribute metahtml
query.collection_base_metadata_url: CollectionHTML.jsp?collection=
query.collection_metadata_html_file:
```

RUNNING COLLECTIONS LIST

```
query.running_collections.class: edu.ucsb.adl.webclient.RunningCollectionsParameter
```

```
# this class is used to crete the menu.
```

```
OPML_CLASS: edu.ucsb.adl.webclient.CollectionHierarchyOPML
```

```
query.collection_hierarchies: DEFAULT,DEFAULTFRAME,GAZ
query.collection_hierarchies.default: DEFAULTFRAME
```

```
query.collection_hierarchies.display_xlst: adl_collection_short.xsl
```

```
DEFAULTFRAME.class: {OPML_CLASS}
DEFAULTFRAME.file: {DEFAULT_OPML}
DEFAULTFRAME.default: {DEFAULT_COLLECTION}
DEFAULTFRAME.page_base_path: query_page?style={DEFAULT_STYLE_FRAME}&hierar-
chy={DEFAULT_HIERARCHY_FRAME}&coll=
```

```
DEFAULT.class: {OPML_CLASS}
DEFAULT.file: {DEFAULT_OPML}
DEFAULT.default: {DEFAULT_COLLECTION}
DEFAULT.page_base_path: query_page?style={DEFAULT_STYLE}&coll=
```

```
GAZ.class: {OPML_CLASS}
GAZ.file: /hierarchies/gaz_opml.xml
```

GAZ.default: adl_gazetteer
GAZ.page_base_path: query_page?style={DEFAULT_STYLE}&hierarchy=GAZ&coll=

E. 1. 4 access-report-template.xml

```
<?xml version="1.0"?>
<ADL-access-report>

  <identifier>${collection}:${holding}</identifier>
  <!-- when web-interface is activated, will need to add an alternatives wrapper -->
  <alternatives>
    <web-interface>
      <title>Description (HTML)</title>
      <rights>Copyright 2004 National Library of Norway</rights>
      <!-- this needs to be a link to the object, it possible -->
      <url>http://www.nb.no/cgi-bin/galnor/
gn_sok.sh?id=$main.fotoid&skjema=2&fm=4</url>
    </web-interface>
    <download>
      <title>Image only</title>
      <rights>Copyright 2004 National Library of Norway</rights>
      <url>http://www.nb.no/cgi-bin/galnor/bilde.sh?id=$main.fotoid&size=1</url>
      <format>JPEG</format>
      <mime-type>image/jpeg</mime-type>
    </download>
  </alternatives>

</ADL-access-report>
```

E. 1. 5 browse-report-template.xml

```
<?xml version="1.0"?>
<ADL-browse-report>
  <identifier>${collection}:${holding}</identifier>
  <image>
    <url>http://www.nb.no/cgi-bin/galnor/bilde.sh?id=$main.fotoid&size=4</url>
    <format>JPEG</format>
    <width>512</width>
    <height>512</height>
  </image>
  <image>
    <url>http://www.nb.no/cgi-bin/galnor/bilde.sh?id=$main.fotoid&size=16</url>
    <format>JPEG</format>
    <width>120</width>
    <height>120</height>
  </image>
</ADL-browse-report>
```

E. 1. 6 bucket-report-template.xml

```
<?xml version="1.0"?>
<!DOCTYPE ADL-bucket-report SYSTEM
"http://www.alexandria.ucsb.edu/middleware/dtds/ADL-bucket-report.dtd">
```

```

<ADL-bucket-report>
  <identifier>${collection}:${holding}</identifier>

  ${cTitle$  <bucket name="adl:titles">
    <textual-value>
      <text>${cTitle.tittel}</text>
    </textual-value>
  </bucket>}$

  ${location$  <bucket name="adl:geographic-locations">
    ${location$<spatial-value>
      <point>
        <latitude>${location.latitude}</latitude>
        <longitude>${location.longitude}</longitude>
      </point>
    </spatial-value>}$
  </bucket>
  }$

  ${main.datering_fra$  <bucket name="adl:dates">
    <temporal-value>
      <range>
        <begin>${main.datering_fra}</begin>
        <end>${main.datering_til}</end>
      </range>
    </temporal-value>
  </bucket>
  }$

  <bucket name="adl:types">
    <hierarchical-value>
      <term vocabulary="ADL Object Type Thesaurus">photographs</term>
    </hierarchical-value>
  </bucket>

  <bucket name="adl:formats">
    <hierarchical-value>
      <term vocabulary="ADL Object Format Thesaurus">JPEG</term>
    </hierarchical-value>
  <!-- <hierarchical-value>
    <term vocabulary="ADL Object Format Thesaurus">HTML</term>
  </hierarchical-value>
  -->
  </bucket>

  <bucket name="adl:subject-related-text">
    <textual-value>
    <field uri="tag:MOTIV"
    name="[Galleri Nor] MOTIV"/>
    <text>${main.motiv}</text>

```

```

        </textual-value>
        ${preciselocation$$[preciselocation.navn$<textual-value>
<field uri="tag:PREISERING"
name="[Galleri Nor] PREISERING"/>
        <text>${preciselocation.navn$$[preciselocation.prisering$, $preciselocation.priser-
ing$$]$</text>
        </textual-value>}]$
        }$
    </bucket>

    ${origin$<bucket name="adl:originators">

        ${origin$<textual-value>${origin.jptype$ <field uri="tag:JPTYPE"
name="[Galleri Nor] $origin.jptype$"/>}$
            <text>${origin.navn$$[origin.occupation$, $origin.occupation$$]$ $[origin.opplysninger$,
$origin.opplysninger$$]$</text>
            </textual-value>
            }$
        </bucket>}]$

    <bucket name="adl:identifiers">
        <identification-value>
            <mapped-identifier
                namespace="GALNOR Photo ID">${holding$</mapped-identifier>
            </identification-value>
            <identification-value>
                <mapped-identifier
                    namespace="GALNOR Inventory Number">${main.invr$</mapped-identifier>
            </identification-value>

    </bucket>

</ADL-bucket-report>

```

E. 1.7 bucket99.conf

```

# +=====+
# |
# | Bucket99 configuration file for collection 'galleri nor'.
# |
# | Caution: this file contains passwords!
# |
# |
# +=====+

# +-----+
# | LOGGING |
# +-----+

collection.log_file: none
metadata.log_file: none
query.log_file: none

# -----

```



```

# VALIDATION
# -----

VALIDATION: off
metadata.validation: @VALIDATION
query.validation: @VALIDATION

# +-----+
# | DATABASE |
# +-----+

DATABASE-DRIVER-CLASS: org.gjt.mm.mysql.Driver
# The fully-qualified name of the database driver class.

DATABASE-URL:\
jdbc:mysql://piru.alexandria.ucsb.edu/galnor?user=xxxxxx&password=xxxxxx
# jdbc:mysql://uranus.idi.ntnu.no/galnor?user=xxxxxx&password=xxxxxx

# The database connection URL.

DATABASE-PROPERTIES: database.properties
# Either the string "none" or the filename of a Java
# properties file containing database connection properties
# (e.g., username and password). If the filename is not
# absolute, it is interpreted relative to the directory
# containing the Bucket99 configuration file.

DATABASE-NAME: GaleriNOR on piru MySQL
# A short, descriptive name for the database

metadata.database_driver_class: @DATABASE-DRIVER-CLASS
metadata.database_url: @DATABASE-URL
metadata.database_properties: @DATABASE-PROPERTIES

query.database_driver_class: @DATABASE-DRIVER-CLASS
query.database_url: @DATABASE-URL
query.database_properties: @DATABASE-PROPERTIES
query.database_name: @DATABASE-NAME

# +-----+
# | COLLECTION DRIVER |
# +-----+

# If using edu.ucsb.adl.bucket99.FileCollectionDriver:

collection.metadata_file: metadata.xml
# The filename of the file containing the collection metadata.
# If the filename is not absolute, it is interpreted relative
# to the directory containing the Bucket99 configuration file.

# +-----+
# | METADATA DRIVER |
# +-----+

```

```

# -----
# ITEM-LEVEL METADATA

metadata.identifier_datatype: integer

metadata.views: BUCKET,BROWSE,ACCESS

BUCKET.name: adl:bucket
BUCKET.class: edu.ucsb.adl.bucket99.GenericQueryMetadataSubdriver
BUCKET.template: bucket-report-template.xml
BUCKET.queries: MAIN-QUERY,BUCKET-STED-QUERY,BUCKET-AREA-QUERY,BUCKET-ORIGINA-
TORS-QUERY,OBJECT-TITLE-QUERY
MAIN-QUERY.prefix: main
MAIN-QUERY.query: select fotoid,invnr,repronr, \
CASE WHEN YEAR(datering_fra) < 1 THEN NULL \
ELSE DATE_FORMAT(datering_fra, '%Y-%m-%d') \
END \
,CASE WHEN datering_til < datering_fra THEN DATE_FORMAT(datering_fra, '%Y-%m-%d') \
ELSE DATE_FORMAT(datering_til, '%Y-%m-%d') END \
,datering_kommentar, \
CASE WHEN tittel = 'null' THEN NULL \
ELSE tittel \
END \
,motiv FROM v_foto where fotoid = ?
MAIN-QUERY.columns: fotoid,invnr,repronr,datering_fra,datering_til,datering_kommentar,tittel,motiv
MAIN-QUERY.num_rows_expected: 1
#
BUCKET-STED-QUERY.prefix: location
BUCKET-STED-QUERY.query: select DISTINCT A.latitude, A.longitude \
FROM omraade_extended_with_lat_long A, sted B, foto_sted C \
WHERE B.stedid = C.stedid AND B.omraade_kode = A.omraade_kode AND C.fotoid = ?
BUCKET-STED-QUERY.columns: latitude,longitude
BUCKET-STED-QUERY.num_rows_expected: 0+

BUCKET-AREA-QUERY.prefix: preciselocation
BUCKET-AREA-QUERY.query: select A.navn, \
CASE \
WHEN B.prisering = A.navn THEN NULL \
ELSE B.prisering END \
from omraade_extended_with_lat_long A, sted B, foto_sted C \
WHERE B.stedid = C.stedid AND B.omraade_kode = A.omraade_kode and C.fotoid = ?
BUCKET-AREA-QUERY.columns: navn,prisering
BUCKET-AREA-QUERY.num_rows_expected: 0+

BUCKET-ORIGINATORS-QUERY.prefix: origin
BUCKET-ORIGINATORS-QUERY.query: select A.fotoid, B.navn, \
CASE WHEN B.yrke = 'null' THEN NULL \
ELSE B.yrke \
END , \
CASE WHEN jptype IS NULL OR jptype = 'null' \
THEN NULL \
ELSE jptype \
END , \

```

```
CASE WHEN   opplysninger IS NULL OR opplysninger = 'null' \
  THEN NULL \
  ELSE opplysninger \
  END \
from foto_jp A, juridisk_person B \
WHERE B.JPNR = A.jpnr AND A.fotoid = ?
BUCKET-ORIGINATORS-QUERY.columns: fotoid,navn,occupation,jptype,opplysninger
BUCKET-ORIGINATORS-QUERY.num_rows_expected: 0+
```

```
OBJECT-TITLE-QUERY.prefix: cTitle
OBJECT-TITLE-QUERY.num_rows_expected: 1?
OBJECT-TITLE-QUERY.columns: tittel
OBJECT-TITLE-QUERY.query: Select DISTINCT \
  CONCAT(CASE WHEN F.navn IS NULL OR F.navn = 'null' \
    THEN '' \
    ELSE CONCAT(F.navn, ' ') \
  END ,\
  concat(CASE WHEN C.tittel = 'null' OR C.tittel IS NULL \
    THEN CONCAT('Motivs: ',C.motiv) \
    ELSE C.tittel \
  END ,\
  CASE WHEN A.navn IS NULL OR A.navn = 'null' \
    THEN '. ' \
    ELSE concat('. By fotografer: ',A.navn ) \
  END \
) \
)\
from juridisk_person A, foto_jp B, v_foto C, foto_sted D, sted E, \
omraade_extended_with_lat_long F \
  where B.fotoid = C.fotoid AND C.fotoid = D.fotoid \
  AND D.stedid = E.stedid AND E.omraade_kode = F.omraade_kode \
  AND A.jpnr = B.jpnr AND A.yrke = 'fotograf' \
  AND C.fotoid = ? LIMIT 1
```

```
BROWSE.name: adl:browse
BROWSE.class: edu.ucsb.adl.bucket99.GenericQueryMetadataSubdriver
BROWSE.template: browse-report-template.xml
BROWSE.queries: MAIN-QUERY
```

```
ACCESS.name: adl:access
ACCESS.class: edu.ucsb.adl.bucket99.GenericQueryMetadataSubdriver
ACCESS.template: access-report-template.xml
ACCESS.queries: MAIN-QUERY
```

```
# -----
# Collection Report Validation
validation.select_query: SELECT fotoid FROM v_foto
validation.count_query: SELECT COUNT(*) FROM v_foto
```

```
# -----
# QUERY PROCESSING
```

```
query.minimum_connections: 1
query.maximum_connections: 2
```

```
query.connection_lifetime: 3600000
query.connection_reaper_cycle_time: 300000
query.test_query: SELECT COUNT(*) FROM v_foto
```

```
# -----
# QUERY TRANSLATION
```

```
query.translator_class: edu.ucsb.adl.bucket99.PythonQueryTranslator
```

```
translator.script: query-translator.py
translator.python_module_path: ../../modules
translator.python_class_path: ../../classes
translator.python_jar_path: ../../lib
```

```
# -----
# VOCABULARIES
```

```
# This collection does not require or use
# the vocabulary interface, but we include
# it here just to show what is possible.
```

```
translator.vocabularies:
```

E. 1.8 database.properties

```
user: xxxxxx
password: xxxxxx
```

E. 1.9 drivers.conf

```
# +=====+
# |
# | ADL middleware configuration file for collection 'doqq'.
# |
# | Catherine Masi
# | masi@library.ucsb.edu
# |
# +=====+
```

```
# Drivers.
```

```
collection.driver.class: edu.ucsb.adl.bucket99.FileCollectionDriver
# [per collection] The fully-qualified name of the collection
# service driver class.
```

```
metadata.driver.class: edu.ucsb.adl.bucket99.MetadataDriver
# [per collection] The fully-qualified name of the metadata
# service driver class.
```

```
query.driver.class: edu.ucsb.adl.bucket99.QueryDriver
# [per collection] The fully-qualified name of the query
# service driver class.
```

ACCESS CONTROL

collection.driver.gatekeeper.class: @DRIVER-ACCESS-CLASS
collection.driver.gatekeeper.argument: @DRIVER-ACCESS-ARGUMENT

metadata.driver.gatekeeper.class: @DRIVER-ACCESS-CLASS
metadata.driver.gatekeeper.argument: @DRIVER-ACCESS-ARGUMENT

query.driver.gatekeeper.class: @DRIVER-ACCESS-CLASS
query.driver.gatekeeper.argument: @DRIVER-ACCESS-ARGUMENT

E. 1. 10 metadata.xml

```
<?xml version="1.0"?>
<ADL-collection-metadata>
  <internal-name>galerinor</internal-name>
  <lowercase-name>galleri nor</lowercase-name>
  <uppercase-name>GALERI NOR</uppercase-name>
  <full-title>Galeri NOR -- National Library of Norway</full-title>
  <short-title>Galeri NOR</short-title>
  <responsible-party>
    <link url="http://idi.ntnu.no/">FIX URL GaleriNOR -- National Library of Norway</link>
  </responsible-party>
  <scope-and-purpose>The purpose of this collection is ????
</scope-and-purpose>
  <content-type>Photographic images compiled by GaleriNOR -- National Library of Norway
</content-type>
  <creation-date>
    <year>2004</year>
  </creation-date>
  <metadata-schema>Custom Metadata Schema in database</metadata-schema>
  <sample-metadata-record>
  </sample-metadata-record>
  <terms-and-conditions>ADD terms-and-conditions</terms-and-conditions>
  <contact>
    <name>CONACT</name>
    <organization>
      <link url="http://www.example.com/">EXAMPLE</link>
    </organization>
    <larger-organization>
      <link url="http://www.example.com/">EXAMPLE</link>
    </larger-organization>
    <postal-address>
      <city>EXAMPLE</city>

    </postal-address>
    <e-mail-address>EXAMPLE</e-mail-address>
    <phone-number>EXAMPLE</phone-number>
  </contact>
  <!-- need some work -->
  <item-type-hierarchy id="item-types">
    <item-type id="Geospatial works">
      <internal-name>geospatial works</internal-name>
      <external-name>geospatial works</external-name>
    </item-type>
  </item-type-hierarchy>
</ADL-collection-metadata>
```

```

    <item-type id="photographs">
      <internal-name>photographs</internal-name>
      <external-name>photographs</external-name>
    </item-type>
  </item-type>
</item-type-hierarchy>
  <item-type-thesaurus>The <link url="http://www.sdc.ucsb.edu/~mary/objtype.tes/index.htm">ADL
Object Type Thesaurus.</link>
</item-type-thesaurus>
  <item-format-hierarchy id="item-formats">
    <item-format id="online">
      <internal-name>Online</internal-name>
      <external-name>Online</external-name>
    <item-format id="image">
      <internal-name>Image</internal-name>
      <external-name>Image</external-name>
    <item-format id="jpeg">
      <internal-name>JPEG</internal-name>
      <external-name>JPEG</external-name>
    </item-format>
  </item-format>
</item-format-hierarchy>
  <item-format-thesaurus>The item formats are loosely based on <link url="ftp://ftp.isi.edu/in-notes/
rfc2046.txt">RFC 2046, Multipurpose Internet Mail Extensions (MIME).</link>
</item-format-thesaurus>
  <!-- end need some work -->

<ADL-search-buckets>
  <bucket id="adl:geographic-locations" constraint-type="spatial">
    <name>Geographic locations</name>
    <description>Geographic locations and regions described by latitude/longitude coordi-
nates.</description>
    <metadata-mapping>
    </metadata-mapping>
    <notes>Queries against this bucket are evaluated using a <link url="http://mathworld.wol-
fram.com/EquiangularProjection.html">Plate Carre projection</link>, i.e., by treating latitude and longi-
tude as Cartesian coordinates. Thus, query regions may not include the North or South Poles.
Additionally, non-rectangular query regions may not intersect the +/-180 meridian.</notes>
    <operators>
      <operator>contains</operator>
      <operator>is-contained-in</operator>
      <operator>overlaps</operator>
    </operators>
  </bucket>
  <bucket id="adl:dates" constraint-type="temporal">
    <name>Dates</name>
    <description>Dates and date ranges.</description>
    <metadata-mapping></metadata-mapping>
    <operators>
      <operator>contains</operator>
      <operator>is-contained-in</operator>
      <operator>overlaps</operator>
    </operators>

```

```

</bucket>
<bucket id="adl:types" constraint-type="hierarchical">
  <name>Types</name>
  <description>Terms drawn from a collection-dependent controlled vocabulary identifying
meaning or content.</description>
  <metadata-mapping>
  </metadata-mapping>
  <domain type="hierarchical" hierarchy="item-types"/>
  <operators>
    <operator>is-a</operator>
  </operators>
</bucket>
<bucket id="adl:formats" constraint-type="hierarchical">
  <name>Formats</name>
  <description>Terms drawn from a collection-dependent controlled vocabulary identifying
form or representation.</description>
  <metadata-mapping></metadata-mapping>
  <domain type="hierarchical" hierarchy="item-formats"/>
  <operators>
    <operator>is-a</operator>
  </operators>
</bucket>
<bucket id="adl:assigned-terms" constraint-type="textual">
  <name>Assigned terms</name>
  <description>Subject-related terms from controlled vocabularies. This bucket is a subset of
the "Subject-related text" bucket.</description>
  <metadata-mapping></metadata-mapping>
  <notes>Non-alphanumeric characters are treated as word separators.</notes>
  <operators>
    <operator>contains-all-words</operator>
    <operator>contains-any-words</operator>
    <operator>contains-phrase</operator>
  </operators>
</bucket>
<bucket id="adl:subject-related-text" constraint-type="textual">
  <name>Subject-related text</name>
  <description>Subject-related text, not necessarily from controlled vocabularies. This bucket
is a superset of the "Assigned terms" bucket.</description>
  <metadata-mapping></metadata-mapping>
  <notes>Non-alphanumeric characters are treated as word separators.</notes>
  <operators>
    <operator>contains-all-words</operator>
    <operator>contains-any-words</operator>
    <operator>contains-phrase</operator>
  </operators>
</bucket>
</ADL-search-buckets>
<ADL-metadata-views>
  <view>adl:bucket</view>
  <view>adl:access</view>
  <view>adl:browse</view>
</ADL-metadata-views>
<item-counts>
  <computation-date>

```



```

# - added fMotiv to subject-related-text
# - originators changed Nat. to National
#   - lower-cased types: images
# - added index notes
#
# Indexes Suggested
# v_foto(fotoid)
# v_foto(MOTIV)
# v_foto(DATERING_FRA,DATERING_TIL)
#   foto_sted(fotoid)
#   foto_sted(stedid)
#   sted(stedid)
#   sted(omraade_kode)
#   omraade_extended_with_lat_long(omraade_kode)
#   omraade_extended_with_lat_long(navn)
#   omraade_extended_with_lat_long(latitude,longitude)
#   foto_jp(fotoid)
#   foto_jp(jpnr)
#   JURIDISK_PERSON(jpnr)
#   JURIDISK_PERSON(navn)
#
# do we want full text (and appropriate paradigms) on
# - v_foto(MOTIV)
# - JURIDISK_PERSON(navn)
#   - omraade_extended_with_lat_long(navn)
# depends on the length and content of the fields
#####

```

```

import UniversalTranslator
UT = UniversalTranslator

```

```

import paradigms
P = paradigms

```

```

import string

```

```

originators = ["National Library of Norway", "GaleriNOR"]
### Can you download a tif? if not, online image jpeg.
## add website later
formats=["Online", "Image", "JPEG"]
types=["images", "photographs",]

```

```

buckets = {}

```

```

buckets["adl:geographic-locations"]=UT.Bucket(
    "spatial",
    UT.standardSpatialOperators,
    P.Adaptor_Relationship(
        "foto_sted",
        "fotoid",
        "stedid",
        UT.Cardinality("0+"),
        P.Adaptor_Relationship(
            "sted",

```

```

    "stedid",
    "omraade_kode",
    UT.Cardinality("1+"),
    P.Spatial_BoxCoordinatesNoCrossing(
        "omraade_extended_with_lat_long",
        "omraade_kode",
        "latitude",
        "latitude",
        "longitude",
        "longitude",
        UT.Cardinality("1")
    )
)
)
)

buckets["adl:dates"]=UT.Bucket(
    "temporal",
    UT.standardTemporalOperators,
    P.Temporal_BeginEnd(
        "v_foto",
        "fotoid",
        "datering_fra ",
        "datering_til",
        UT.Cardinality("1")
    )
)

# adl:types uses a constant, types defined above
buckets["adl:types" ]= UT.Bucket(
    "hierarchical",
    UT.standardHierarchicalOperators,
    P.Hierarchical_Constant(
        "v_foto",
        "fotoid",
        "ADL Object Type Thesaurus",
        types,
        UT.Cardinality("1")
    )
)

# adl:formats uses a constant, formats defined above
buckets["adl:formats" ]=UT.Bucket(
    "hierarchical",
    UT.standardHierarchicalOperators,
    P.Hierarchical_Constant(
        "v_foto",
        "fotoid",
        "ADL Object Format Thesaurus",
        formats,
        UT.Cardinality("1")
    )
)
)

```

```

buckets["adl:originators"]=UT.Bucket(
    "textual",
    UT.standardTextualOperators,
    P.Adaptor_Relationship(
        "foto_jp",
        "fotoid",
        "jpnr",
        UT.Cardinality("0+"),
        P.Textual_LikeSubstring(
            "juridisk_person",
            "jpnr",
            "navn",
            UT.Cardinality("1"),
            P.TextUtils.mappings.uppercaseAlphanumericOthersToWhitespace,
            P.TextUtils.deleteLists.keepAll,
            "UPPER"
        )
    )
)

```

```

# this is a python function, that we will use
# more than once

```

```

fMotiv = P.Textual_LikeSubstring(
    "v_foto",
    "fotoid",
    "motiv",
    UT.Cardinality("1")
)

```

```

buckets["adl:titles"]= UT.Bucket(
    "textual",
    UT.standardTextualOperators,
    fMotiv
)

```

```

buckets["adl:assigned-terms" ]= UT.Bucket(
    "textual",
    UT.standardTextualOperators,
    fMotiv
)

```

```

# keeping this separate, just makes it easier to maintain
# the subject related test bucket

```

```

fNavn = P.Adaptor_Relationship(
    "foto_sted",
    "fotoid",
    "stedid",
    UT.Cardinality("0+"),
    P.Adaptor_Relationship(
        "sted",
        "stedid",

```

```

"omraade_kode",
UT.Cardinality("1+"),
  P.Textual_LikeSubstring(
    "omraade_extended_with_lat_long",
    "omraade_kode",
    "navn",
    UT.Cardinality("1"),
    P.TextUtils.mappings.uppercaseAlphanumericOthersToWhitespace,
    P.TextUtils.deleteLists.keepAll,
    "UPPER"
  )
)
)
)

```

```

buckets["adl:subject-related-text" ]= UT.Bucket(
  "textual",
  UT.standardTextualOperators,
  P.Adaptor_Union({
    "navn":fNavn, # the feild URI's "navn", "motiv" are here because function expects a dictionary
    "motiv":fMotiv
  })
)

```

```

#"adl:originators" : UT.Bucket(
#  "textual",
#  UT.standardTextualOperators,
#  P.Adaptor_Constant(
#    "tag:http://purl.org/dc/elements/1.1/creator",
#    P.Textual_LikeSubstring(
#      "GALNOR.V_FOTO",
#      "FOTOID",
#      "MOTIV",
#      UT.Cardinality("1"),
#      P.TextUtils.mappings.uppercaseAlphanumericOthersToWhitespace,
#      P.TextUtils.deleteLists.keepAll,
#      "UPPER"))),

```

```

buckets["adl:identifiers"]=UT.Bucket(
  "identification",
  UT.standardIdentificationOperators,
  P.Identification_String(
    "v_photo",
    "fotoid",
    "motiv",
    ["GALNOR"],
    UT.Cardinality("1")
  )
)
)

```

```
translator = UT.Translator(buckets,rewriteSetOpsAsSubqueries=1)
```

```
def translate ():  
    return translator.translate(constraint, vocabularies)
```

E. 1. 12 web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<!DOCTYPE web-app  
PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"  
"http://java.sun.com/dtd/web-app_2_3.dtd">  
<!-- created with the webxml ant task -->  
<web-app>  
  
    <context-param>  
        <param-name>middleware_configuration_file</param-name>  
        <param-value>WEB-INF/config/middleware.conf</param-value>  
    </context-param>  
  
    <!-- ADL Monitor servlets -->  
  
    <servlet>  
  
        <servlet-name>collectionMonitor</servlet-name>  
        <servlet-class>edu.ucsb.adl.monitors.CollectionMonitorServlet</servlet-class>  
        <init-param>  
            <param-name>middleware_configuration_file</param-name>  
            <param-value>WEB-INF/config/middleware.conf</param-value>  
        </init-param>  
        <!-- load after everything -->  
        <load-on-startup>998</load-on-startup>  
    </servlet>  
  
    <servlet>  
  
        <servlet-name>collectionStartup</servlet-name>  
        <servlet-class>edu.ucsb.adl.monitors.CollectionStartupServlet</servlet-class>  
        <init-param>  
            <param-name>middleware_configuration_file</param-name>  
            <param-value>WEB-INF/config/middleware.conf</param-value>  
        </init-param>  
        <!-- load after everything -->  
        <load-on-startup>999</load-on-startup>  
    </servlet>  
  
    <!-- END ADL Monitor servlets -->  
  
    <!-- Start ADL Library Middleware servlets -->  
    <servlet>  
        <servlet-name>cancel</servlet-name>  
        <servlet-class>edu.ucsb.adl.middleware.CancelServlet</servlet-class>
```

```

</servlet>

<servlet>
  <servlet-name>collection</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.CollectionServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>configuration</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.ConfigurationServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>metadata</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.MetadataServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>query</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.QueryServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>reference</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.ReferenceServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>results</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.ResultsServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>status</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.StatusServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>unload</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.UnloadServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>unreference</servlet-name>
  <servlet-class>edu.ucsb.adl.middleware.UnreferenceServlet</servlet-class>
</servlet>

<servlet>
  <servlet-name>bucket99-status</servlet-name>
  <servlet-class>edu.ucsb.adl.bucket99.StatusServlet</servlet-class>
  <init-param>
    <param-name>style_sheet_url</param-name>
    <param-value>bucket99-status.css</param-value>
  </init-param>

```

```

        <init-param>
            <param-name>map_server_url</param-name>
            <param-value>http://maps.alexandria.ucsb.edu/mapserver/foot-
print?width=300& height=150& north=$N& south=$S& east=$E& west=$W& for
mat=gif</param-value>
        </init-param>
    </servlet>

<servlet>
    <servlet-name>collection-availability</servlet-name>
    <servlet-class>edu.ucsb.adl.middleware.CollectionAvailabilityServlet</servlet-class>
</servlet>
<!-- END ADL Library Middleware servlets -->

<!-- ADL Webclient on Middleware servlets -->

    <servlet>
        <!-- class that displays the collection metadata summary -->
        <servlet-name>display_info</servlet-name>
        <servlet-class>edu.ucsb.adl.webclient.CollectionMetadataServlet</servlet-class>
        <init-param>
            <param-name>org.apache.velocity.properties</param-name>
            <param-value>/velocity.properties</param-value>
        </init-param>
        <init-param>
            <param-name>middleware_configuration_file</param-name>
            <param-value>WEB-INF/config/middleware.conf</param-value>
        </init-param>
        <init-param>
            <param-name>web_client_configuration_file</param-name>
            <param-value>WEB-INF/config/webclient.conf</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

<servlet>
    <!-- class that displays the query page -->
    <servlet-name>query_page</servlet-name>
    <servlet-class>edu.ucsb.adl.webclient.QueryPageServlet</servlet-class>

    <init-param>
        <param-name>org.apache.velocity.properties</param-name>
        <param-value>/velocity.properties</param-value>
    </init-param>
    <init-param>
        <param-name>web_client_configuration_file</param-name>
        <param-value>WEB-INF/config/webclient.conf</param-value>
    </init-param>
    <init-param>
        <param-name>middleware_configuration_file</param-name>
        <param-value>WEB-INF/config/middleware.conf</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>

```

```

<servlet>
  <servlet-name>wimp_metadata</servlet-name>
  <servlet-class>edu.ucsb.adl.webclient.MetadataXslServlet</servlet-class>
  <init-param>
    <param-name>middleware_configuration_file</param-name>
    <param-value>WEB-INF/config/middleware.conf</param-value>
  </init-param>
  <init-param>
    <param-name>web_client_configuration_file</param-name>
    <param-value>WEB-INF/config/webclient.conf</param-value>
  </init-param>
</servlet>
<servlet>
  <servlet-name>wimp_query</servlet-name>
  <servlet-class>edu.ucsb.adl.webclient.QueryServlet</servlet-class>
  <init-param>
    <param-name>middleware_configuration_file</param-name>
    <param-value>WEB-INF/config/middleware.conf</param-value>
  </init-param>
  <init-param>
    <param-name>web_client_configuration_file</param-name>
    <param-value>WEB-INF/config/webclient.conf</param-value>
  </init-param>
</servlet>
<servlet>
  <servlet-name>wimp_cancel</servlet-name>
  <servlet-class>edu.ucsb.adl.webclient.CancelServlet</servlet-class>
  <init-param>
    <param-name>middleware_configuration_file</param-name>
    <param-value>WEB-INF/config/middleware.conf</param-value>
  </init-param>
  <init-param>
    <param-name>web_client_configuration_file</param-name>
    <param-value>WEB-INF/config/webclient.conf</param-value>
  </init-param>
</servlet>
<servlet>
  <servlet-name>wimp_results</servlet-name>
  <servlet-class>edu.ucsb.adl.webclient.ResultsXslServlet</servlet-class>
  <init-param>
    <param-name>middleware_configuration_file</param-name>
    <param-value>WEB-INF/config/middleware.conf</param-value>
  </init-param>
  <init-param>
    <param-name>web_client_configuration_file</param-name>
    <param-value>WEB-INF/config/webclient.conf</param-value>
  </init-param>
</servlet>
<!-- END ADL Webclient on Middleware servlets -->

<!--Start ADL Monitor mappings -->

```



```
<servlet-mapping>
  <servlet-name>collectionMonitor</servlet-name>
  <url-pattern>/cmon</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>collectionStartup</servlet-name>
  <url-pattern>/startup</url-pattern>
</servlet-mapping>
```

```
<!-- END ADL Monitor mappings -->
```

```
    <!--Start ADL Library Middleware mappings -->
```

```
<servlet-mapping>
  <servlet-name>cancel</servlet-name>
  <url-pattern>/cancel</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>collection</servlet-name>
  <url-pattern>/collection</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>configuration</servlet-name>
  <url-pattern>/configuration</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>metadata</servlet-name>
  <url-pattern>/metadata</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>query</servlet-name>
  <url-pattern>/query</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>reference</servlet-name>
  <url-pattern>/reference</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>results</servlet-name>
  <url-pattern>/results</url-pattern>
</servlet-mapping>
```

```
<servlet-mapping>
  <servlet-name>status</servlet-name>
  <url-pattern>/status</url-pattern>
```

```

</servlet-mapping>

<servlet-mapping>
  <servlet-name>unload</servlet-name>
  <url-pattern>/unload</url-pattern>
</servlet-mapping>

<servlet-mapping>
  <servlet-name>unreference</servlet-name>
  <url-pattern>/unreference</url-pattern>
</servlet-mapping>

<servlet-mapping>
  <servlet-name>bucket99-status</servlet-name>
  <url-pattern>/bucket99_status</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>collection-availability</servlet-name>
  <url-pattern>/collection_availability</url-pattern>
</servlet-mapping>
<!-- END ADL Library Middleware mappings -->

<!-- Start ADL Webclient on Middleware mappings -->
<servlet-mapping>
  <servlet-name>wimp_metadata</servlet-name>
  <url-pattern>/wimp_metadata</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>wimp_query</servlet-name>
  <url-pattern>/wimp_query</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>wimp_cancel</servlet-name>
  <url-pattern>/wimp_cancel</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>wimp_results</servlet-name>
  <url-pattern>/wimp_results</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>display_info</servlet-name>
  <url-pattern>/display_info</url-pattern>
</servlet-mapping>
<servlet-mapping>
  <servlet-name>query_page</servlet-name>
  <url-pattern>/query_page</url-pattern>
</servlet-mapping>
<!-- END ADL Webclient on Middleware mappings -->

<!-- welcome file list -->
<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>

```

```
</welcome-file-list>
<!-- error page list here-->
```

```
<!-- taglibs -->
<taglib>
  <taglib-uri>http://jakarta.apache.org/taglibs/xtags-1.0</taglib-uri>
  <taglib-location>/WEB-INF/tld/taglibs-xtags.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>http://jakarta.apache.org/taglibs/response-1.0</taglib-uri>
  <taglib-location>/WEB-INF/tld/taglibs-response.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>http://jakarta.apache.org/taglibs/io-1.0</taglib-uri>
  <taglib-location>/WEB-INF/tld/taglibs-io.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>http://jakarta.apache.org/taglibs/input-1.0</taglib-uri>
  <taglib-location>/WEB-INF/tld/taglibs-input.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>http://java.sun.com/jstl/core</taglib-uri>
  <taglib-location>/WEB-INF/tld/c.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>http://java.sun.com/jstl/xml</taglib-uri>
  <taglib-location>/WEB-INF/tld/x.tld</taglib-location>
</taglib>
```

```
<!-- end taglibs -->
```

```
</web-app>
```

E. 1. 13 frame_blank.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head><title></title>
  <link href="default_css/webclient.css" rel="stylesheet" type="text/css">
</head>
<body>
  <div ID="banner"> The Alexandria Digital Library node at NTNU</div>
  <p>At present the Fenris server at NTNU (Norwegian University of
  Science and Technology) is running a ADL node that provides geospatial access
  to the Galleri Nor collection</p>

  <h2><b>How to search</b> the Galleri Nor collection:</h2>
  <ol>
    <li>Select <b>Galleri Nor</b> from <b>'Collection to search'</b></li>
    <li><b>Zoom in</b> on Norway in the interactive map</li>
    <li><b>Draw a box</b> around the region you want to find photographs from</li>
    <li><b>Scroll down</b> the search menu and click the <b>'Start search'</b> button</li>
```

```
<li>Wait for the results (can take some time)</li>
</ol>
<p>&nbsp;</p>
<p>&nbsp;</p>

<p>The above is a quick introduction to searching of the Galleri Nor collection, it is
possible to use other search parameters obviously</p>
<ul>
<li>Using placenames (&quot;Trondheim&quot;) as 'Words to Search for'
will find items with &quot;Trondheim&quot; in the title.</li>
<li>Focusing on a map region (region where Trondheim is located) and
using &quot;kirke&quot; as 'Words to Search for' will find
churches in Trondheim, this is the preferred method of searching.</li>
<li>I suggest a keyword ('Words to Search for') + defining a map region to search, or just a
keyword search. Avoid a text, geographic and type search. In the
cases where you get too many results, then limit the search further.</li>
</ul>
<p>The Quick Placename Search is not implemented at the moment.</p>
<p>Questions:<a href="mailto:kaitorge@idi.ntnu.no"> contact me</a></p><p>&nbsp;</p>
</body>
</html>
```

E. 1. 14 CollTest.properties

```
# a file to store a list of testing properties
#   for offline testing
collection=galerinor
# just the holding id (last part of the id)
holding_id=115740
```