Extending the Open Journals System OAI repository with RDF aggregation and querying (African Journals Online)

Ed Crewe

University of Bristol
Institute of Learning and Research Technology,
8-10 Berkeley Square, Bristol BS8 1HH, UK
ed.crewe@bristol.ac.uk

Abstract. This paper reports on the progress of an extension to a journal publishing system (OJS) to add RDF aggregation. The system is a lightweight open source repository solution with full CMS features for devolved publishing, a new semantic module is in development for enhanced navigation, searching and for content re-purposing.
African Journals Online is used as the example live installation of the extended OJS. Further functionality is being developed to aggregate the large existing metadata resource as a RDF Store, which can be queried using RDQL, to uncover underlying data relationships expressed via formats such as FOAF, or for reuse via RSS.
In order to preserve data relationships when harvesting the existing OAI:PMH interface now offers RDF format metadata – and an extension of the set support within the OAI protocol has been prototyped.

1. Introduction

There are well established standards for appropriate metadata formats for various content types held in repositories, and the Open Archives Initiative protocol for metadata harvesting (OAI-PMH) [1] is the most widely accepted standard for exposing and retrieving them. However OAI has evolved around the concept of a flat record archive (e-prints) and fixed metadata schemas for record level description.
This is very limiting for communicating hierarchical structures and uniquely identifying resources across records, repositories and the internet in general.

To address this limitation the Resource Description Framework (RDF) [2] can be partitioned via OAI, to provide a flexible means of uniquely identifying resources and expressing their interrelationships. Its flexibility also makes it ideal for uncovering new relationships and reusing content.

African Journals Online (AJOL), a repository holding many hundreds of journals, was chosen to prototype the provision of content navigation and reuse by RDF exposed via OAI. AJOL is an extended version of the Open Journal System by PKP [3]. OJS offers an easily installed system for devolved publishing of a journal. It comes with an OAI interface and a separate OAI harvester.

This paper describes the progress of the creation of an RDF module for AJOL, along with outlining possible future work.

2. RDF Module Architecture

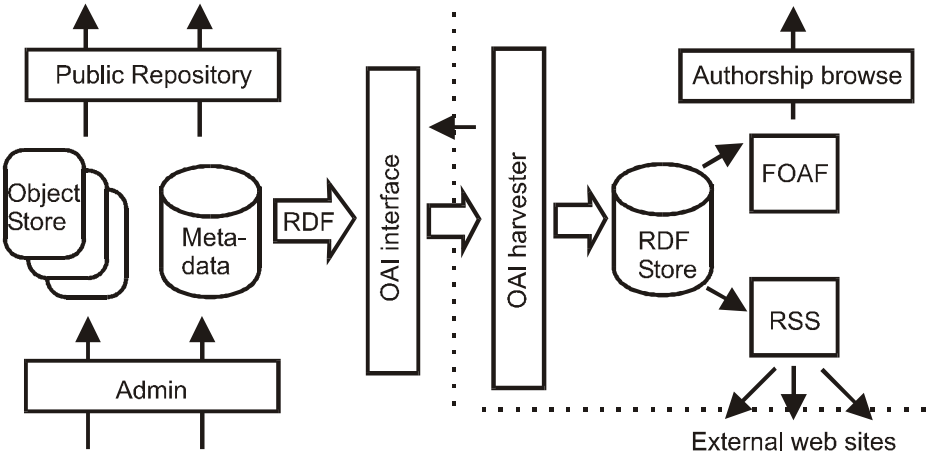

Fig. 1. A schematic of the repository emphasizing the RDF module components.

The original OJS system contains a range of modules to enhance the functionality. These include the OAI interface and the Research Support Tool utilities.

The RDF functionality is added to the extended OJS system in a similar manner via an RDF module. The RDF module uses the RAP RDF library for PHP [4] with some extensions. These extensions are mainly functions to allow faster cross model querying, and cross linking of models for navigation.

The system works on the principle of harvesting the metadata available from OJS into an RDF Store either as a bulk initial population function, or there after, whenever issues are published. This Store then forms the basis for the content reuse and navigation via extraction of a range of established RDF vocabularies.

The extracted vocabularies so far investigated being RDF Site Survey (RSS) to allow for syndication of content to other sites, and Friend Of A Friend (FOAF) to provide for people centric analysis of the information, since this is particularly relevant to academics in terms of professional collaboration.

The Store itself is made available for direct harvesting via RDF models wrapped in OAI:PMH in order to maintain the data relationships that might other wise be lost if standard publication centric OAI Dublin Core were used.

2.1 RDF Aggregation

RDF is fundamentally a directed labeled graph of relationships which can be broken down in to statements, or triples, of the form subject predicate object. Within the RAP framework partitions of RDF are referred to as models. Models can be created as in memory models for population by loading RDF documents or by a variety of APIs (interfaces) for statement creation and manipulation. They are persisted by placing them in the RDF store. Models can be output as graphs or serialized in a number of formats, e.g. RDF/XML or N3 notation.

The partitioning into documents in this case was determined by three priorities. The main priority was that the important parts of the metadata could be extracted and reused. A secondary concern was being able to replicate the informational model of a journal, whilst embedding the appropriate sub-models; the third concern was to arrive at models that would work well in terms of size and number with the RAP framework. [1]

Two basic ways to partition the metadata were used. The first was at the journal level which included cross journal categorisations and relevant editor information. Each of the journal documents linked to documents for each issue that in turn held article level and author information. At this stage the information concerning readership, CMS data, statistics and AJOL specific features was left out of the RDF Store, partly to simplify things and partly due to it being inappropriate for public exposure.

The RDF/XML representation of a simplified form of the journal model is boxed in Fig. 2. The main features that are of note are the use of standard vocabularies [5] for expressing the metadata in RDF, wherever possible. The vocabularies used for the representation of all information going into the store were limited to,

vCard: http://www.w3.org/2001/vcard-RDF/3.0#

Dublin Core: http://purl.org/dc/elements/1.1/

Dublin Core Terms: http://purl.org/dc/terms/

With just a couple of insignificant metadata elements using an implementation specific vocabulary: ajol: http://www.ajol.info/.

The reason that the perhaps rather more outdated vCard schema was chosen over FOAF for initial storage being that the limited vCard expression of a person's details was more suitable to this embedded context. However the expression of vCard details in RDF was in a manner easily extractable to FOAF, based on an adaptation of past W3C recommendations [6].

---

[1] The RAP library is limited by PHP's nature as a runtime compiled, web based scripting language – restricting both speed and memory (the latter is adjustable in the PHP installation configuration file). Hence a model should contain less than a couple of thousand triples and the number of models should be minimised to reduce the necessity for less efficient cross model querying. The published data held in AJOL constitutes around half a million triples.

3. OAI Harvesting

The information provided by the OJS system is largely made available by the existing OAI [1] interface. However by default this interface provided little or no journal level information (since OJS was developed to hold a single journal rather than numerous journals). Perhaps more importantly the data that is returned via the default OAI schemas is based upon the origins of OAI with e-prints, and other non-hierarchical repositories, where virtually all the metadata is attached to the concept of an independent record which tends to be associated with the smallest publication unit – ie. an academic paper.

The problem associated with this is that the value of RDF modeling is in mapping all the relations between data and obtaining additional information from these relations along with those derived from aggregation of information followed by some 'smushing' (identity reasoning) [7] of a number of sources together. In the case of OJS information concerning people is held about registered users, authors and journal contacts in different forms. These are extracted, assigned a URI and expressed via the vCard vocabulary. People can then be related to resources with statements such as articleURI . creator . personURI.

From an RDF perspective, therefore it is best to expose the maximum amount of relations that are held within a repository, and can be smushed from it, to harvesters. The standard OAI DC and related XML metadata schemas provided via the OAI interface tend not to express many of these relations and hence they would be lost if harvested as OAI DC and re-expressed as RDF.

To avoid this loss of information a further metadata format has been added to the OAI interface specified by rdf_dc, i.e. RDF largely using Dublin Core predicates [5]. This expresses all metadata in RDF for direct OAI harvesting to an RDF Store.

OAI version 2.0 [1] provides some guidelines on using the optional sets description with colons to express more hierarchical relationships. These have been used and extended. The two extensions are a new optional command of metadataPrefix for ListSets and the addition of a new verb, GetSet to aid harvesting. By simply omitting the format specifier for ListSets (or refraining from using GetSet) the system retains compliance with OAI 2.0. Specifying rdf_dc results in the description tags being populated with full RDF for the set, rather than the default oai_dc (see Fig 1.).

It can be envisaged that this means of expressing complex models with OAI could lend itself to numbers of applications where a significant packet of metadata (rather than just categorisation information) applies to large subsets of records.

This perhaps cavalier usage of OAI:PMH to provide full exposure of the RDF model may obviously need major revision as standards evolve, but the functionally serves to allow complete and direct aggregation of the RDF models of a number of extended OJS systems, and could serve as a model for RDF delivery via OAI where maintenance of the relationships between personnel, publication and set based information is required. Other developing areas of OAI such as OAI-Rights have recently proposed both the use of embedded RDF and its use at the sets level to provide more complex metadata concerning access rights [8].

Fig. 2. The GetSet request and response encapsulating a simplified version of the Journal level RDF (boxed with the vCard sub-model double line boxed). http://www.ajol.org/oai/?verb=GetSet&identifier=oai:ajol.www.ajol.info:journal-2&metadataPrefix=rdf_dc

```
<?xml version="1.0" encoding="UTF-8" ?>
 <OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
<responseDate>2005-04-19T20:40:26Z</responseDate>
<request verb="GetSet" idenfier="oai:ajol.www.ajol.info:journal-2"
metadataPrefix="rdf_dc"> http://www.ajol.info/oai/</request>
<GetSet><set>
<setSpec>journal-2</setSpec>
<setName>African Journal of Livestock Extension</setName>
<setDescription>
 <rdf:RDF xml:base="http://www.ajol.info/"
   xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xmlns:dcq="http://purl.org/dc/terms/"
   xmlns:xml="http://www.w3.org/XML/1998/namespace"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

<rdf:Description rdf:ID="vcard.php?uid=oyesola.dr.l.a"
   vcard:FN="Dr. L.A.Oyesola"
   vcard:Locale="University of Ibadan, Ibadan, NIGERIA">
   <vcard:Orgname="Faculty of Agriculture &amp; Forestry">
   <vcard:TEL rdf:resource="tel:+27123286970"/>
   <vcard:TEL rdf:resource="fax:+27123238153"/>
   <vcard:EMAIL rdf:resource="mailto:vichenfel@yahoo.com"/>
</rdf:Description>

<rdf:Description rdf:ID="journal_index?jid=2"
   dc:format="text/html" dc:type="Collection"
   dc:source="http://www.ajol.info/"
   dc:publisher="University of Ibadan"
   dcq:isReferencedBy="issn 1596-4019"
   dc:title="African Journal of Livestock Extension"
   dc:subject="livestock,  agriculture, Agricultural Sciences and-
   Management">
   <dc:creator rdf:resource="#vcard.php?uid=oyesola.dr.l.a."/>
   <dcq:spatial
   rdf:datatype="http://purl.org/dc/terms/ISO3166">NGA</dcq:spatial>
   <dcq:created rdf:datatype="http://purl.org/dc/terms/W3CDTF">
2002-01-01</dcq:created>
   <dcq:available rdf:datatype="http://purl.org/dc/terms/W3CDTF">
2004-02-11</dcq:available>
   <dcq:abstract><![CDATA[African Journal of Livestock Extension
aims to bring to the fore the role and significance of livestock in
maintaining rural  peri-urban and urban housholds.]]></dcq:abstract>
    </rdf:Description>
   </rdf:RDF>
 </setDescription>
 </set>
</GetSet></OAI-PMH>
```

4. Content Reuse

4.1 Browse Interface

In order to navigate the RDF view of the repository there is an interface derived from the tools available within RAP with some minor extensions. This allows for navigation through the site from the standard journal centric navigation screen to linked tabular displays of RDF models available in a range of serialized or image formats.

Retrieval of models that equate to the whole of a document from the RDF store is trivial. Other models are built by combining the result statements of RDQL querying across a number of documents.

4.2 Querying the Store

An example of useful information derived from the store that is otherwise difficult to gather via the existing OJS system search methods, is that expressed in FOAF models. Currently FOAF has just been applied to express co-authorship details within the repository, however the relationships with editors and signed up readers could be similarly derived. With appropriate metadata, publication references could also be used.

4.2.1 Example Querying and RDF Re-Expression Software Process

A function was written to generate the FOAF [9] in memory model from the RDF store. The function takes arguments of person URI, vocabulary map and recurse number. The map gives vCard to FOAF translation and recurse specifies the level of FOAF:Knows connections (in this case indicative of co-authorship) to follow.

The software process of returning FOAF results is at follows:

1. Firstly the RDF journal or issue document URIs for use in RDQL queries are gathered by querying the store passing person URI as a resource to match against subject.
2. For each document the RDQL query is applied to pull out the statements that relate to the person URI.
3. Next the articles authored by this person are found and a query built up to extract the desired information from this model, by stepping through each article and adding a clause for it:

```
SELECT ?article ?title ?ref ?person
FROM <http://www.ajol.info/RDF/viewissue.php?jid=47&id=1983>
WHERE
(?article <http://purl.org/dc/elements/1.1/title> ?title)
(?article <http://purl.org/dc/terms/isReferencedBy> ?ref)
```

```
(?article <http://purl.org/dc/elements/1.1/creator> ?person)
/* added clause 1 */
(<http://www.ajol.info/viewarticle.php?id=15427>
<http://purl.org/dc/elements/1.1/creator> ?person)
/* added clause 2 */
(<http://www.ajol.info/viewarticle.php?id=19080>
<http://purl.org/dc/elements/1.1/creator> ?person)
```

4. RAP doesn't yet provide multiple document querying in RDQL so each document is looped through and the results from the queries in steps 2 and 3 are appended to a total results set and to a list of creators. The total combined result set is then converted to a new memory model and foaf:Knows statements added for each creator.
5. Lastly if recursion is specified the recurse number is decremented and the function calls itself for each of the creators.
6. Finally the model is returned and passed to display functions.

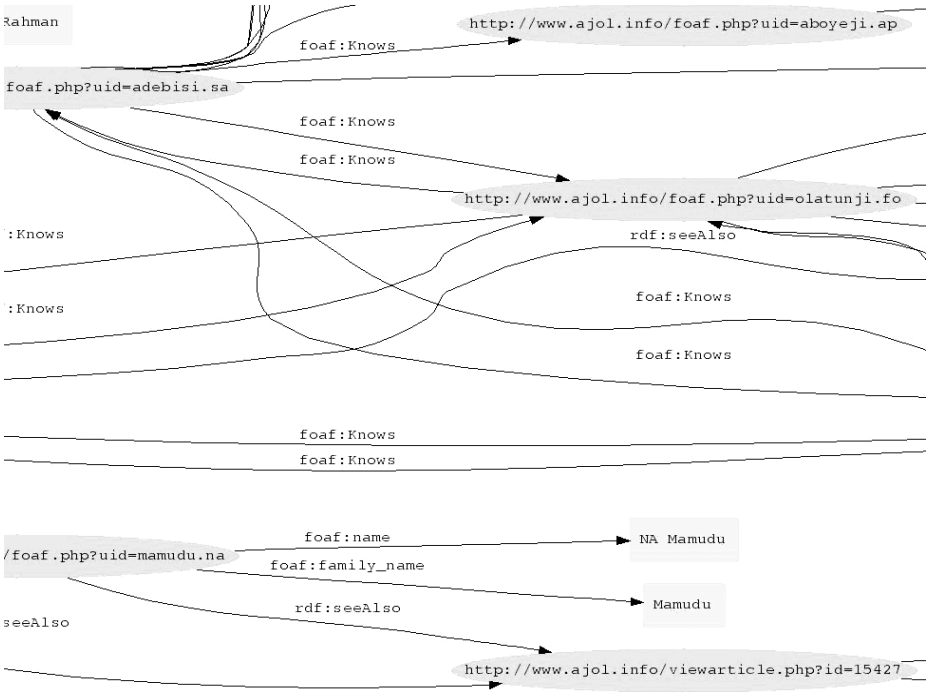

Fig. 3. Display of a section of the FOAF mapping of co-authorship with one level of recursion indicating the complexity of the derived network of authorship.

4.2.2 FOAF browsing

The FOAF function yields 14 direct links to Rahman, however if we ask to follow foaf:Knows links to a depth of 1 level then the resulting model balloons to 59 statements (see Fig. 3), largely through one of Rahman's co-authors, Adebisi, being more prolific. Another level gives 222 statements.

In SVG viewing mode the directed graphs nodes can be linked on to the next graph. This tool can be refined to be a browse and zoom browser listing summary metadata from coauthored resources. Resources relating to academic specialisms can then be navigated irrespective of journal origin or keyword based searches.

4.3 RSS Feed generation

By default the system provides an RSS feed based on the currently published issue for each journal. The feed uses the RSS 1.0 standard [10], with elements from the PRISM RSS [11] module created for use with academic publications. This provides a syndicated content alternative to the existing system where readers sign up to a

range of journals to receive email alerts whenever new issues are published. A readers home page now connects to one displaying links to their chosen news feeds.

However this is just a first step, for authorised users, tools are being developed to allow the selection of content across journals to be delivered via RSS for populating other web sites. These involve developing an interface for easy creation of the appropriate cross journal queries and subsequent storage of them as news feed generator configurations. An example configuration might be this months South African medical papers (or their author's details), matching a keyword set. The search functionality of the system could also be more directly integrated to deliver search results via RSS according to the A9 OpenSearch™ RSS module for search engine results syndication.

5. Future Developments

Of first priority are refinements to improve the functionality of the RSS based content reuse interface to deliver a seamless addition to the existing system. Feeds could provide person or organisation based information as well as publications.

In terms of the infrastructure of the system, work needs to be done in improving the smushing of people and organisations. New functionality could be added to utilize the opportunities for semantic based searches. Both of these functions could be served by use of the RAP inference and ontology models.

On a more general note, the concepts that have been rapidly prototyped with the php based OJS system could be applied using more scalable RDF frameworks to the open source repository heavyweights such as DSpace and Fedora.

Lastly the experimental use of OAI:PMH to deliver more complex interrelated metadata models, via RDF, and an extension to the sets functionality of OAI could be further investigated and put to the OAI Community.

References

1. Lagoze, C., Nelson, M., Van de Sompel, H., Warner, S.: The Open Archives Initiative Protocol for Metadata Harvesting v2.0 (12 Oct 2004) http://www.openarchives.org/OAI/openarchivesprotocol.html
2. Beckett, D., McBride, B.: RDF/XML Syntax Specification (Revised) W3C Recommendation (Feb 10 2004) http://www.w3.org/TR/2004/REC-RDF-syntax-grammar-20040210/
3. Willinsky J.: The Nine Flavours of Open Access Scholarly Publishing (2003) http://www.pkp.ubc.ca/publications/
4. Bizer, C, Oldakowski, RAP: RDF API for PHP (2004) http://www.wiwiss.fu-berlin.de/suhl/radek/pub/RAP-oldakowski.pdf
5. Kokkelink, S., Schwänzel, R.: Expressing Qualified Dublin Core in RDF / XML (15 May 2002) http://dublincore.org/documents/dcq-RDF-xml/

6. Iannella, R.: Representing vCard Objects in RDF/XML (22 Feb 2001) http://www.w3.org/TR/vcard-RDF
7. Brickley, D.: RDFWeb notebook: aggregation strategies (2002) http://RDFweb.org/2001/01/design/smush.html
8. Lagoze, C., Nelson, M., Van de Sompel, H., Warner, S.: Conveying rights expressions about metadata in the OAI-PMH framework (5 Nov 2004) http://www.openarchives.org/OAI/2.0/guidelines-rights.htm
9. Brickley, D., Miller, L.: FOAF Vocabulary Specification (3 Apr 2005) http://xmlns.com/FOAF/0.1/
10. RSS-DEV Working Group: RDF Site Summary (RSS) 1.0 (2001) http://web.resource.org/rss/1.0/modules/content/
11. Hammond, T., Hannay, T., Lund, B.: RDF Site Summary 1.0 Modules: PRISM (11 Aug 2004) http://www.prismstandard.org/resources/mod_prism.html