Masters Proposal

Meta-standardisation of Interoperability Protocols

Name: Jorgina Kaumbe do Rosario Paihama
jpaihama@cs.uct.ac.za

Supervised by: Dr Hussein Suleman
hussein@cs.uct.ac.za

Department of Computer Science
University of Cape Town
May 2009
1. Introduction
The amount of data available on the Internet increases on daily basis. One of the main objectives of the Digital Libraries community is to preserve that data and allow users to access it as easily as possible. Access to online information can be enhanced by allowing heterogeneous data providers to interoperate (communicate and exchange data amongst themselves). Currently there are many communication protocols that facilitate interoperability between systems. Some are simplistic and can be easily implemented but may lack some efficiency while others are very efficient and just as complex. Nevertheless they have been able to provide interoperability for Web resources. This study is an attempt to design a set of specifications that combines simplicity with efficiency to improve on the degree of interoperability.

1.1 What is interoperability?
Interoperability has many definitions across a wide spectrum of research areas. In computer science these definitions can be said to be all different because each of them is adapted in accordance to the context within which it is defined, but also similar because they all portray the same message of communication between heterogeneous systems.

In the ISO/IEC 2382-01 information technology vocabulary, interoperability is defined as “the capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units” [7].

Interoperability also is defined as “the capability of different programs to exchange data via a common set of exchange formats, to read and write the same file formats, and to use the same protocols” [25].

For this study, interoperability is defined as: The capability of different systems to communicate and exchange data with each other, using a set of predefined formats and protocols that will allow the systems to use each other’s services successfully. For this definition just like in the two definitions above there is the common objective of establishing communication between different programs, and while the user can have little or no knowledge of the characteristics of the different systems (like in definition 1), in most cases it is necessary to have a common set of exchange formats and use the same protocols (like in definition 2).

Interoperability can be achieved at different levels of abstraction. **Syntactic interoperability** occurs when two or more systems are capable of exchanging data using specified data formats and common communication protocols, in other words using the same syntax [25]. Syntactic interoperability is a requirement for any other level of interoperability to be achieved. XML and SQL are popular standards used to achieve syntactic interoperability.

**Semantic interoperability** is the ability to automatically and accurately interpret the information exchanged and produce useful results as expected by the end users of both systems involved [25]. This level of interoperability can only be achieved after two systems are syntactically interoperable.

**Pragmatic/functional interoperability** relies on a common set of functional primitives or on a common set of service definitions [4].
Technical/basic interoperability is based on common tools, interfaces and infrastructure providing the user with a universal API [4].

Legal interoperability deals with the legal implications of allowing free access to digital information and intellectual propriety rights and ownership issues [12].

1.2 The importance of interoperability
A lot of work goes into creating and preserving digital information systems. We may benefit even more if all those different systems could work together in the preservation of information.
This level of interoperability in digital library (DL) systems would [10] increase information accessibility, promote open access, allow easier creation of federated metadata archives, improve efficiency and reduce costs. Financial costs would be reduced by saving on development costs as well as storage costs as fewer repeated records may be necessary (because if one system has a record that a user needs, the user can get that by accessing that system via any another system).

The remainder of this proposal is structured as follows: section 2 presents a discussion on related work, past and state-of-the-art communication protocols; section 3 presents the research question, what problems are going to be tackled and the hypothesis; section 4 describes how the problem presented above will be solved; section 5 states what results are expected to be generated by this study; and section 6 describes the work detail.

2. Background Information/ Related Work
There are many communication protocols that offer various levels of interoperability. This section presents a discussion of some of the older and state-of-the-art interoperability protocols.

Z39.50
The Z39.50 protocol is a NISO application layer protocol that supports distributed search and retrieval between structured network services [8]. This protocol stipulates data structures and interchange rules that allow a client machine to search and retrieve records from databases on a server machine, across different platforms [11]. The protocol works in the following way: to initiate a session a connection is established via messages exchanged between the client and server; they then negotiate expectations and limitations on the activities that will occur (things like version used and maximum record size). After these agreements are negotiated, the client may submit a query. The server executes the search against a database(s), and a result set is created. Afterwards the client asks for records from the result set or requests that the server performs some additional processing of the result set before sending it to the client. After receiving the response set the client machine may do some further processing (depending on the interface software) in the records before displaying it to the user.
The Z39.50 protocol first appeared in 1988, followed by version 2 in 1992 and version 3 in 1995. But even the improvements presented by each version did not stop critics from slating the protocol. They pointed out that Z39.50 required a particular type of software to be installed, configured and maintained in order to work, it was expensive, and even though there was search and retrieval the protocol
did not allow data exchange [3]. Due to this drawback researchers were hoping that newer technologies like XML and RDF would fill the gaps left by the Z39.50 protocol.

The OAI-PMH - Open Archives Initiative Protocol for Metadata Harvesting

The aim of the OAI is to allow Web-accessible repositories to interoperate by sharing, publishing and archiving one another’s metadata records [13]. One of the OAI projects is the OAI-PMH, a protocol that provides an application-independent interoperability framework based on metadata harvesting [13]. There are two actors in this framework: the data providers and the service providers. A data provider uses the OAI-PMH to expose metadata about the repository content to service providers [9]. A service provider uses OAI-PMH to harvest metadata from data providers and use the metadata to provide value-added services [9]. OAI-PMH allows communication between data and service providers via HTTP (Hypertext Transfer Protocol) and XML [13]. To provide a basic level of interoperability the specifications require the use of the unqualified Dublin Core (DC) format, but individual communities can agree to use any other metadata format.

In the OAI-PMH [13], the service provider harvests exposed metadata records from one or more data providers/repositories. The harvested metadata is often stored in a central database. The value-added services are provided based on the resources of this central database. The protocol does not provide a link between the metadata records and the actual content; it also does not provide a search function for the aggregated data, all it does is to bring the data together into a central location. So in order to provide services the harvester has to be combined with other mechanisms. This protocol aims to:

1. Surface hidden resources,
2. Provide access to the archives’ metadata,
3. Provide low cost interoperability, and
4. Support a new pattern for scholarly communication by providing a world-wide consolidation of scholarly archives.

While the protocol itself is technically simple, the complexity is in building services that satisfy user needs. Version 2 (preceded by Santa Fe convention and OAI-PMH v.1.0/1.1) is the latest version of this protocol.

RSS – Really Simple Syndication

RSS is a Web content syndication format written in XML (Extensible Markup Language) [15]. Websites with constantly changing content syndicate their content as an RSS feed to interested users [23]. An RSS aggregator regularly checks sites to search for updates and delivers them to users who have subscribed to the feed or simply displays them to anyone who is interested [27]. Version 0.9 was created in 1999 by Netscape; the latest is version 2.0 by UserLand. It uses an XML file that follows the specifications of the RSS protocol [22]. To have an RSS feed a webmaster creates an RSS file (save as .xml), uploads the file to the website and then optionally registers the feed with an RSS aggregator [6]. The client or aggregator does a daily check of the RSS file on the website, verifies the links and then displays a summary or a snapshot of the content for easy access by clients. RSS files can be automatically updated. Web pages that have an RSS feed may place an RSS or XML icon on the pages to make users aware of it. By clicking on the icon users get a list of links with notifications of the updates to the website.
One of the benefits of RSS feeds is that it saves the time that the user would have wasted in case he/she needs to go through a website looking for new content, and also it allows the user to subscribe only to feeds of his/her interest. Because RSS data is small and fast-loading it can be used with smaller devices like cell phones and PDAs [6].

RSS is one of the most popular ways of sharing information on the Web, and is used in news websites and blogs [28], but that has not stopped it from being criticised. According to the ISS (Internet Information Services) [21] RSS is a “frozen” specification and some of its limitations are: a fixed namespace, it only recognises an <rss> type document and therefore cannot be used outside the RSS context and the failure to differentiate between plain text and escaped HTML.

**ATOM Syndication Format**
The Atom syndication format is an XML based feed format for representing lists of related Web resources known as feeds [1]. Feeds contain a number of items individually known as entries. Each entry has an extensible set of metadata that conform to the protocol specifications. Atom’s primary use case is the syndication of Web content such as news headlines to websites and directly to users [2].

This protocol was developed by the AtomPub group in an attempt to fill the gaps left by RSS they hoped Atom would be the successor of RSS [27]. Believed by some [21] [14] to be superior to RSS in terms of content model, partial content, auto discovery, format flexibility, extensibility, levels of encryption and modularity, Atom has yet failed to overtake its predecessor. This is attributed to facts such as [24]:

- RSS was introduced first and has become almost a synonym for Web syndication;
- RSS 2.0 support for enclosures led to the development of podcasting and it is still supported by most podcasting applications;
- Big news websites (like CNN, BBC and New York Times) only use one feed format and choose RSS over Atom;
- Websites that publish their feeds using Atom usually also use RSS feeds; and
- RSS satisfied users see no reason to switch to Atom feeds.

**APP – ATOM Publishing Protocol**
APP is an application level protocol used to generate and edit Web resources of constantly updated websites, like news websites [19]. This XML based protocol uses HTTP to transfer Atom-formatted representations [5]. These Atom-formatted representations describe the state and metadata of the Web resources. APP works on Web resources that support the GET, POST, PUT and DELETE verbs from HTTP, to provide facilities for:

1. Collections: search for whole or partial sets of resources,
2. Services: finding collections and their corresponding descriptions, and
3. Editing: creating, editing and deleting resources.

The APP is the publishing protocol developed by the AtomPub group. An example use of the APP would start with a client finding a collection (it does so by locating and requesting an introspection document from the server). After the collection is located the client requests a list of the members of the collections. The server responds with an Atom feed document that lists the collection membership. After receiving the list, the client can then add entries, read entries, delete entries and update entries in the collection, in other words edit the collection as necessary.
**SWORD – Simple Web-service Offering Repository Deposit**

SWORD is a lightweight protocol for depositing content from one location to another [18]. SWORD is a profile of the APP, and its aim is to lower barriers to deposit. Its main focus is on depositing content into repositories, but this can potentially be used to deposit content into any system that is willing to receive it.

With SWORD depositing is a two-stage process.

1. First an authenticated user sends a request to the implementation of the service document. This returns a list of collections in which the user is allowed to make deposits.
2. After receiving the service document the user can then deposit files into the collection. Things like lack of authentication and unacceptable file format can cause the repository to send an error report, but if all goes well the repository sends a successful message.

This can only be used with repositories that support SWORD, and because SWORD is only a profile of APP it does not support all the functionalities of APP.

**SRW – Search/Retrievable Web service**

SRW [26] is a service for search and retrieval of Web resources across the Internet. This protocol uses a SOAP (Simple Object Access Protocol) interface and the Common/Contextual Query Language (CQL). SRW’s aim is to promote interoperability between distributed databases by providing a common utilization framework [17]. The protocol can be used in two ways [16]:

1. Via SOAP (called SRW), or
2. As parameters in a URL (this form is called SRU)

SRU (Search/Retrieval via URL) is a standard search protocol for Internet search. SRU employs a URL interface, and like SRW it uses CQL. CQL is a formal language for representing queries to information retrieval systems [20]. Known as Common Query Language in version 1.1 it was later changed to Contextual Query Language in version 1.2.

SRW and SRU are XML based and support three main functions [16]:

1. **Search**
   The most important function of these protocols is the search function. This is done using the SearchRetrieve operation, in which the client sends a SearchRetrieveRequest and the server responds with a SearchRetrieveResponse which is a list of XML records and the full count of the number of records that matched the query.

2. **Browsing** (done via the scan request)
   SRW/U also provides the Scan request, which allows the user to browse through the information in the records one by one. The scan request returns a portion from the sorted list of terms in the database for a given index.

3. **Server capability**
   The final operation provided by this protocol is a server capability request which allows the client to find out which protocols the server supports and what it supports in terms of CQL.

There are many other interoperability protocols available; it is not possible to discuss all of them in this proposal document. But these are among the most popular standards.
3. Research Question
The goal of this study is to re-engineer popular interoperability protocols and develop a set of specifications that can potentially improve efficiency by simplifying communication between data providers. The research question that will shape this study is the following:

Is it possible to develop a uniform suite of simple and efficient interoperability protocols to improve on the current medley of protocols?

It is important to build a more efficient model for communication between heterogeneous systems, but after successfully designing an interoperability model it is even more important to be able to prove that the model designed does in fact bring some improvement and show at what level the improvement is observed.

4. Methodology

4.1 Step 1
Thoroughly study the background of interoperability protocols.

4.2 Step 2
The next step in the study is to conduct an online survey. The survey will generate information about the protocols from experts who have implemented them. This will be done by designing a questionnaire and getting programmers who have already implemented any of the protocols being investigated to answer the questions. The results will help to better understand the best features and/or the shortcomings of the protocols (if any), and get their expert opinion, which will allow informed decisions to be made for the design of the protocol for this study. This will be the first iteration with possible users.

4.3 Step 3
To answer the research question it is necessary to know how the current protocols work. This will be done by creating prototypes (implementing) of the different protocols. By doing so it will be possible to:

- Determine which one is the simplest protocol to implement and why;
- Determine if there is such a thing as too much simplicity, so much so that the standard actually leaves out important features;
- Determine which protocols are so complicated to understand/implement that people either give up on them or never even bother to try;
- Determine what are the flaws in the specifications of the protocols;
- Investigate how unpopular protocols can be improved by reducing the complexity on the client side at the cost of increasing the complexity on the protocol designer’s side;
- Find out if there are unused (deemed unnecessary) functions that make the specifications more complicated than they really need to be;

This is a good way to get hands-on experience that will prove or disprove the findings from steps 1 & 2.
4.4 Step 4
Choose which repository types to use. Decide what type of data services the study will be providing interoperability for. It can be for DLs, Web servers, etc.

4.5 Step 5
Using the knowledge from steps 1, 2 and 3 design and implement a simpler and more efficient set of specifications for what can be a more usable communication protocol for interoperability amongst systems. This phase will also include user iterations, one during the design, to get feedback on the proposed features, and again during the implementation in order to check that the design synchronizes with the implementation.

4.6 Step 6
And finally test/evaluate the new specifications. This may be the most difficult part of this research because existing protocols are evaluated by a large group of users who implement them in their websites, repositories, etc and then give the developers useful feedback. This type of evaluation can take a long time and usually some convincing to have designers using the protocol, and that kind of time span is not possible for this project. Therefore a method of evaluating communication protocols in controlled environments in a shorter period of time should be applied here. This is the step where the final user iteration occurs. For usability evaluation randomly chosen users can perform a set of pre-defined tasks in a controlled environment and the feedback will come from observation by the researcher combined with the answer from a questionnaire that they will be given after performing the designated tasks. Efficiency and simplicity evaluation on the other hand require the participation of experts, and for that reason there is a good chance that this evaluation will be conducted online, so that users can do it on their own time and have a chance to explore the protocol.

5. Expected Outcomes
This study is expected to provide evidence that it is possible to improve on the efficiency and usability of existing interoperability protocols without sacrificing on functionality; and as a result produce a set of experimental specifications for simpler and more efficient communication that may increase interoperability, by allowing data providers to exchange information in a seamless manner.

6. Work detail

6.1 Resources required
1. A standard PC (available in the DL lab).
2. Access to the CS departmental publications repository. In order to test the communication protocol I will need to replicate the departmental repository.
3. Experts in the implementation of communication protocols, for the user iterations. It is important to work with experts because novices would not have a deep understanding of all aspects of a protocol, which is a crucial aspect for an efficiency evaluation.
6.2 Deliverables
Communication protocol specifications (working).

6.3 Timeline (see appendix 1)
- Literature survey and prototype design – From the 07\textsuperscript{th} of April to the 12\textsuperscript{th} of September
- Online survey (\textit{user iteration 1}) - From the 1\textsuperscript{st} of June to the 12\textsuperscript{th} of September
- Research proposal – 22\textsuperscript{nd} of June 2009
- Background chapter - 12\textsuperscript{th} of September 2009
- System design – From the 13\textsuperscript{th} of September to the 30\textsuperscript{th} of November
  - \textit{User iteration 2}
- System implementation – From the 1\textsuperscript{st} of December to 11\textsuperscript{th} of February
  - \textit{User iteration 3}
- Evaluation – From the 12\textsuperscript{th} of February 2010 to the 31\textsuperscript{st} of March
  - \textit{User iteration 4}
- First Draft - 12\textsuperscript{th} June 2010
- Final draft - 1\textsuperscript{st} of August 2010
- Thesis Submission - 12\textsuperscript{th} August 2010

7. References
http://www.atomenabled.org/developers/syndication Last accessed on the 05\textsuperscript{th} of May 2009.
http://www.atomenabled.org/developers/syndication/atom-format-spec.php Last accessed on the 05\textsuperscript{th} of May 2009.
http://www.chin.gc.ca/English/Standards/interchange_standards.html Last accessed on the 03\textsuperscript{rd} of May 2009.
[8] JISC Information Environment Architecture Glossary. At: 
http://www.ukoln.ac.uk/distributed-systems/jisc-ie/arch/glossary/ Last accessed on the 03\textsuperscript{rd} of May 2009.


## Appendix 1: Gantt chart

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Duration</th>
<th>Start</th>
<th>Finish</th>
<th>Predecessors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Literature survey and prototypes design</td>
<td>114 days</td>
<td>Tue 09/04/07</td>
<td>Fri 09/09/11</td>
<td></td>
</tr>
<tr>
<td>Online survey (user iteration 1)</td>
<td>75 days</td>
<td>Mon 09/05/01</td>
<td>Fri 09/09/11</td>
<td></td>
</tr>
<tr>
<td>Research proposal</td>
<td>35 days</td>
<td>Tue 09/05/05</td>
<td>Mon 09/06/22</td>
<td></td>
</tr>
<tr>
<td>Background chapter</td>
<td>85 days</td>
<td>Fri 09/05/15</td>
<td>Fri 09/09/11</td>
<td></td>
</tr>
<tr>
<td>System design</td>
<td>9 days</td>
<td>Tue 08/11/10</td>
<td>Fri 09/11/20</td>
<td>1,2,4,3</td>
</tr>
<tr>
<td>User iteration 2</td>
<td>9 days</td>
<td>Tue 09/11/10</td>
<td>Fri 09/11/20</td>
<td></td>
</tr>
<tr>
<td>System implementation</td>
<td>11 days</td>
<td>Fri 10/01/15</td>
<td>Fri 10/01/29</td>
<td>5</td>
</tr>
<tr>
<td>User iteration 3</td>
<td>11 days</td>
<td>Fri 10/01/15</td>
<td>Fri 10/01/29</td>
<td></td>
</tr>
<tr>
<td>Evaluation</td>
<td>34 days</td>
<td>Fri 10/02/12</td>
<td>Wed 10/03/31</td>
<td>7</td>
</tr>
<tr>
<td>Thesis first draft</td>
<td>204 days</td>
<td>Tue 09/05/01</td>
<td>Fri 10/06/11</td>
<td></td>
</tr>
<tr>
<td>Thesis final draft</td>
<td>230 days</td>
<td>Tue 09/05/01</td>
<td>Sun 10/08/01</td>
<td></td>
</tr>
<tr>
<td>Thesis submission</td>
<td>245 days</td>
<td>Tue 09/05/01</td>
<td>Thu 10/03/12</td>
<td></td>
</tr>
</tbody>
</table>

![Gantt chart image](image-url)