ECP 2007 EDU 417008

ASPECT

ASPECT Approach to Federated Search and Harvesting of Learning Object Repositories

Deliverable number	*D2.1*
Dissemination level	*Public*
Delivery date	*28 February 2009*
Status	*Final*
Author(s)	*KUL,* *EUN,* *VMG,* *RWCS,* *KOB*

e*Content*plus

This project is funded under the *e*Content*plus* programme[1],
a multiannual Community programme to make digital content in Europe more accessible, usable and exploitable.

[1] OJ L 79, 24.3.2005, p. 1.

1 Introduction

The objective of our work is to foster adoption of standards and specifications necessary to support educational content discovery scenarios (discovery & evaluation, obtain). This deliverable focuses on best practices for connecting various learning object repositories to the ASPECT Service Centre (ASC) through federated search and harvesting. The ASC will provide a set of support services that will facilitate the interoperability of learning content.

This deliverable will start with an overview of existing standards and specifications for these solutions in sections 2 and 3.

Section 4 presents two usage scenarios for connecting content providers to the ASPECT infrastructure. Content providers usually use a custom format for describing their metadata. Section 4.1 explains the need for mapping this format to an agreed metadata application profile for ASPECT. Section 4.2 presents a scenario for harvesting the metadata while section 4.3 presents the scenario for enabling federated search. Section 4.4 describes the metadata validation service that will be used to validate the metadata against the chosen application profile.

To facilitate interoperability between repositories that choose one or more of these scenarios, we need a registry where this information can be stored. This registry is discussed in section 5. We conclude this deliverable in section 6 with an overview of the next steps towards the ASPECT infrastructure.

2 Metadata Standards & Specifications

A limited set of existing standards specifications are described below that are frequently used in educational settings:

- The IEEE Learning Object Metadata (LOM, 2002) is a hierarchical metadata standard usually encoded in XML, published by the IEEE in 2002. Its purpose is to enable the description of learning objects through attributes that include the type of object, author, owner, terms of distribution, and format, as well as pedagogical attributes, such as typical learning time or interaction style. LOM is based on early work in ARIADNE (Duval, et al., 2001) and IMS.

- Dublin Core (DC) (DCMI, 2003) is a standard for generic resource descriptions. The simple DC metadata element set consists of 15 elements, including title, creator, subject, description, publisher, contributor, date, type, format, identifier, source, language, relation, coverage, and rights.
  Currently, the DC-Education Community's Application Profile Task Group (DC-Ed, 2004) is working on the DC-Education Application Profile. This will be modular in nature, only defining properties or elements of relevance to educational use of resources. It will plug into other application profiles. It will use some existing Dublin Core elements, and may propose new ones and/or reuse elements from other metadata standards such as the IEEE LTSC LOM.

- MPEG-7 (ISO/IEC, 2004) is an ISO/IEC standard for describing multimedia content. MPEG-7 Multimedia Description Schemes (DSs) are metadata structures in XML that facilitate searching, indexing, filtering, and access.

Application Profiles

The goal of standardization is to produce a broadly acceptable specification, which does not impose unnecessary restrictions that may mitigate against its wider uptake and use. The normal way of addressing the need for interoperability is to define a profile of a standard. An application profile (Duval, Smith, & Van Coillie, Application profiles for learning., 2006) seeks to address the interoperability requirements between systems by:

- Retaining conformance with a base standard or specification; and
- Defining any new requirements in an open manner

Communities or organizations can adopt metadata standards in various ways. One can impose restrictions on existing metadata standards and for instance constrain the value space on some elements. The purpose of an application profile is to adapt or combine existing schemas into a package that is tailored to the functional requirements of a particular application, while retaining interoperability with the original base.

For instance, in (MELT, 2006), the consortium partners agreed on a MELT LRE application profile of LOM to describe the resources they would offer within the project, to enable educational content discovery within the project. MELT is an eContent*plus* project that has been designed to provide users of learning content in schools with access to more useful types of metadata that will allow them to find resources that fit their needs, language, cultures and preferred ways of teaching and learning.

Multilingual Vocabularies

Much of semantic interoperability of metadata builds on shared multilingual vocabularies. Important specifications related to multilingual vocabularies are CEN/ISSS WSLTs XVD, IMS (VDEX, 2004), (ZTHES, 2006) and (SKOS, 2006).

3 Specifications for Educational Content Discovery

A number of existing specifications are introduced in this section that can be used for educational content discovery. A combination of these will be discussed in section 4 as best practices for enabling effective educational content discovery. The specifications are divided in search services, harvesting and publishing services.

Furthermore, search services typically use one or more query languages. Specifications for those are added in Section 3.2. Search specifications usually return their results in one or metadata standards or specifications from Section 2.

Publishing specifications sometime use a standardised syndication format. Those are listed in section 3.5.

3.1 Search Specifications

Annex 1 presents detailed information on a number of search specifications. This annex has been taken from a Becta Report (Collett, et al., 2007). Only a subset of the services in this annex has been added below.

3.1.1 SQI

The Simple Query Interface (SQI) (Assche, et al., 2006) provides interoperability between search applications and learning object repositories and is designed to support many types of search technologies. The final SQI specification has been published as CEN ISSS Workshop Agreement (CWA) 15454:2005 (Simon, Massart, Assche, Ternier, & Duval, 2005). SQI is currently used in ARIADNE, the European e-content*plus* projects MACE (MACE, 2006) and MELT (MELT, 2006), the GLOBE consortium (GLOBE, 2004), etc.
Main characteristics of SQI are:

- Simplicity and ease of implementation,
- Neutrality in terms of query languages and result formats, and
- Support for both a synchronous and an asynchronous query mode.

3.1.2 SRU/SRW

With respect to searching the Internet, the Library of Congress maintains two search protocols(McCallum, 2006):

- Search/Retrieve via URL (SRU) is a REST [Fielding, 2000] style protocol that encodes the search method and parameters as a URI and returns an XML instance.
- Search/Retrieve Web Service (SRW) binds the same protocol to a SOAP implementation.

These search protocols were meant to replace the older ANSI/NISO Z39.50 (Z39.50, 2002), a protocol for searching libraries that was also maintained by the Library of Congress. HTTP is introduced as a new communication protocol.

3.1.3 Z39.50

Z39.50 (Z39.50, 2002) is a binary encoded protocol, which uses RPN (RPN, 1992) to represent its query structure. The queries are encoded and transmitted via TCP/IP to the Z server. As with SRW/SRU the Z39.50 protocol is synchronous and is tightly bound to a query format but only loosely coupled to result set formats meaning that a single instance can support many result set formats.

3.1.4 OASIS Search Web Service

The purpose of OASIS Search Web Services (OASIS, 2008) has been to define Search and Retrieval Web Services, combining various current and ongoing web service activities like Z39.50, SQI, SRU, OSIDs etc. The development of the web service interface specification includes:

- Search/Retrieve

- Query
- Sorting
- Record Retrieval
- Index Browsing

One of the advantages of this work is that they decouple query languages (e.g., CQL) and messaging protocols (e.g. SQI).

3.1.5 OpenSearch

OpenSearch (openSearch, 2009) is a collection of simple formats for the sharing of search results. The focus is on using existing specifications as a way to "publish" search results in order to facilitate further syndication and access by commonly available tools. OpenSearch uses its own simple query format transferred via HTTP.

3.1.6 O.K.I. OSIDs (Open Service Interface Definition)

The Open Knowledge Initiative is an MIT-lead, community effort to improve interoperability among applications and enterprise systems. OSIDs are contracts between service consumers and providers. Currently, there are OSIDs such as authentication, authorization, repository scheduling, workflow, and eLearning services (O.K.I, 2008).
The Repository OSID describes generic methods for searching, accessing, and updating content, including discovery of the metadata structures.

3.2 Query Language Specifications

Numerous query languages have been designed and are used in different contexts. A number of them are listed below:

3.2.1 VSQL

VSQL (VSQL, 2006) stands for “Very Simple Query Language”. This is a very lightweight query language that is supported within ARIADNE, MELT, Prolearn, GLOBE, etc as a query language for the SQI standard. It allows the user to issue a query with a number of search terms or keywords. Therefore, this only allows basic search within the networks.

3.2.2 PLQL

The "ProLearn Query Language" (Ternier S. , Massart, Campi, Guinea, Ceri, & Duval, 2008), a query language has been developed for repositories of learning objects. PLQL is primarily a query interchange format, used by source applications (or PLQL clients) for querying repositories (or PLQL servers). PLQL consists of a number of layers where each layer adds functionality. PLQL Layer zero e.g. offers the same functionality as VSQL. PLQL has been developed in a way that it can deal with hierarchical metadata schemas.

3.2.3 CQL

The “Contextual Query Language” (CQL, 2007) is a well-established abstract query language used for library search. SRU/W has the restriction that only CQL is supported as a query language.

3.2.4 QEL

The “Query Exchange Language” (Qu & Nejdl, 2004) is an RDF query language that can be expressed using the Prolog syntax, making it syntactically a subset of Prolog.

3.2.5 XQuery

A number of metadata standards and specifications have a XML-binding. XQuery (W3C, 2007) is a query language that allows for querying collections of XML data.

3.3 Harvesting Specifications

The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH, 2002) is a protocol for metadata harvesting (i.e., selecting metadata records from repositories based upon their identity, the date of their last modification, and their membership in predefined sets). OAI has its roots in the open access and institutional repository movements. Continued support of this work remains a cornerstone of the Open Archives program. Over time, however, the work of OAI has expanded to promote broad access to digital resources for eScholarship, eLearning, and eScience.

OAI-PMH is agnostic about what kind of metadata can be harvested, but conforming implementations must support the harvesting of Dublin Core metadata. Other projects have demonstrated how to harvest other metadata formats, e.g., LOM.

3.4 Publishing Specifications

The list in this section presents three specifications for publishing material into a repository.

3.4.1 OAI-ORE

The Open Archives Initiative Object Reuse and Exchange (OAI-ORE, 2008) defines standards for the description and exchange of aggregations of Web resources. These aggregations, sometimes called compound digital objects, may combine distributed resources with multiple media types including text, images, data, and video. The goal of these standards is to expose the rich content in these aggregations to applications that support authoring, deposit, exchange, visualization, reuse, and preservation.

3.4.2 SWORD

SWORD (SWORD, 2008) is a lightweight protocol for deposit. SWORD is a profile of the Atom Publishing Protocol. SWORD is a JISC-funded project 2007-2008. SWORD stands for Simple Web-service Offering Repository Deposit. The motivator for SWORD is 'lowering the barriers to deposit', principally deposit into repositories, but potentially deposit into any system, which wants to receive content from remote sources.

3.4.3 SPI

The Simple Publishing Interface (Ternier & Massart, 2008) provides a simple lightweight protocol for publishing data and metadata to a repository. It is easy to implement and integrate in existing applications. Some characteristics are that SPI

- Is neutral in terms of metadata standard
- Is an abstract interface

3.4.4 AtomPub

The Atom Publishing Protocol (AtomPub, 2007) is an application-level protocol for publishing and editing Web resources. The protocol is based on HTTP transfer of Atom-formatted representations. The Atom format is documented in the Atom Syndication Format, which is described in the next section.

3.5 Publishing Syndication Formats

3.5.1 RSS

RSS (RSS, 2007) is a web feed syndication format that is used to publish frequently updated content like blog entries, news entries, audio, video, etc. in a standardized XML format.
An RSS document includes full or summarized text, plus metadata such as publishing dates and authorship. Teachers to publish learning material for their students could for instance use this.

3.5.2 ATOM

ATOM (ATOM, 2005) is an XML-based document format that describes lists of related information known as "feeds". Feeds are composed of a number of items, known as "entries", each with an extensible set of attached metadata. For example, each entry has a title.

4 Content Discovery Scenarios

Imagine content providers that want to offer access to their materials. Content providers can maintain either a “repository” or a “referatory”. A “repository” contains objects whereas a “referatory” provides links to objects. However, in the remainder of this deliverable, we use the term “repository” to mean for both “repository” and “referatory” because our scenarios for content discovery are based on the metadata that describe the content.

In this section, we present two content discovery scenarios how they can establish this:

- **Harvesting**, where content providers enable harvesters of third-party entities to copy metadata from them and save a copy of this locally (section 4.1).
- **Federated search**, where content providers create a binding of an interoperable search service and therefore allow third-party entities to issue queries to be able to find and possibly use their content (section 4.3).

Both K.U.Leuven/ARIADNE and EUN have experience with this scenario in numerous projects (MELT & MACE), networks (GLOBE & PROLEARN) and specification and

standardization bodies (CEN/ISSS WS/LT & IMS). RWCS has similar experience in ADL-R, CORDRA and FRED.

4.1 Mapping Metadata

Repository owners typically use an internal metadata format for describing their content. This internal format has to be mapped to the metadata standard, which was agreed on using. This mapping phase is necessary in both the presented content discovery scenarios.

The first ASPECT content audit survey presented among other things, a number of questions about the characteristics of the content and the metadata that describes the content. One of the results has been that a number of the consortium partners already expose their metadata with the LOM LRE application profile. Based on common experiences of all partners, this application profile will be the starting point within ASPECT. It will be investigated where adaptations are needed in the continuation of the project.

Our approach to keeping multilingual vocabularies will be the vocabulary bank (VBE). This will be available for the different consortium partners. For instance, the vocabularies that are used in the above mentioned LRE application profile will be made available through the VBE. For a complete overview on the ASPECT approach on multilingual vocabularies, we refer to deliverable D2.3.

4.2 Harvesting

Frameworks for metadata harvesting (like OAI-PMH) enable harvesters to copy metadata from a repository and save a copy of the metadata locally. On top of this local copy, search services can be added to enable search in the metadata of the contents of the content providers. Much of the existing ARIADNE, EUN, MELT and GLOBE federated search infrastructures are based on the use of OAI-PMH. On top of that, a number of content providers in ASPECT already support OAI-PMH and have good experiences with this scenario in previous projects. Therefore, we have chosen the OAI-PMH protocol for our harvesting scenario.

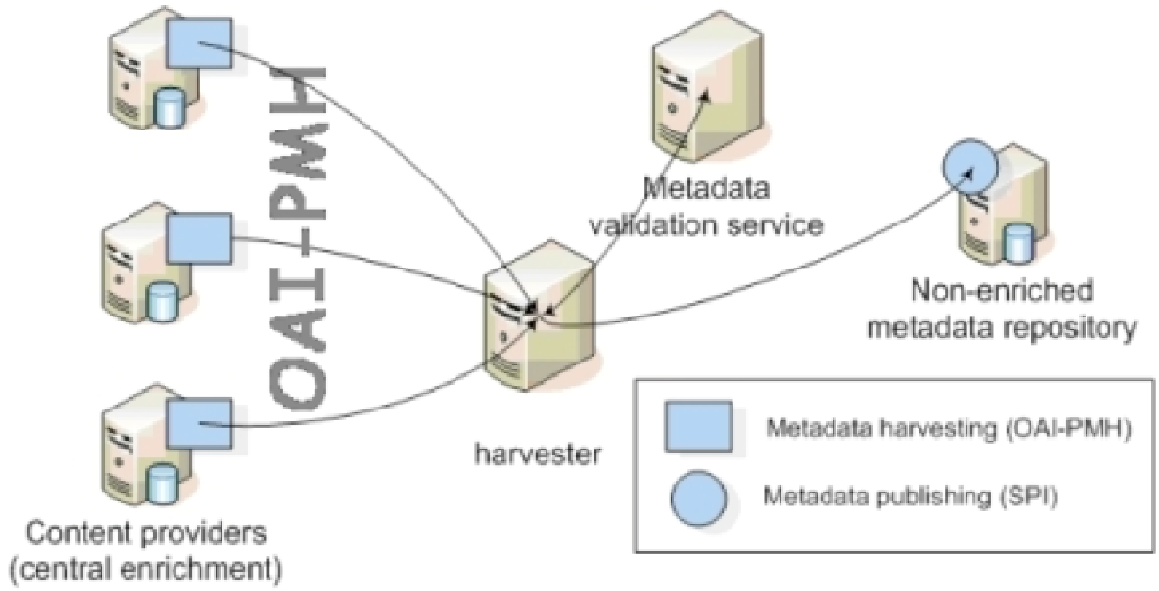

Figure 1 Basic Harvesting Infrastructure

Figure 1 shows an overview of a basic harvesting infrastructure. On the left hand side, one can see the content provider repositories that all have an OAI-PMH target on top of their repository (see 4.2.1). The harvester, shown in the middle, contacts the different OAI-PMH targets to harvest all, or part of their metadata. It uses the metadata validation service to validate the metadata against a validation scheme (see 4.4), and then stores the harvested metadata in a local metadata repository. For storing this metadata into a repository, publishing specifications can be used (section 3.4).

4.2.1 Setting up an OAI-PMH target

To expose metadata through OAI-PMH, a content provider needs to bind an OAI-PMH "target" to its repository. Figure 2 shows the four basic steps needed to set up this up.

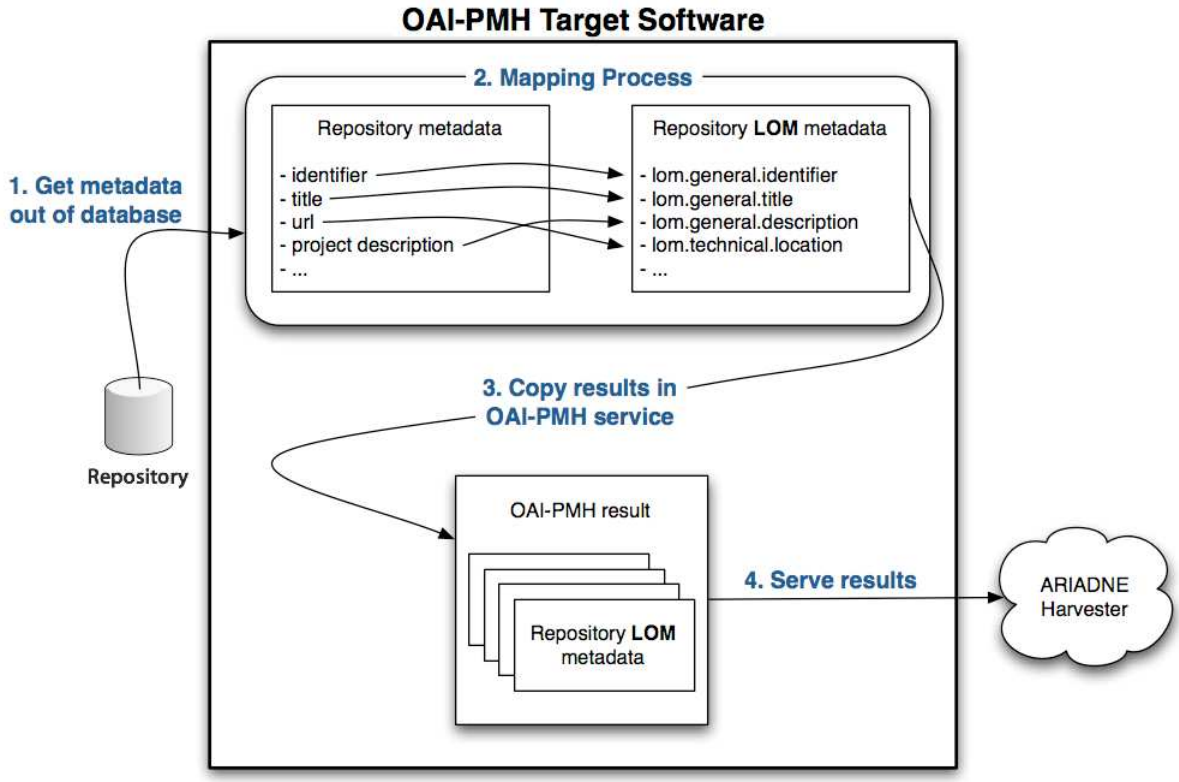

Figure 2: OAI-PMH Target Software

1. **Get metadata from database**: a connection has to be made to the content providers' database to get their metadata. This connection can be directly like e.g. performing SQL-queries on top of a relational database, but also indirectly like making use of one of the standards, specifications or other protocols that we have described in section 3.1.

2. **Mapping metadata:** The mapping process is best done on two different levels: conceptually (section 4.1) and technically. Different people can do the mapping on these two levels in parallel. A technical person can already start with the implementation of the mapping of some very basic fields (such as the title, the description, etc.), while another person does the mapping on the conceptual level.

3. **OAI-PMH results**: a record in the OAI-PMH format has to be expressed in a specific single XML format. All the metadata records in the have to be wrapped in the OAI-PMH format, so the result is conforming the OAI-PMH specification.

4. **Serving the results**: all results have to be exposed to the harvester through a REST web service. More information about this, can be found here in

In order to easy the process of connecting to the ASPECT, software libraries have been implemented for the third and forth step. In this way, the content provider only has to take care of the first two steps.

4.2.2 Lessons Learned

Because we have supported numerous repository owners in setting up harvesting targets, we noticed some reoccurring issues:

- *Connecting to the repository database*:
    - Sometimes the repository owners have to deal with an old database infrastructure, as they were not the ones that created them. This makes it sometimes technically difficult to retrieve the right information while implementing the conceptual mapping between the internal metadata format and the agreed metadata format.
    - Similarly, for some reason some repositories have no notion implemented of a last modified datestamp, which is needed in the OAI-PMH protocol to allow incremental harvesting. This allows for harvesting only those records after a specified datastamp.

- *Mapping on the conceptual level*:
    - It may happen that the repository uses different vocabularies than the ones used in the LRE Application Profile and that it is difficult and even not possible to map them completely. Therefore, in ASPECT, we will use the ASPECT Vocabulary Bank. For a complete overview of this concept, we refer to deliverable D2.3.

- *Mapping on technical level:*
    - Once the conceptual mapping has been done correctly, the technical mapping is typically easy as long as the technical persons possess the knowledge on XML and the standards in use. If not, a small learning curve is typically needed.
    - We have experienced this to be an error prone process. For instance, metadata instances that do not have a title but need one in the agreed metadata application profile. Therefore, we have introduced a validation service, which we will describe in section 4.4

- *OAI-PMH implementation:*
    - The OAI-PMH protocol leaves a number of decisions to the developer. For example, he can decide the size of the result set that is returned to the harvester. For our harvester, this is not a problem as long as the ‘resumption

token' is implemented. This token tells the harvester from which instance it needs to resume the harvesting process. These kinds of technical details are explained in detail in deliverable D2.3 that consists of material to support training and dissemination of the ASPECT approach.

Besides those issues, this approach has some major advantages:

- Once the conceptual and technical mapping has been done and approved, newly created and updated metadata can automatically be harvested too and therefore exposed through the ASPECT Service Centre (ASC).

- This setup allows for the use of an automatic metadata validation service (see section 4.4) while harvesting.

- Once the metadata has been harvested, all standards and specifications of section 3.1 can be implemented on top of the harvested metadata store. All results of the different content providers are cached within this store. Therefore, even if some providers are temporarily unavailable, complete results of these providers are still returned to the client.

- To implement an OAI-PMH target, there are several existing open source software libraries in different programming languages. Those can be freely (re-)used.

4.3 Federated Search

Searching beyond the borders of a local repository of a content provider is of great value to the end users as it enables searching in a vast amount of learning objects. Opposed to the harvesting scenario, federated search enables *real time* searching of repositories. This scenario is decentralised; it allows content providers to manage their collections autonomously.

Figure 3 shows this scenario where a client issues a query to the federated search engine. This engine is then responsible for

- issuing the query to all repositories in the network, and

- returning all the results to the client.

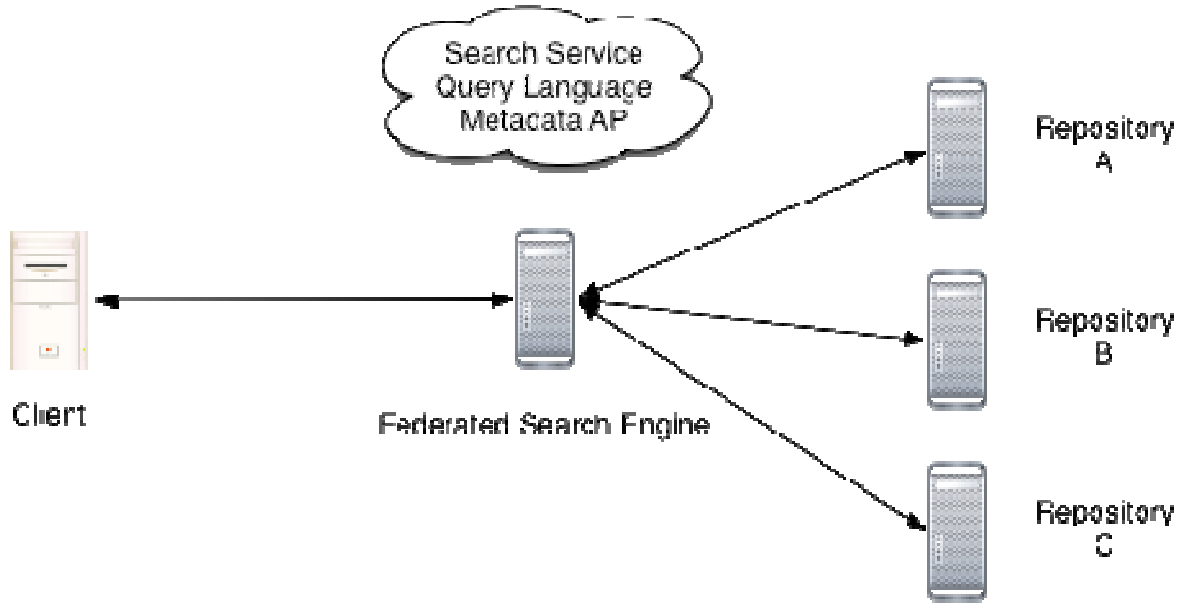

Figure 3: Federated Search

To achieve federated search (Ternier S. , 2008), you need a

- a search service and a binding (sections 3.1.1, 3.1.2),
- a metadata application profile and a binding (section 2), and
- a format for interchanging queries (section 3.2).

These requirements are presented in the next sections.

4.3.1 Search Service

Much of the existing ARIADNE, EUN, MELT and GLOBE federated search infrastructures are based on the use of SQI. In the ASPECT consortium, a number of partners already support SQI. Therefore, we will support SQI in the first version ASPECT infrastructure. SQI can serve as a gateway to other existing search protocols. Code has already been developed that maps SQI to SRU/W, ECL and O.K.I. These mappings could be integrated in the following versions of the ASPECT infrastructure.

4.3.2 Query Language

Experimentation efforts around the Prolearn Query Language (PLQL) have been conducted in a number of organisations and projects like

- ARIADNE
- the EUN Learning Resource Exchange initiative (MELT, CALIBRATE) (LRE-QL)
- Prolearn network-of-excellence
- The e-contentplus project MACE
- GLOBE
- Etc.

This query language is supported by SQI and will be used in the first version of the ASPECT infrastructure. The support of other query languages besides PLQL will be considered while designing the following generation of the ASPECT infrastructure.

4.3.3 Setting up an SQI Target

To expose metadata through SQI, content providers should realise a binding of the abstract SQI specification. As an example, we explain how to create a web service binding of SQI with the SOAP protocol. However, note that bindings of SQI can be made in other technologies as well.

Two steps are involved when setting up an SQI Target.:

1. Create an SQI Target: A WSDL-file (Web Services Description Language) has been created for SQI that implements the basic profile proposed by the web services interoperability (WS-I) organisation. The SQI WSDL binding can be used to generate stubs and skeletons for different environments like PHP, JAVA, .NET, etc. This is shown in Figure 4. Once the skeleton has been automatically created, a developer only needs to bind the generated skeleton code to his local environment for returning results.

2. Serve the metadata when a query is issued to the SQI Target and return the metadata in the correct format. Therefore, the metadata has to be mapped from the local metadata format to the one that has been agreed to use within the network. This mapping phase is therefore the same as the step in the harvester scenario. Note that the search services of the local environment can be used to match the issued query with the query results.

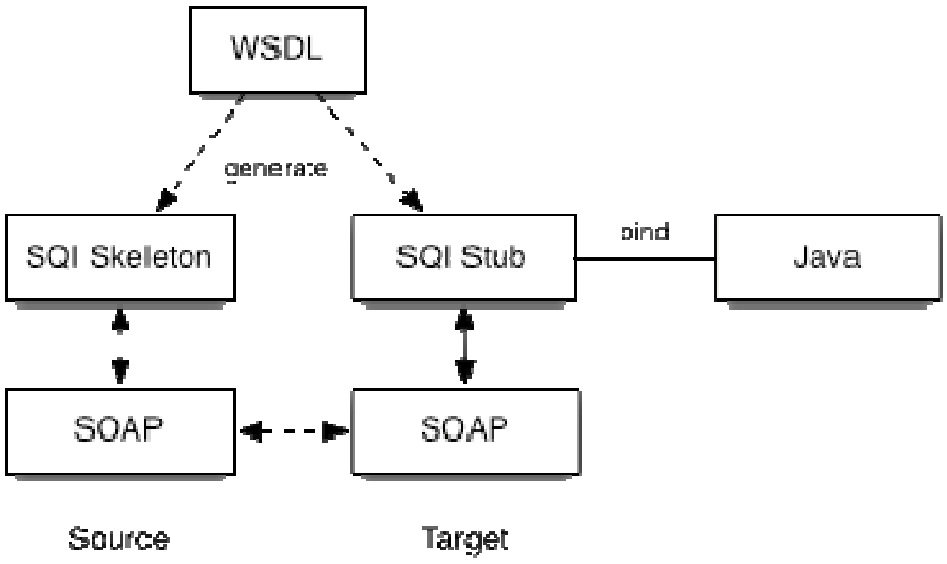

Figure 4: Creating a WSDL for SQI

4.3.4 Lessons Learned

Because we have been maintaining a federated search network for a number of years, we noticed some reoccurring issues

- *Connecting to the repository database:*

- Not all repositories are willing or able to implement a standardised search specification on top of their local repository. Although this puts little burden on the connecting repository, it requires adding code to the federated search engine each time such organisation wants to join. This is not a scalable solution.

- *Metadata Mapping on the conceptual and technical level*:

  - The same issues as in the harvesting scenario surface when mapping metadata from the internal format to the external metadata application profile. Therefore, we refer to section 4.2.1

- *Availability:*

  - Federated Search enables real time searching of repositories. The disadvantage of real time searching is that repositories in the network can be temporarily unavailable. Therefore, end users might be confronted with different query results when for instance issuing the same query twice.

Besides those issues, this approach has some major advantages:

- Federated search enables users to search in numerous repositories at the same time.

- It provides up-to-date results. This is an advantage when collections are volatile with frequent updates. Searching a cached metadata store can result in outdated results. However, by frequently re-harvesting of a provider, this disadvantage would be minimized.

4.4 Metadata Validation Service

Harvesting from or enabling federated search within numerous learning object repositories has revealed an important issue. As we mentioned before, mapping from the internal to the agreed metadata application profile in the network can be an error prone process. Manually checking every mapped instance does not scale. Therefore, we need a service for automatically validating instances. This service is described in the following sections.

4.4.1 Validation Components

The ARIADNE metadata validation service is a framework that can be extended with various state of the art metadata validation components:

- XML Schema (XSD) validation enables reusing various XSDs that check the structure of the XML instances.

- Schematron rules complement XSD in many ways. For instance, some conditional constraints cannot be expressed with XSD and can be easily encoded in schematron.

- The framework has been supplemented with other third party applications, such as VCARD validators, specific vocabulary validators of the ASPECT vocabulary bank (see Deliverable D2.3), etc.

The framework allows for combining these components to support validation against multiple application profiles of LOM that exist in various networks. For every application profile that is supported, the framework maintains a validation scheme URI that identifies a specific configuration of the validation components.

Figure 5 illustrates this by showing the LOM loose and ASPECT validation schemes. As one can see, the LOM loose validation scheme uses XSD schema, a custom vcard validator and an empty fields checker. The ASPECT validation scheme inherets these validation components and adds some specific ones like its own XSD schema, the vocabulary bank component and extra schematron rules.

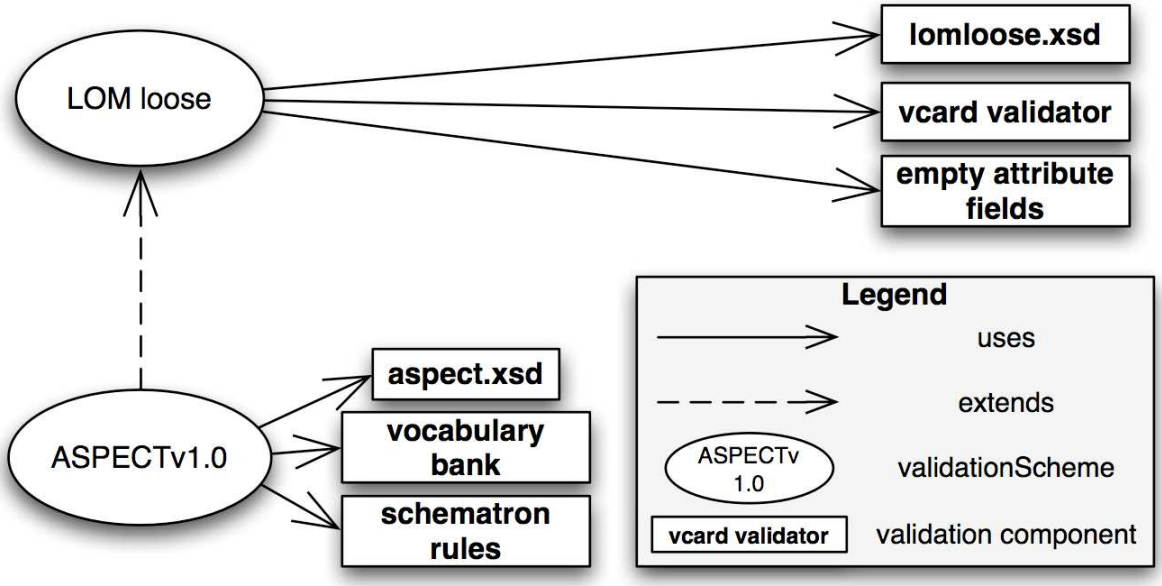

Figure 5 Aspect Validation Scheme

4.4.2 Automated Workflow

The validation framework supports a partial harvest. It tries to validate every metadata instance in the harvested set of the content provider. All validated metadata instances from the set are harvested. Non-validated instances are rejected. An error log is presented to the content provider, which can be used to resolve them.

The validation framework produces specific errors per instance. As these errors are very fine grained, it can be difficult to get a grasp on them, the size of the harvested metadata collection. For this reason,

- a complete overview is presented of all errors present in the harvested set. Different ways of grouping errors are available. For instance, every error can be summed up, along with the identifier of every instance that contains this particular error.
  This overview is available in a custom format.

- The content provider is given the opportunity to verify a single instance against the application profile himself through an online validation web service, which will be available in the Aspect Service Centre.

This error report is sent to the content provider, who has the responsibility of correcting the errors and thus, making sure that his content is available through the ASPECT Service Centre. The harvester will try to validate every non-validated metadata instance the next time the content provider is harvested. Typically, this happens on a weekly base.

4.4.3 Lessons Learned

The validation service gives a very detailed overview of the validation errors present in a metadata collection of a repository. For instance, a number of common validation errors are listed below:

- Empty fields or attributes in the metadata instances.
- VCARD errors:
  - VERSION is not 3.0 or higher
  - Mandatory VCARD elements N (“Name”) or FN (“Full Name”) are missing.
- No general identifier present
- No license information present
- No mapping from content providers’ vocabulary to the one that is used in the agreed metadata application Profile

Based on the automatic metadata validation, we can provide support to the content providers. However, we can only see the result of the mapping and therefore, we don’t know **how** the repository owner achieves the mapping. We can only guess whether the issue arises at the mapping level or the original metadata level.

A substantial number of content providers do not update the last modification date after resolving validation errors. This is likely due to the fact that it concerns an update of their mapping and thus involves all of their instances. In order to get the updated metadata, incremental harvesting cannot be used in this case. This leads to disabling incremental harvesting and thus, can result in more overhead, i.e. reharvesting all instances every time.

5 ASPECT Registry of Learning Object Repositories

In the previous section, we have proposed two scenarios for connecting content providers to the ASPECT infrastructure. However, to facilitate interoperability between repositories in the ASPECT infrastructure, it is necessary to develop one or more LOR registries where we can describe which scenarios content providers follow. This registry will therefore need to hold parameters that are for instance needed for SQI or OAI-PMH.

For a complete description on the data model of such registries, we refer to the ASPECT deliverable D2.2 on the design of a data model and architecture for a registry of learning object repositories and application profiles.

6 Conclusion and Outlook

The content discovery scenarios above are not exclusive and that whenever required, a hybrid approach can be considered for the inclusion of other different data sources. The presented content discovery scenarios will be disseminated to the partners in WP5 in a technical workshop that will be organized march 10-11 2009 in Leuven. We will support them in every way for implementing a connection to the ASPECT.

Our experience from prior work is that a number of metadata mapping issues (Section 2) could be resolved at a fast pace in a face-to-face contact discussing the issues amongst the technical people of the content provider. For this, we will evaluate if it's needed to organise another workshop with the WP5 partners that is specifically targeted on validating the metadata.

The three main components of the ASPECT infrastructure for content discovery are the

- Federated search and harvesting infrastructure (D2.1)
- Registry of learning object repositories and application profiles (D2.2)
- Vocabulary bank (D2.3)

All the main parts of the ASPECT infrastructure will be implemented in a first version in M9 of the project. We will use the consortium meeting in Vigo march 4-6 2009 to bring all the different components together for a complete design of the ASPECT architecture. All details of this architecture will be described on the wiki that has been created for D2.4 to support training and dissemination during the project.

7 References

Assche, F. V., Duval, E., Massart, D., Olmedilla, D., Simon, B., Sobernig, S., et al. (2006). Spinning interoperable applications for teaching & learning. *Journal of Educational Technology & Society , 9* (2), 51-67.
ATOM. (2005). *Atom Syndication Format*. From http://www.atompub.org/2005/07/11/draft-ietf-atompub-format-10.html
AtomPub. (2007). *RFC5023: The Atom Publishing Protocol*. From http://bitworking.org/projects/atom/rfc5023.html
CQL. (2007). *The Common Query Language*. From http://www.loc.gov/standards/sru/specs/cql.html
DC-Ed. (2004). *DC-Education Application Profile*. Retrieved 2009 from http://dublincore.org/educationwiki/DC_2dEducation_20Application_20Profile
DCMI. (2003, February). ISO 15836-2003 Dublin Core Metadata Element Set. From Dublin Core Metadata Initiative: http://dublincore.org/
Duval, E., Forte, E., Cardinaels, K., Verhoeven, B., Durm, R. V., Hendrikx, K., et al. (2001). The ARIADNE Knowledge Pool System: a Distributed Digital Library for Education. *Communications of the ACM , 44* (5), 73-78.
Duval, E., Smith, N., & Van Coillie, M. (2006). Application profiles for learning. *Sixth International Conference on Advanced Learning Technologies* (pp. 242-246). Kerkrade: IEEE.
GLOBE. (2004). *The Global Learning Objects Brokered Exchange (GLOBE) alliance*. From http://globe-info.org/
ISO/IEC. (2004, October). JTC1/SC29/WG11 MPEG-7.
LOM. (2002). 1484.12.1-2002 Standard for Learning Object Metadata (LOM). *IEEE LTSC LOM* .
MACE. (2006). *Metadata for Architectural Contents*. From www.mace-project.eu/
McCallum, S. (2006). A Look at New Information Retrieval Protocols: SRU, OpenSearch/A9, CQL, and XQuery. *IFLA.* Seoul.
MELT. (2006). *Learning Resources for Schools*. Retrieved 2009 from http://info.melt-project.eu
O.K.I. (2008). *OSID v3*. From http://www.okiproject.org/view/html/site/oki/node/2289
OAI-ORE. (2008). *Open Archives Initiative Object Reuse and Exchange* . From http://www.openarchives.org/ore/
OAI-PMH. (2002). *OAI-PMH*. From http://www.openarchives.org/
OASIS. (2008, July). *OASIS Search Web Services*. From http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=search-ws
openSearch. (2009). *OpenSearch Specification 1.1* . From http://www.opensearch.org/Home
Qu, C., & Nejdl, W. (2004). Interacting the Edutella/JXTA peer-to-peer network with Web services. *Applications and the Internet, 2004.*, (pp. 67-73).
RPN. (1992). *Reverse Polish Notation (RPN)*. From http://www.cni.org/pub/NISO/docs/Z39.50-1992/Z39.50.part3.html
RSS. (2007). *RSS 2.0 Specification*. From http://www.rssboard.org/rss-specification
Simon, B., Massart, D., Assche, F. V., Ternier, S., & Duval, E. (2005). A simple query interface specification for learning repositories. CEN Workshop Agreement (CWA 15454).
SKOS. (2006). *Simple Knowledge Organization System*. From http://www.w3.org/2004/02/skos/

SWORD. (2008). *SWORD: Simple Web-service Offering Repository Deposit*. From http://www.swordapp.org/
Ternier, S. (2008). *Standards Based Interoperability for Searching In and Publishing To Learning Object Repositories.* Leuven: Katholieke Universiteit Leuven.
Ternier, S., & Massart, D. V. (2008). A Simple Publishing Interface for Learning Object Repositories. *World Conference on Education Multimedia, Hypermedia and Telecommunications* (pp. 1840-1845). AACE.
Ternier, S., Massart, D., Campi, A., Guinea, S., Ceri, S., & Duval, E. (2008). Interoperability for Searching Learning Object Repositories, The ProLearn Query Language . *D-Lib Magazine , 14* (1/2).
VDEX, I. (2004). *IMS Vocabulary Definition Exchange*. From http://www.imsglobal.org/vdex/
VSQL. (2006). *Very Simple Query Language (VSQL)*. From http://ariadne.cs.kuleuven.be/lomi/index.php/QueryLanguages_stable
W3C, X. (2007). *XQuery 1.0: An XML Query Language*. From http://www.w3.org/TR/xquery/
Z39.50. (2002). *ANSI/NISO Z39.50*. From http://www.loc.gov/z3950/agency/
ZTHES. (2006). *The Zthes specifications for thesaurus representation, access and navigation*. From http://zthes.z3950.org/

8 Annex 1 – Search Service Specifications

This annex is taken as such from a Bectra report (Collett, et al., 2007). Search Services under the remit of this report can be broadly categorised as follows:

- open specifications that are designed to be repository agnostic and thus provide access to a wide range of repository data
- open specifications that provide proprietary access to a single repository of data which is useful due to the breadth of the content of such a repository
- open specifications that provide proprietary access to families of repositories which are useful mainly due to the number of repositories implementing the specification.

Search services can be either synchronous or asynchronous in operation (indeed some offer both options). Synchronous services provide response messages directly to query requests (one-to-one). Asynchronous queries let the data provider return multiple asynchronous responses that are merged by the requesting client.

8.1 SRW (Search/Retrieve Web Service)

SRW is an XML-based protocol for information retrieval. Its development was motivated, in part, to provide a web-oriented protocol similar to Z39.50. It is designed to be used with a specific query language (CQL see 9.5.1.1) and therefore richness of query functionality is inherent in its specification. It is however not tightly bound to a particular result set format and indeed does not specify the format of the result records within the standard. This leads to the possibility that an SRW service can support multiple formats which can be designed to be tailored to particular domains. The response itself can contain results or be a pointer to a named ‘result set’.
It defines three operations in its Service Definition:

- The searchRetrieve Operation: the basic operation by which queries and retrieval requests, and their responses, are passed between client and server.
- The scan Operation: enables the client to browse terms from indexes defined for at the server.
- The explain Operation: retrieves a document describing the capabilities of the server.

Both the structure contained in the request and the structure of the response are defined using a WSDL definition specific to SRW. SRW is a synchronous protocol and authentication is not defined in the specification but can be combined with a separate authentication model.

8.2 SRU (Search/Retrieve via URL)

SRU is the same in operation to SRW except that the query is encoded within a URL as opposed to within a SOAP request body.

8.3 Z39.50

Z39.50 is a binary encoded protocol which uses RPN to represent its query structure. The queries are encoded and transmitted via TCP/IP to the Z server. As with SRW/SRU the Z39.50 protocol is synchronous and is tightly bound to a query format but only loosely coupled to result set formats meaning that a single instance can support many result set formats. In addition, a server can support multiple databases (equating to multiple collections of records which can be queried as if they are different targets). A Z39.50 client maintains an 'association' with the Z39.50 server and operations are bound to that association. The server can choose to support multiple functions but there are a core set of functions which must be supported. These are:

- The Init Operation: The session negotiation phase which contains (optionally) authentication
- The Search/Response Operation: The creation of the result set and returning of information about the result of the search, including (optionally) the first 'n' results.
- The Present Operation: The retrieval of additional results. If the target (optionally) supports named result sets, then multiple searches can be conducted on one session concurrently. Otherwise there can only ever be 1 set of results live in a session at any one time.

Z39.50 also supports (optionally) browsing (scan), sorting and multiple extended services (including record update). Listing and describing all these features is considered to be beyond the scope of this report.

8.4 SQI (Simple Query Interface)

SQI is an abstract model for query and response messages. It is a session-based protocol and is designed to be independent of query language, messaging protocol (e.g. SOAP, RPC, RMI) and results format and can support both synchronous and asynchronous return of results. It includes an optional simple authentication specification and separates messages for commands from the messages for queries. An 'application profile' of SQI with associations for data representations, query language and messaging is required to implement an SQI interface between a client and data provider. SQI provides a method to set the "format" of the query result, however the specification of how all of the individual results are combined into the entire results set and the format of the entire results is left to the application profile.
The SQI API describes:

- A Service Interface, consisting of synchronous and asynchronous Query methods, and result set iterators.
- A set of Service Configurations, including modifiers on the Query including maximum results to return, start and end of result set and so on.

Several Service Bindings are available including Java APIs and Web Services WSDL interfaces.

8.5 OpenSearch

OpenSearch is a collection of simple formats for the sharing of search results. The focus is on using existing specifications as a way to "publish" search results in order to facilitate further syndication and access by commonly available tools. OpenSearch uses its own simple query format transferred via HTTP. Simple HTTP get requests are used for query the query. OpenSearch defines a synchronous only request-response model with no authentication in

specification but can be implemented as an extension. OpenSearch is widely supported and is integrated with Internet Explorer 7 and Firefox 2.0. OpenSearch consists of:

- An xml description document which is machine readable and describes to open search enabled clients how they should use your search engine, the type of content it searches, owner information etc.
- Optional extensions to support relevance, referrer (to allow a search engine to identify where the results came from), query extensions and suggestions for complete search terms.
- An extended RSS or Atom format for results to enable further syndication.

It is possible to use the description element without the syndication result format if you only want a search engine to be able to properly search your site, however if you wish to augment the result set data with specific targeted metadata the response elements are also required.

8.6 NISO Metasearch Specifications

These specifications consist of 4 distinct standards (currently in draft form). Two of these relate specifically to result and interchange formats and are covered in section 10, the other two relate to search services:

NISO Z39.92-200x, Information Retrieval Service Description Specification: defines a method of describing Information Retrieval oriented electronic services, including but not limited to those services made available via the Z39.50, SRU/SRW, and OAI protocols. The ZeeRex standard addresses the need for machine readable descriptions of services in order to enable automatic discovery of and interaction with previously unknown systems. It specifies an abstract model for service description and a binding to XML for interchange.

NISO RP-2006-02, NISO Metasearch XML Gateway Implementers Guide: describes a gateway which is based on the NISO-registered SRU protocol. This gateway provides a mechanism for information service providers to expose their content and services to a Metasearch engine. While the task group recognized that the longer term goal is some type of standardized query protocol based on SRU/SRW, an XML gateway provides an immediate, low entry barrier method for content providers to interact with metasearch services.

8.7 Google (Ajax) and Google Base

Google is made up of many services and discussion of them all is beyond the scope of this document. However there are 2 main services of relevance to resource discovery.

Google AJAX Search API

This is a JavaScript library that allows developers to embed Google Search in web pages and other web applications. The API provides simple web objects that perform inline searches (Web Search, Local Search, Video Search, Blog Search, News Search, and Book Search).

Google Base

This is a service that allows content providers to submit online and offline digital content to make it searchable on Froogle, Google Maps or the main Google web search (when submitted, offline content is put online). Content can be submitted using a web form, a bulk upload option (e.g. submitting an excel sheet containing multiple descriptions of content, or by developing an ad hoc application that uses the Base Application Programming Interface (API). The latter supports services for searching for data items using both the structured and unstructured languages, discovering metadata, and inserting, updating, and deleting data items.

8.8 Google Scholar

Google Scholar (GS) is a web search engine that indexes the full-text of scholarly literature across an array of publishing formats and disciplines. GS index includes most peer-reviewed online journals, except for those published by Elsevier, the world's largest scientific publisher. It is similar in function to the freely available Scirus from Elsevier, CiteSeer, and getCITED. GS allows users to search for digital or physical copies of articles, whether they be online or in libraries. The service can be called from the GS website (http://scholar.google.com/) or by including the appropriate html form on a remote web page.

8.9 Yahoo!

See Section 9.5.1.5 of original report for information about the Yahoo! Web services.

8.10 Amazon

See Section 9.5.1.6 of original report for information about the Amazon Web services.

8.11 Vivisimo

Vivisimo is a private company that develops technology to improve search on the web and in enterprises. Vivisimo's solutions are based on the concept of clustering search results around topics; for example, dividing the results of a search for "cell" into groups like "biology," "battery," and "prison.", which, they claim, allows users to intuitively narrow their search results to a particular category or browse through related fields of information, and seeks to avoid the "overload" problem of sorting through too many results.
Vivisimo technology is available to enterprise in the form of a cohesive search suite, Vivísimo Velocity, which includes the Velocity Search Engine, Velocity Clustering Engine and Velocity Content Integrator. The technology is also freely available to the public in the form of Clusty: a free, clustering search engine at http://clusty.com .

8.12 Scholar SFX

ScholarSFX is a service provided freely by Ex Libris. It enables libraries to create customized links based on their institution's electronic journal holdings and to display these links in Google Scholar search results. The library users are then able to link from the Google Scholar results to articles that are available through local institutional subscriptions or for free on the Web.

8.13 WebFeat

WebFeat is a commercial federated search engine for libraries developed by WebFeat. It allows library users to search any or all of a library's databases simultaneously with a single interface. WebFeat can search any database, including licensed databases, free databases, catalogues, Z39.50, Telnet, and proprietary databases.

8.14 LIMBS

LIMBS is an open source brokerage system that relies on open standards and open contents to promote exchanges of learning resources within a federation of e-learning systems. Contrary to the CELEBRATE brokerage system, from which it derives, LIMBS' role is limited to

carrying and routing messages exchanged by the federation members rather than to enforcing semantic interoperability. With LIMBS, semantic interoperability becomes the responsibility of the federation members that rely on “clients” to communicate with the brokerage system and to support the negotiation of common query languages and metadata formats. LIMBS itself adopts a service-oriented architecture so that each service (e.g. resource discovery, digital rights management) can be used separately and combined with any (group) of the others. The discovery service of LIMBS offers a mixed solution that combines harvesting (based on OAI-PMH) and federated searching (based on a Java Message Service (JMS) implementation of SQI).

8.15 IMS DRI (ECL implementation)

The IMS Digital Repositories Interoperability specification defines a reference model for pairs of services exposed by repositories including Search and Retrieve. A number of projects have implemented query services based on the DRI reference model, however, these projects have needed to define many implementation details which were left undefined in the DRI specification as the specification itself doesn’t ensure interoperability. This section discusses the ECL (eduSource Communication Layer) implementation.

The eduSource Communication Layer is an interoperability platform for connecting learning services repositories into the eduSource network. These repositories already offer their services through existing protocols. The ECL protocol enables these repositories to communicate with each other and enables other repositories and services to become a part of the eduSource network. The protocol is independent of any existing protocols and enables developers to build universal tools and services that will enable their users to connect and use services provided by any repository connected to the eduSource network. It can operate both synchronously and asynchronously.

8.16 ebXML

The OASIS ebXML Registry specifications were developed to achieve interoperable registries and repositories, with an interface that enables submission, query and retrieval on the contents of the registry and repository. It is a synchronous service.
There is support for different protocol bindings including SOAP and REST and the standard itself includes scope for federating queries to groups or repositories.
The ebXML search service mandates the support of a search service comprising of 2 distinct sections exposed via the query manager

The Browse and Drill Down operation: This service provides access to items which have been classified against an internal classification schema and as such is not a free text query using Boolean logic. It is accessed via specific web service call as opposed to utilising a query language and only supports the wildcard operator for ‘like’
The Filter Query operation: This type of submission provides the capability to execute rich queries to the query manager. The filter query supports an ebXML specific query structure which is tied to the ebRIM (registry information model).

ebXML is a complex and heavyweight technology and, as such, a this discussion of its search service implementation is only included for completeness. A more detailed discussion was considered beyond the scope of this report.