A Technical Review of Open Access Repository Registries

OpenDOAR and ROAR

Cottage Labs on behalf of UKOLN, 1st July 2011
http://cottagelabs.com/
http://www.ukoln.ac.uk/

Note: this version is provided for general distribution without Appendices, which are omitted for privacy reasons

Contents

1 Executive Summary

This document provides a technical review of the capabilities, benefits and drawbacks of two leading Open Access Repository Registries (OARRs) – OpenDOAR and ROAR. Both systems are considered qualitatively and quantitatively with a view to identifying those facets which provide value for a repository registry service.

A methodology is identified to investigate the relative strengths of each system based on four main parameters: rate of growth, software, API capabilities, and the quality of data held in each system.

Interviews were conducted with members of the software development teams from both OpenDOAR and ROAR to provide insight into current working practices and technical roadmap for both systems. The output from these interviews are included below along with detailed investigation of each system. This included developing and using software client libraries in Python to review each API.

Additional interviews were also carried out with two API users to provide qualitative input on each systems usability in relation to a specific use case. These were Repository66 a repository mapping service and OA-RJ a deposit broker tool

Although a direct comparison of OpenDOAR and ROAR is avoided the output is summarized for each system to identify the best and worst aspects. These can be seen as underpinning the shape of a new idealised repository registry.

2 Method

This technical review and analysis used the following process:

1. At-desk technical review of the OpenDOAR and ROAR user interfaces
2. A basic at-desk review of the Directory of Open Access Journals as an external reference point for the reviewer
3. Develop and use software client libraries (in Python) to review the APIs for both systems
4. Conduct in-depth technical interviews with the developers of each system
5. Conduct technical interviews with developers of systems which rely on the OpenDOAR and/ or ROAR APIs
6. Analyse the results of the review process and extract opportunities and best practices that could be realised in this space

The following sub-sections describe in detail some of the methods and techniques employed in this review and analysis.

Note that as this review took place over a period of time, the content of the registries changed a small amount. This may be seen in graphs which indicate the numbers of repositories included in calculating statistics, for example. We do not believe that the changes are significant enough to alter any of the conclusions presented here.

Most of the graphs and tables used in the document are taken directly from the relevant services and represent the view that a user would get on the data via the web interface. For this reason there are some discrepancies in styles and in some cases text is clipped.

2.1 Measuring Growth

We will be looking for growth of each system using the following metrics:

1. Total number of repositories indexed
2. Total number of repository records indexed/identified
3. Daily usage of UI and API

2.2 Software Coverage

We will be looking at how wide the coverage of the directories is, and a key measurement will be coverage of the repository software variants, as this is something that we can assess independently of OpenDOAR and ROAR[1] by looking at the community pages for those software.

We will be using the following baseline data, which only takes into account the most common/ popular software products:

Software	**Count**	**Source**
DSpace	1083	http://www.dspace.org/whos-using-dspace/Repository-List.html
EPrints	407	http://roar.eprints.org/view/software/eprints.html
OPUS	104	http://www.opus-repository.org/anwendung/installationen.html
Digital Commons (bepress)	159	http://digitalcommons.bepress.com/subscriber_gallery/all.html
Fedora Projects (not strictly installations)	173	https://wiki.duraspace.org/display/FCCommReg/Fedora+Commons+Registry

This data can be conveniently visualised as per *Figure 0*.

[1]This is almost true; in reality the software registry for EPrints is held in ROAR by default.

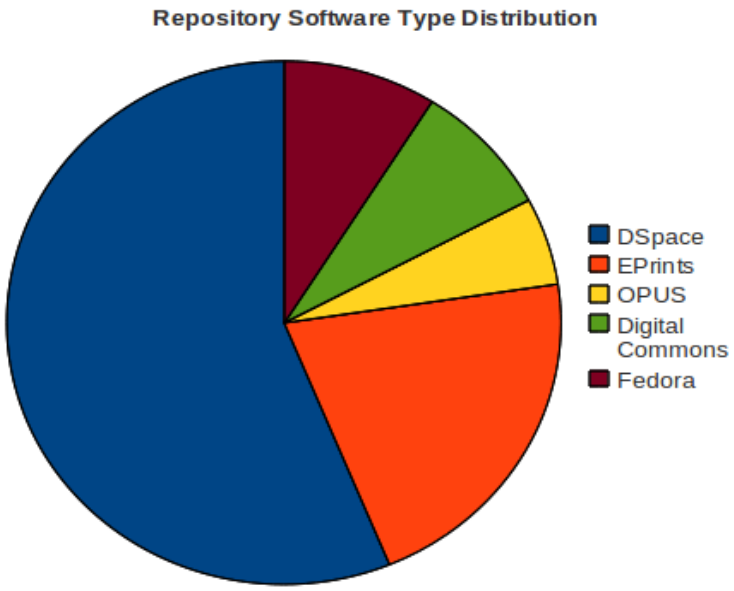

Figure 0: Repository Software Type Distribution

When we use this in the technical review we must be aware of some caveats. The first is that the above list is not the full list of all possible repository types; the directories themselves list upwards of 30 different software platforms. What we have here is only the most widely used repository platforms. Secondly, Fedora is not as easy to identify as the other systems as it often provides a back-end service, and it is mis-represented in both directories; further to that the Fedora community page for installations lists "projects" not "installs" which makes it even harder to be confident about the numbers.

When we review each directory, we will extract the data for these top rated repositories and re-graph their proportions to compare with the benchmark above.

2.3 API Consumers

To gather information about how well the APIs for OpenDOAR and ROAR function we conducted interviews with developers from two systems which rely on both APIs. The interview transcripts can be found in the Appendices. In the main text we summarise both viewpoints together for each registry.

Repository 66: A mash-up of Google Maps with geo-tagged repositories in OpenDOAR and ROAR [8]. It was originally intended as a sales tool for repositories to academics.

Open Access Repository Junction (OA-RJ): A broker tool to assist open access deposit into existing repository services, including facilities to discover the appropriate repository for your content [9].

2.4 Data Analysis

When analysing the data stored in each system, we will consider the following 4 questions:

1. *What data can be held?*

This questions the size and structure of the schema within which the data is stored. How flexible is it? What are the field names, and do they make sense?

2. *How well populated are the fields?*
   This questions the proportion of fields which are populated at all, irrespective of the content of the field and that content's actual quality.
3. *How accurately populated are the fields?*
   This questions whether the content of a field is the correct kind of content for that field. For example, are there things which look like titles in the "title" field? Are there integers in the "size" field, and so on.
4. *How accurate is the data?*
   This questions the actual truth of the data contained in a field. For example, is the title in the "title" field actually the title; or - easier to check - does the URL in the "url" field resolve to a repository.

This set of questions addresses each of the possible stages of error in the data, and becomes correspondingly harder to answer with certainty as we go down the list.

2.5 Client software

In order to test the APIs it was necessary to write a primitive client library with some associated evaluation tools for each of OpenDOAR and ROAR. The code for these clients is available for download [20].

Technical Review

3 OpenDOAR

URL: http://www.opendoar.org/
Software: in-house development, not open source
Programming language: PHP
Technology stack: Linux, Apache, MySQL

3.1 Introduction

The Open Directory of Open Access Repositories (OpenDOAR) was initially launched in 2005 in response to the fact that there was "a number of different lists of repositories and open access archives, but no single comprehensive or authoritative list which recorded the range of academic open access repositories" [1]. They also set out to provide details about repositories for end-users such as "clear information on their policies regarding tagging peer-reviewed/non-peer-reviewed material, their subject coverage, the constituency they draw on for content, their collection and preservation policies, etc" [1]. It is run and maintained in the Centre for Research Communications at the University of Nottingham which runs the SHERPA [2] services, including RoMEO [3] and JULIET [4].

The approach that OpenDOAR takes to its content curation is one of manual intervention: all

submissions are vetted by staff before entering the registry, and the records are manually populated and updated (where appropriate). The inclusion policy which a repository must meet before being incorporated into the directory is "sites that wholly embrace the concept of open access to full text resources that are of use to academic researchers" [1]. This excludes "sites where any form of access control prevents immediate access" [1] and "sites that consist of metadata records only" [1].

Further to providing tools and services around an open access registry, OpenDOAR also provides some tangential support in the form of the Policies Tool [5] which is their response to a percieved problem that repositories do not, as a general rule, have well defined or easily accessible policy information [6].

3.2 Growth

As *Figure 1* shows, the growth of the total number of repositories held in OpenDOAR has climbed steadily and linearly since mid-2006. The step which appears from early to mid 2006 is an artifact of the data due to re-engineering work which prevented updates to the system for the period. Then, in mid-2006 the backlog of data was cleared and the current trend began.

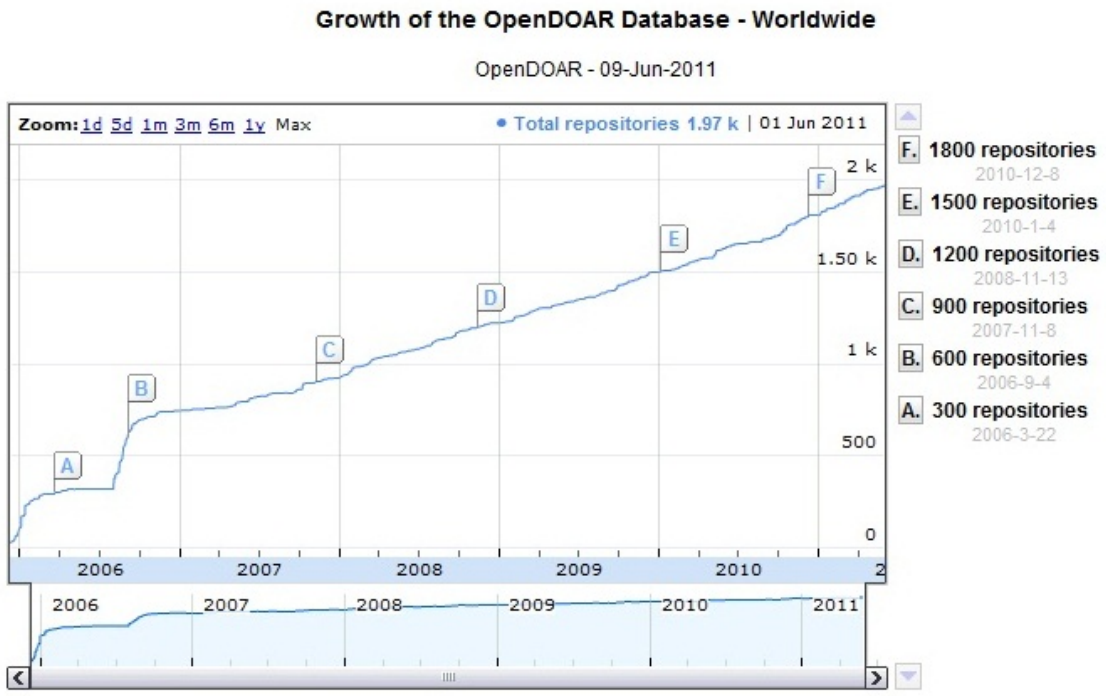

Figure 1: Growth of OpenDOAR Database - Worldwide

This graph shows an average rate of acquisition of around 300 repositories per year, with a dataset of 1972 repositories. In comparison, the DOAJ is growing at a rate of approximately 2500 journals per year, with a dataset of 8617 journals [7]. Therefore a key question to ask is whether the rate of acquisition of each of these systems represents a maximum throughput or a reflection of the rate of growth of the space.

OpenDOAR believes that the rate of registration is now approximately a reflection of the rate of growth of the sector. They do not have much in the way of backlog, and most backlogged items are due to queries about the entry, so there is no throughput issue. UK registrations are also starting to flatten out, which might suggest coverage there is approaching complete (see *Figure 2*). New registrations in established countries - such as the UK - are also increasingly non-bibliographic, such as datasets.

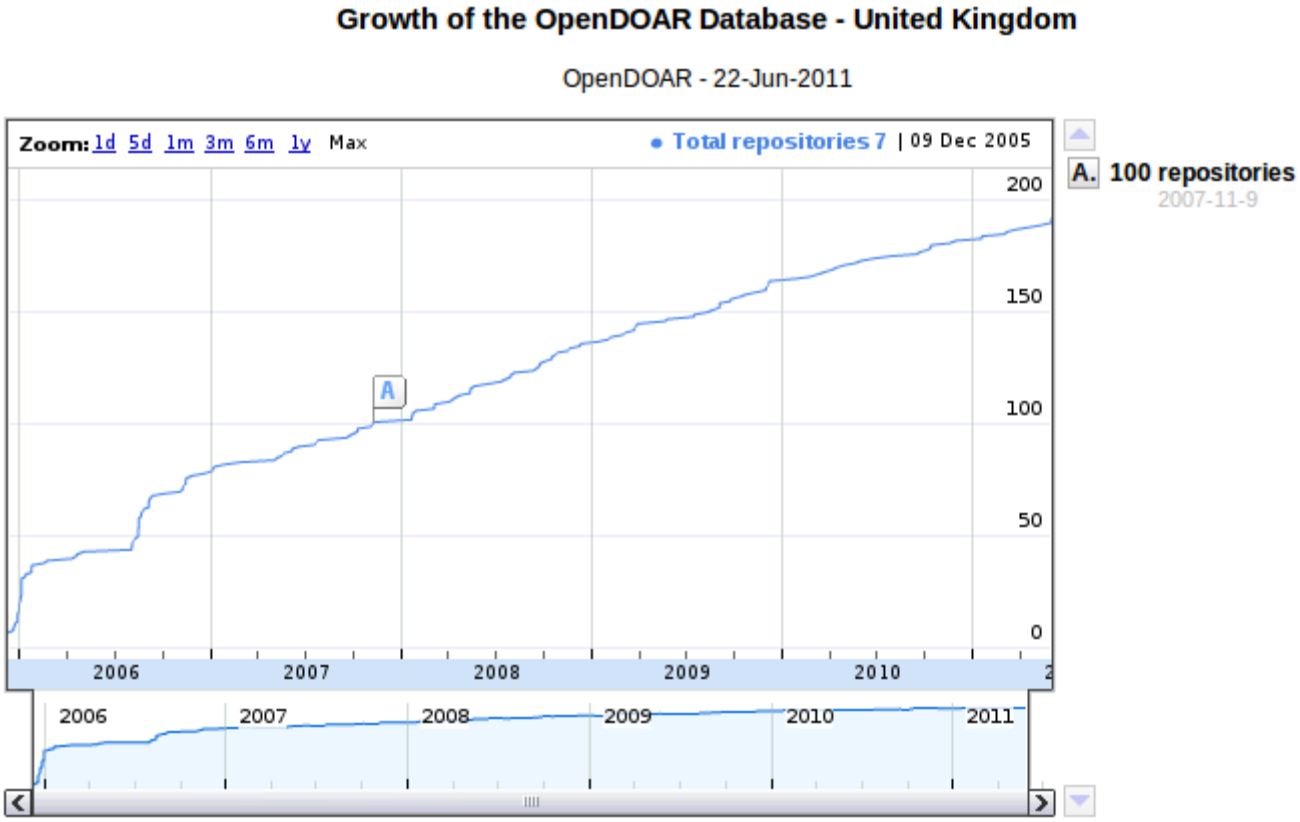

Figure 2: Growth of the OpenDOAR Database - United Kingdom

It is worth noting that ROAR both allows simultaneous registration in both ROAR and OpenDOAR and also synchronises from OpenDOAR periodically. Therefore, it is meaningless to compare and contrast the acquisition rates for the two systems, as they largely share datasets.

Growth of the service in terms of usage is difficult to guage. OpenDOAR do not make their usage statistics publicly available, so it has not been possible to analyse them. There are a number of services which are known to use the OpenDOAR API, though:

- Repository66 [8]
- Open Access Repository Junction [9]
- Institutional Repository Search [10]
- BASE - Bielefeld Academic Search [11]

This would indicate at least some uptake.

3.3 Coverage

The geographical scope of OpenDOAR is global, and it will accept any open access repositories which meet its collection policy [1]. In this section we present a series of charts taken from the OpenDOAR website [12] which provide some insight into its overall coverage. A large quantity of information is available via the charts page, which makes analysing the OpenDOAR content relatively straightforward.

OpenDOAR indicate that there are variations in the extent and quality of coverage depending on the part of the world. There are language barriers to inclusion, such as for Russian repositories, as well as infrastructural barriers (for example, connectivity), such as for African repositories.

This section demonstrates that coverage is not particularly skewed towards any country, software, repository type, etc; the statistical distributions of the coverage appear to be in-line with what you would expect globally. What it does *not* show is whether the coverage is complete. See the section **Growth** for details about the size of the dataset; without further authoritative data about the actual number of repositories, it is impossible to verify the completeness of the coverage.

3.3.1 Geographical

Figure 3 shows the distribution of repositories by continent, and we can see that all continents (and the Caribbean) are represented in the database. The presence of an "Other" option most likely indicates a cataloguing issue.

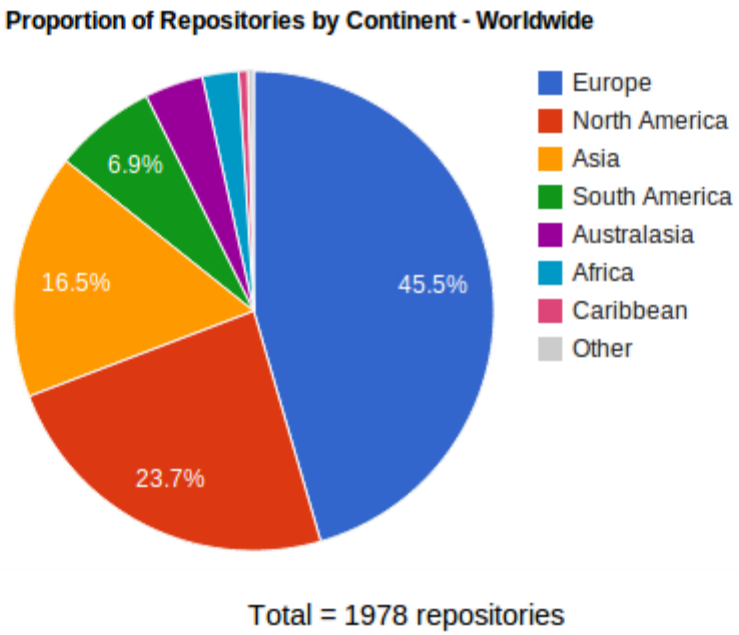

Figure 3: Proportion of Repositories by Continent - Worldwide

Figure 4 shows the distribution of repositories by country. The United States and the United Kingdom have significant proportions of the coverage, and it is unclear whether this is an artifact of the service being English-language oriented and originating in the United Kingdom, or whether this accurately reflects the proportions of repositories worldwide. It is also worth noting that 35% of the repositories registered are not in one of the large country categories, and that this accounts for a significant proportion of the total repositories. This may suggest a wide distribution of repositories

around the world; such a large proportion would suggest *against* a cataloguing issue.

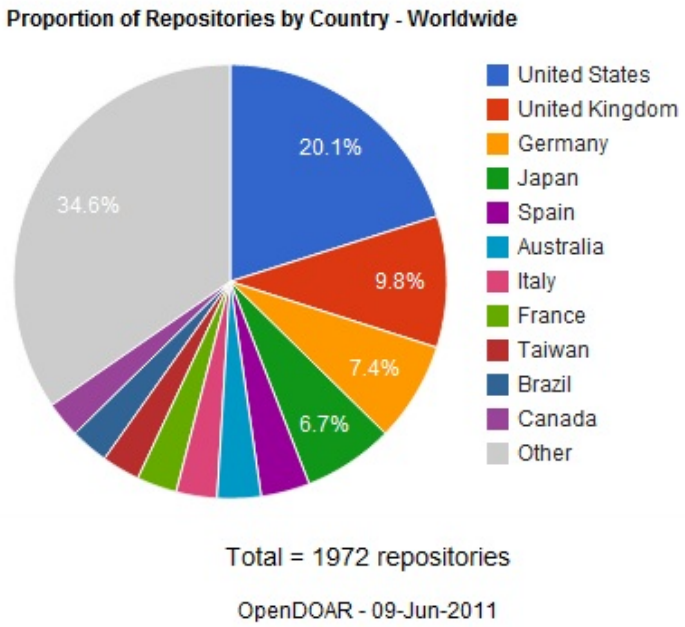

Figure 4: Proportion of Repositories by Country - Worldwide

Figure 5 is a screenshot of the Repository66 [8] interface showing the distribution of the repositories listed in both OpenDOAR and ROAR (it is therefore only indicative, as it does not differentiate between repositories from different registries). We can clearly see repositories represented in countries not listed in *Figure 4*, such as Argentina, Russia, China, Mexico, India, Malaysia and Suadi Arabia, which would suggest that they make up some of the 35% of "Other" countries in *Figure 4*.

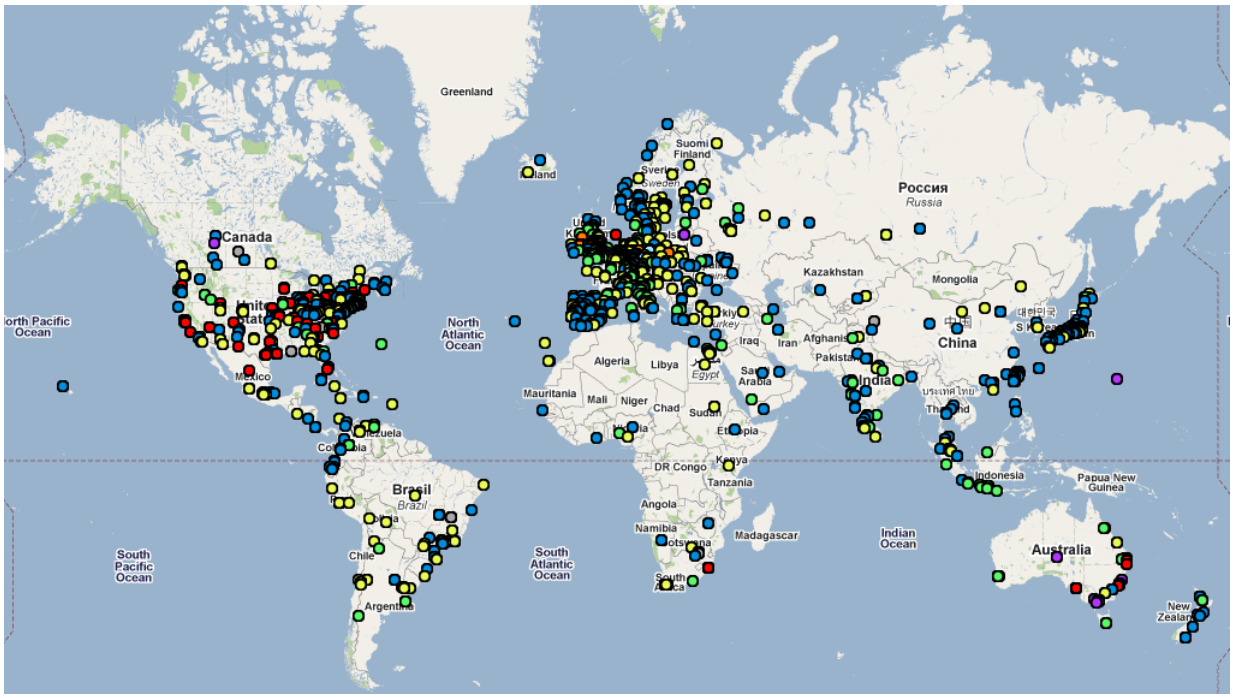

Figure 5: Repository 66 World Map - data from ROAR and OpenDOAR representing all listed repositories

3.3.2 Repository Type

Figure 6 shows the 4 categories that OpenDOAR places its entries into: Institutional, Disciplinary, Aggregating and Governmental. Superficial inspection would suggest that this distribution is likely to be generally true, with there being a large number of Institutional Repositories (one or more per institution); a smaller but still significant number of Subject Repositories (potentially one or more per broad subject area); a small number of aggregating repositories (limited by their value as single points of contact for large collections of other repository types); a small number of governmental repositories (one or more per country).

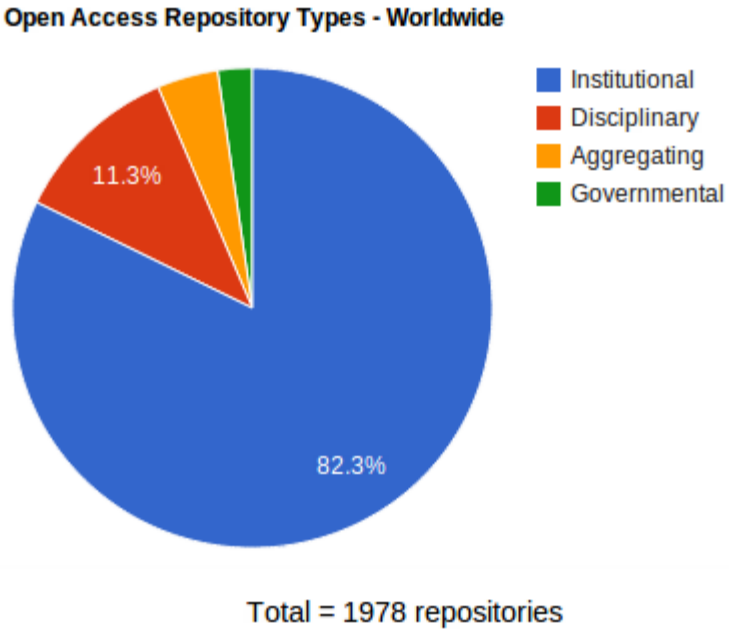

Figure 6: Open Access Repository Types

There may be some classification errors in the OpenDOAR data, particularly with regard to ownership of disciplinary repositories. For example, arXiv [13] appears in the user interface as belonging to Cornell University, which is technically true but difficult to locate from a user perspective.

3.3.3 Content Types

Figure 7 shows the content types that OpenDOAR recognises against the number of repositories which contain that content type. Superficial inspection suggests that this distribution is quite likely to be generally true. Journal articles and Theses and Dissertations have long been the focus of repository development so are no doubt the most common, while working papers, conference papers and book chapters follow naturally from them. The low number of bibliographic references (non-full-text records) is likely an artifact of the OpenDOAR collection policy which rejects bibliographic-only repositories [1].

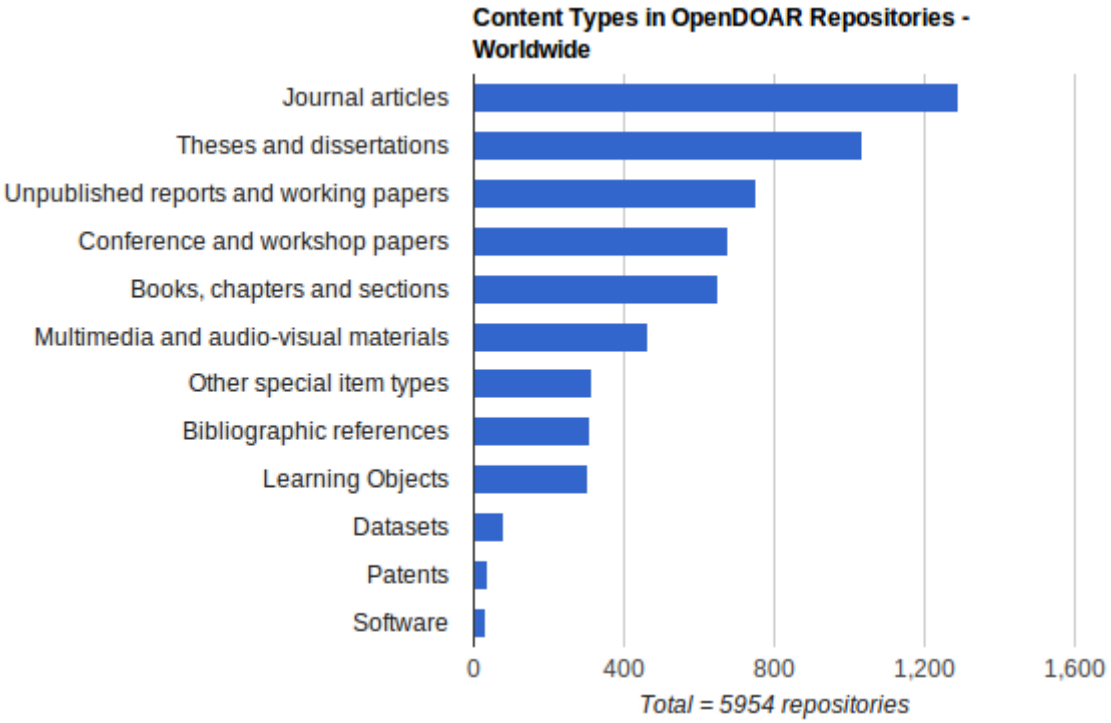

Figure 7: Content Types in OpenDOAR Repositories

3.3.4 Usage

OpenDOAR does not provide publicly accessible usage statistics, so it is not possible to do an accurate analysis of the kinds of usage coverage that it receives.

3.3.5 Software

Figure 8 shows the distribution of repository software in the registry. Referring back to *Figure 0* (and the caveats associated with it), we can create an adjusted chart (*Figure 9*) which shows the relative weights of the largest repository types. Comparing *Figure 9* with *Figure 0*, it is clear that the overall proportions of the repository software types is in line, suggesting a reasonable degree of general coverage. It is likely that the "Unknown" and the "Other" sections of *Figure 8* include the Fedora instances, but as Fedora tends to run as a back-end repository solution it is less commonly indexed as "Fedora" in the directory. For example, the list of repository types in OpenDOAR contains "Fedora", "Fez", and "VITAL" separately, although each of them is effectively Fedora.

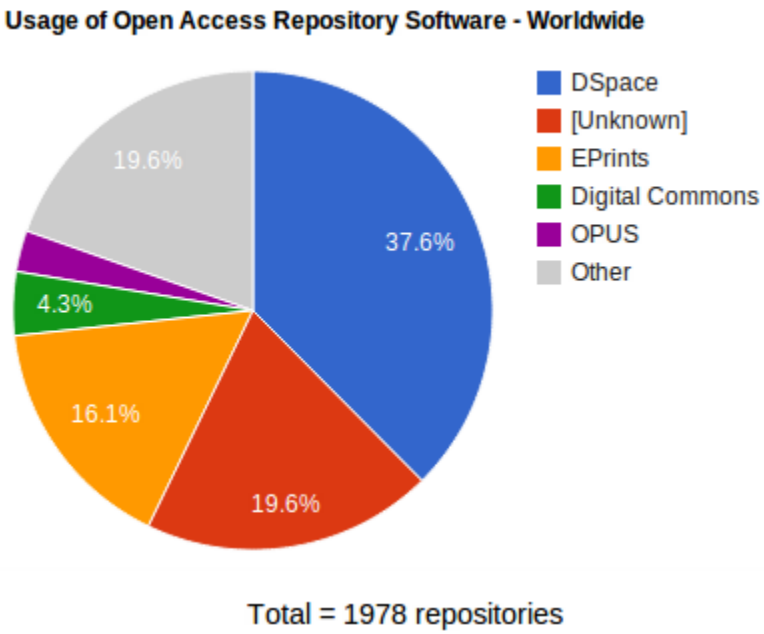

Figure 8: Usage of Open Access Repository Software

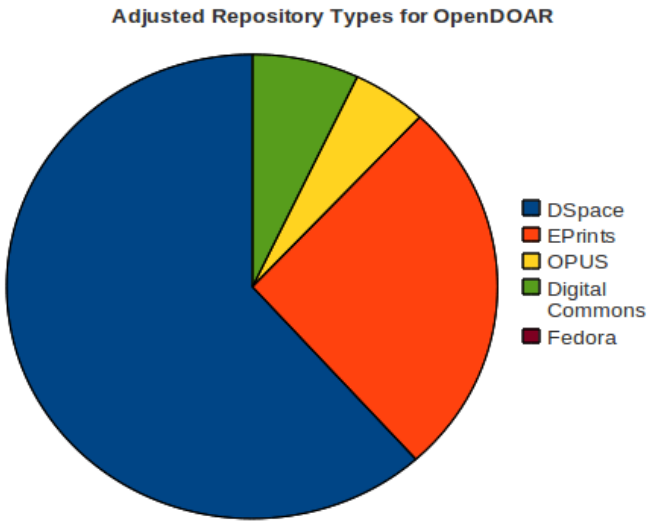

Figure 9: Adjusted Repository Types for OpenDOAR

Due to the large number of "Unknown" software types, OpenDOAR acknowledges the need for a campaign to identify them and enhance the metadata and related categorisations.

3.4 Functional Analysis

OpenDOAR has an end-user interface, an administrative interface (which is not publicly available) and an API.

3.4.1 End-User Interface

The following fields are available to the end-user as determined by a period of use of the system:

Address	Contacts	Content	Content Policy
Country	Data Policy	Date Added	Description
Established	Fax	Languages	Location
Metadata Policy	OAI-PMH	Organisation	Preservation Policy
Repository Name	Repository URL	Size	Software
Subjects	Submission Policy	Tel	Type

Table 1: Non-exhaustive list of fields available to end users through the OpenDOAR UI

There is no documentation on the site regarding what the total set of fields looks like or what the exact meaning of each field is, so the above should be considered indicative only.

The User Interface is mostly well designed, although there are a number of inconsistent behaviours observed, such as clicking on the Repository Name in the search results takes you to the home page of that repository, while doing the same thing in the browse results takes you to a full page in OpenDOAR about the repository. The UI also presents a large range of display formats for search results including tabular layouts, Google Maps, and charts.

The feature set for the end-user to interact with the directory contents is quite rich, and includes the following 3 general areas:

- **Search**
  The end-user can search the repository using free-text field, and/or by specifying constraints on: Subject Area, Content Type, Repository Type, Country, Language or Software.
- **Browse**
  The end-user can browse through the full directory listing which is broken down into a hierarchy of Continents, Countries, Organisations, then Repositories.
- **Statistics**
  The end-user can retrieve their Search results as a set of charts which can then be used to analyse the result-set. The charts available are Repositories by Continent, Repository Organisations by Continent, Repositories by Country, Repository Organisations by Country, Usage of Open Access Repository Software, Open Access Repository Types, Repository Operational Statuses, Most Frequent Content Types, Most Frequent Languages, OpenDOAR Subjects, Metadata Re-Use Policy Grades, Data Re-Use Policy Grades, Content Policy Grades, Submission Policy Grades, and Preservation Policy Grades [12].

End users may also suggest new repositories for the directory, or suggest updates to existing records. The user is presented with a form which requests the following information (pre-populated with the existing values if the user is suggesting an update):

Contact Email	Contact Name	Description	Location
OAI Base URL	Organisation*	Repository Name*	Repository URL*
Suggester Email*	Suggester Name		

*Table 2: Full list of fields required during suggestion. * indicates required field*

This information is then sent to the site administrators for manual creation or modification.

Further to interactions with the directory, OpenDOAR also provides a Google Custom Search over the repositories in the dataset [14], and a number of tools including an Email Distribution Service [15] and a Policies Tool [5] to help organisations produce better specified repositories for inclusion into the directory.

3.4.2 Administration

As OpenDOAR's collection policy is predicated on manual review and entry of the repositories into the directory, there are a number of administrative tasks which take place.

End-users can submit suggestions [16] to OpenDOAR for new repositories or updates to existing records, where they provide the details listed above in *Table 2*. By comparison with *Table 1* which lists the fields which are available to view from the user interface, it is clear that a lot of additional manual entry is required per record. OpenDOAR estimate that an average repository takes an administrator approximately 15 minutes to process (factors such as the native language of the repository, or the complexity of the policy data have an effect on this).

The process for create and update is essentially the same: The suggestion is added to the administrator's work queue via both email and addition to the OpenDOAR work queue. The administrator will manually inspect the repository for compliance with the collection policy, and complete a back-end web form with all of the system's fields available. During the creation/update process the administrators make ad hoc usage of tools such as Google Translate/Babelfish, Google Maps, and OAI-PMH to obtain the relevant data.

OAI-PMH is used in a number of ways:

- The *Identify* verb is used to acquire more metadata about the repository
- To acquire content counts. It should be noted that OpenDOAR regard this as unreliable due to variations in the implementation choices and quality of this interface.

The metadata held in OpenDOAR has a Size element for storing the number of items in the repository, which is partially populated as above with OAI-PMH but also by manual inspection. This figure, though, is only captured once when the repository is registered, so it cannot be used to examine repository growth over time. OpenDOAR are currently looking into extending their functionality to automatically collect content counts in the future (there is no way of extrapolating this data retrospectively).

There is no external mechanism for an end-user to request the removal of a repository from the directory. To date, nothing has been technically removed from the database. OpenDOAR runs a link-checker in the background to monitor the URLs held in the database. If feedback is received from the link checker or via the website, the repository may be flagged as "malfunctioning". If the repository is still malfunctioning a couple of weeks later they are "deleted" - nothing is deleted from the database, but the records are hidden from public view. During migrations from one repository to another, the original repository may be marked as "closed" and the record retained, while the new repository is added as a new entry.

3.5 Data Analysis

The OpenDOAR data curation model is to manually enter and manage all of the metadata in the dataset. As a consequence, data is only updated upon external request unless the link-checker flags an issue, although occasionally the OpenDOAR team carry out a campaign to improve certain aspects of the metadata or sets of records.

In general, the OpenDOAR dataset is very traditional; it could benefit with some modernisation to fit more closely with the way the web now works. In particular, it lacks good identifiers for the entities within the data: not just the repositories, but the organisations and units which it also models. Given URIs in the data where appropriate, OpenDOAR could present a much more coherent REST API, which was compatible with the goals of Linked Open Data [21].

3.5.1 What data can be held?

The data that can be held by the OpenDOAR system is the same (ignoring the administrative fields) as that presented in *Table 3*. This represents some clear descriptive metadata about the repository, its owning organisation, and some of its technical details.

classes	contacts	contentTypes	country	languages
oAcronym	oName	oNamePreferred	operationalStatus	oUrl
paFax	paLatitude	paLongitude	paPhone	policies
postalAddress	rAcronym	rDateHarvested	rDescription	repositoryType
rName	rNamePreferred	rNumOfItems	rOaiBaseUrl	rRemarks
rSoftWareName	rSoftWareVersion	rUrl	rYearEstablished	uAcronym
uName	uNamePreferred	uUrl		

Table 3: Full list of fields available via the API

It is lacking in the following areas:

- Repositories which are owned by more than one Organisation are difficult to represent (e.g. White Rose [22])
- It cannot hold multilingual data
- It does not hold retrospective data (a change history)
- It does not include ongoing representation of repository statistics, only a one-off content count

OpenDOAR are aware of these issues.

In addition, it is worth noting that the data format is not of any pre-existing standard - it has been created as necessary throughout development. There is no one metadata format which is appropriate to the needs. Meanwhile the subject classifications are not borrowed from any existing classification (such as Library of Congress) but have been developed by analysis of the data needing to be classified; as a further note, OpenDOAR report that subject classification of repositories is very difficult as they mostly tend to fit under the heading of "multidisciplinary".

3.5.2 How well populated are the fields?

Figure 10 shows how well populated the fields in the OpenDOAR schema actually are. This is based on data extracted via the API; the maximum value is 1972 instances of the field being populated, which represents the full size of the dataset at the time of analysis. Some of these fields will be very difficult to populate (such as the Fax number for the contact), and others may not be available for all of the repositories (such as the OAI URL). It is non-trivial to distinguish between these two factors for each field.

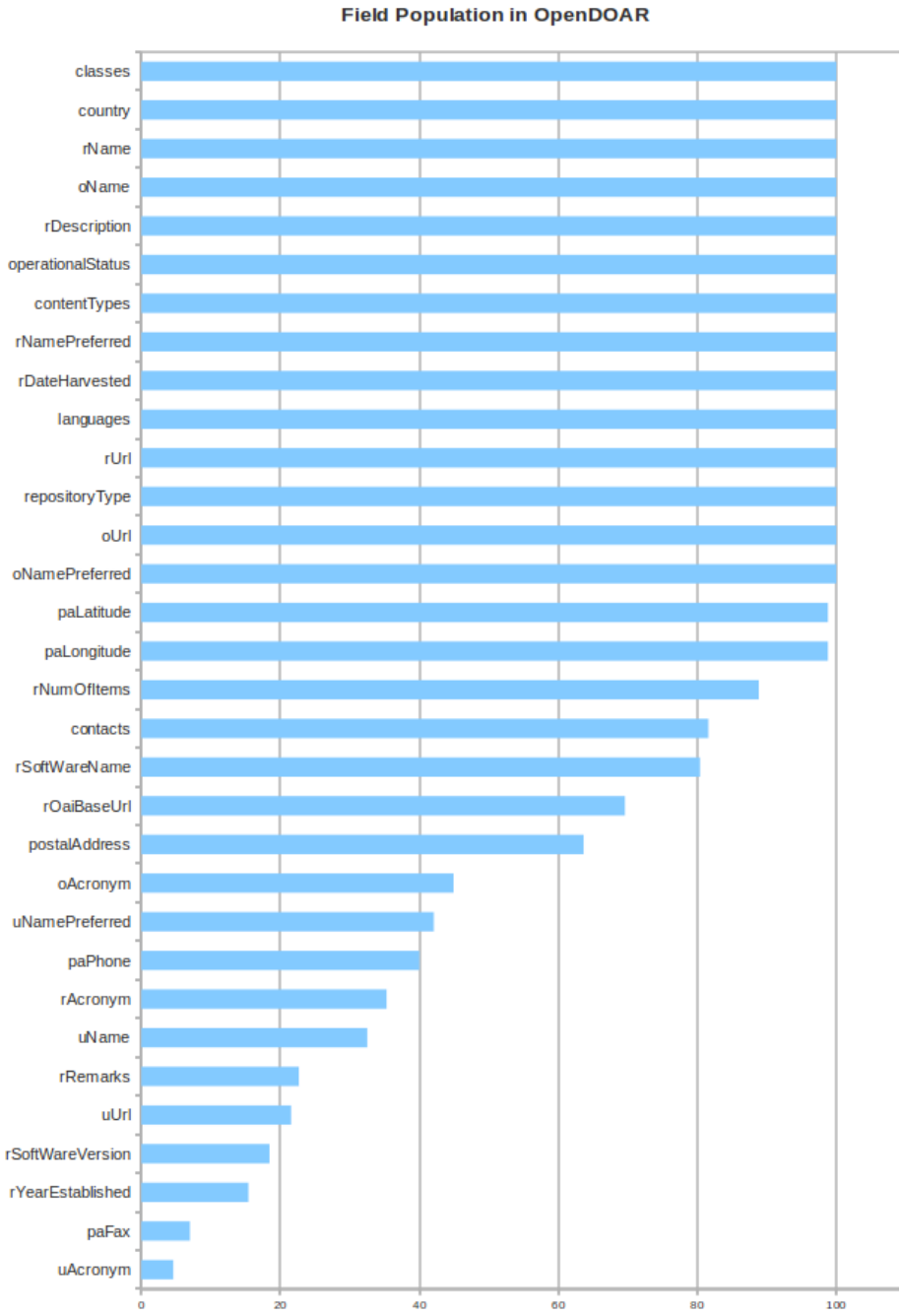

Figure 10: Field Population in OpenDOAR (% complete)

3.5.3 How accurately populated are the fields?

Inspection of the dataset converted to a regular spreadsheet indicates that the data is relatively accurately populated, with few fields filled in with content which should be in a different field. It is estimated that the incorrectly assigned values account for less than 1% of the total content, which is a very low error rate.

If there are any obvious systematic errors with the content it is that HTML entities appear in the data itself. Presumably this is a workaround to errors encountered in the API, but the correct approach would be to fix the API, not the data. An example of this is that the ampersand sign "&" can be found in the data itself represented as "&", which is its HTML entity name. Another commonly

encountered issue is extraneous punctuation in the Address field which probably arises from the code not cleanly handling incomplete/partial addresses.

3.5.4 How accurate is the data?

Without considerably more work it is difficult to tell how accurate all the data is. OpenDOAR do run link checking software against their dataset, though, so we can have a reasonable degree of confidence in the URLs provided, although independently verifying them would be of value. A number of the URLs in the data contain IP addresses, which is likely to be fragile, though. Anecdotal evidence from the API users indicate that there are a lot of spelling and formatting errors in the data.

3.6 API Analysis

In contrast to the User Interface, the API has a much fuller list of fields readily available to the API user. These are as in *Table 3*.

Unlike the user interface, there is some documentation available regarding the fields, in the form of the XML DTD [17], although this only defines the fields and their structure and does not provide any detail as to the field usages. Note that there are several fields which are held internally which are not available by either the UI or the API: these are limited to private notes fields for administrators and other administrative data.

Upon retrieving a record from the API, the developer will see XML much like this (some information omitted for brevity):

```
<OpenDOAR apiVersion="1.3">
   <copyright>Copyright 2011, University of Nottingham</copyright>
   <licence>OpenDOAR data is available for re-use under a Creative Commons Attribution-Non-
   Commercial-Share Alike licence</licence>
   <repositories>
      <repository rID="330">
         <rName>Academic Bibliography and Institutional Archive of Ghent University</rName>
         <rAcronym>Biblio at UGent</rAcronym>
         <rUrl>https://biblio.ugent.be/</rUrl>
         <uName>Universiteitsbibliotheek Gent</uName>
         <uUrl>http://lib.ugent.be/</uUrl>
         <oName>Universiteit Gent</oName>
         <oUrl>http://www.ugent.be/</oUrl>
         <country><cIsoCode>BE</cIsoCode><cCountry>Belgium</cCountry></country>
         <paLatitude>51.055600</paLatitude>
         <paLongitude>3.738600</paLongitude>
         <rDescription>This site is a university repository providing ....</rDescription>
         <rNumOfItems>120000</rNumOfItems>
         <rDateHarvested>2009-10-29</rDateHarvested>
         <repositoryType>Institutional</repositoryType>
         <operationalStatus>Operational</operationalStatus>
         <classes><class><clCode>C</clCode><clTitle>Multidisciplinary</clTitle></class></
         classes>
         <languages>
            <language><lIsoCode>en</lIsoCode><lName>English</lName></language>
            <language><lIsoCode>fr</lIsoCode><lName>French</lName></language>
            <language><lIsoCode>nl</lIsoCode><lName>Dutch</lName></language>
         </languages>
         <contentTypes>
            <contentType ctID="1">Journal articles</contentType>
            <contentType ctID="6">Theses and dissertations</contentType>
         </contentTypes>
         <policies>
            <policy>
```

```
            <policyType potID="3">Data</policyType>
            <policyGrade pogID="15">Full data item policies explicity undefined</
            policyGrade>
            <poStandard>
               <item>Anyone may access full items free of charge.</item>
               <item>No full-item re-use policy defined. Assume no rights at all have been
               granted.</item>
            </poStandard>
         </policy>
         <policy>
            <policyType potID="5">Preserve</policyType>
            <policyGrade pogID="31">Preservation policies defined</policyGrade>
            <poStandard>
               <item>Withdrawn items are deleted entirely from the database.</item>
               <item>Withdrawn items' identifiers/URLs are retained indefinitely.</item>
            </poStandard>
         </policy>
      </policies>
      <contacts>
         <person>
            <pJobTitle>Site Administrator</pJobTitle>
            <pEmail>****@ugent.be</pEmail>
         </person>
      </contacts>
   </repository>
  </repositories>
</OpenDOAR>
```

The API for accessing this data is sophisticated and quite well documented [18]. It is possible to pass in URL query parameters to filter by country, language, subject, repository type, content type, policy grade codes, modified dates, OAI URL availability, and to pass in search keywords. Further to this, the developer can specify how much detail the result set should give back, and how the data should be sorted. The limitations of the documentation are around defining exactly what the fields in the returned data mean, although in many cases they are sufficiently well named that this is not an issue.

The API itself is effectively RESTful insomuch as it is HTTP GET requests with URL query parameters which return data in XML. Nonetheless, there are no URLs for the individual records - they have to be located through the search API with the appropriate keyword parameters. Additionally, calling the root of the API [19] returns an empty result set, where it might be more appropriate to return some resources for accessing the data. Unlike a standard Atom style feed interface, there is also no paging provided over the result set. The returned XML is of a custom format (albeit well defined with a DTD [17]), and does not use any existing vocabularies for describing the data. The reason for this latter point is that at the time of development (2005/2006) there was no clear metadata set which would be used to describe a repository, and that likely remains true today.

There is no client software library available which can consume the API (although a very basic one was developed during the creation of this report [20]), but due to the complexity of the API - and the extensive lists of codes for country, language, subject and so forth - there would be a good argument for making one available.

Overall the API is well designed, but it is also very complex and it is not clear whether the level of functionality that it provides is of genuine use. It would also benefit significantly from improved documentation of the data.

3.6.1 API Consumer Feedback

There are 4 formally known users of the API:

- Repository66 [8]
- Institutional Repository Search [10]
- Open Access Repository Junction [9]
- BASE search [11]

This section summarises feedback from developers with Repository66 and Open Access Repository Junction on how they interacted with the API; see the Method section for introductions to those systems.

As OpenDOAR does not have any client libraries, and there are no plans to produce client libraries, the developers were required to work from scratch.

3.6.1.1 Documentation

Both developers found it very quick to get up and running with the API, and found the documentation useful and well written.

3.6.1.2 Data

It was felt that the human involvement in the data in some cases caused issues. In particular there are many typos in the text fields, and these tended not to get corrected because updates to the data happen only when actioned by an administrator. The presence of HTML entities in the source data was also confusing for the machine-to-machine interfaces.

Furthermore the data tended to be directed more towards human users than machine users; better data structuring and the use of boolean fields for identifying properties could be valuable. It is also not possible to represent multiple organisations per repository which causes issues for repositories such as White Rose [22].

Multilingual data was also a problem. In cases where the name of the repository was not in English, it had been translated, but this was not the expected behaviour; the expectation was that the name of the repository be in its native language, with alternative translations available in appropriate ways.

Better handling of repositories which have ceased trading would also have been useful: to be able to know when a repository ceased and whether it was replaced by another.

Some use cases require access to data about not only open access repositories but also closed access ones, and there may be some value to consumers of the API in being able to provide that data. This means that the collection policy for OpenDOAR in some ways works against the requirements of parts of the community.

3.6.1.3 API

Both developers ignored the complex query parts of the API and downloaded a complete dump of the dataset to then work on locally. Both were complimentary about the design of the API, despite having not actually needed to use it. The Open Access Repository Junction API is partly inspired by OpenDOAR.

Taking data dumps regularly, it was noted that the ability to take a feed of latest additions to the database was extremely useful as this saved having to download the entire dataset each time updates were wanted.

3.6.1.4 Data Schema

Neither developer felt that a standard set of metadata elements would have been of significant value. The schema used by OpenDOAR is sufficiently self-explanatory that the important fields were easy to identify. If the API were to be extremely heavily used, though, by a variety of consumers it was suggested that a standards-based approach might be worthwhile.

The format of the data is XML, but the developers would have been happy with any consistent, structured format such as JSON.

3.6.1.5 Support

Both developers contacted OpenDOAR during their work for informal support, and were given that support outside of the normal roles of the individuals concerned. There is no ticketing system or support address.

During some of the work, improvements were made to the OpenDOAR data which could have been useful to feedback, but there is no formal route by which this can be done. For example, Open Access Repository Junction identified equivalences with ROAR which could have been useful to either service.

3.7 Future/Technical Roadmap

The technical roadmap for OpenDOAR is not a well defined list of objectives, but a rather loose wishlist. It includes:

- Automated repository record counts for constructing repository growth charts, etc.
- Support for registering API endpoints other than OAI-PMH. This would include but not be limited to SWORD and OAI-ORE.
- Multilingual data
- Internationalised User Interface
- More work on capturing subject proportions of content
- Sort repository list by date of registration and/or date of update
- Better comparisons/links with ROAR and any other repository lists (such as the software specific lists maintained by their communities)

4 ROAR

URL: http://roar.eprints.org
Software: EPrints 3, incorporating SNEEP extensions from ULCC.
Programming language: Perl
Technology stack: Linux, Apache, MySQL

4.1 Introduction

The (Registry of Open Access Repository) ROAR has been developed with an aim "to promote the development of open access by providing timely information about the growth and status of repositories throughout the world." It is part of the wider EPrints initiative and is hosted by the University of Southampton as part of the EPrints.org network [1].

Being integrated with the development of EPrints, ROAR is able to take advantage of the EPrints ecosystem. The front end is a relatively lightweight implementation of EPrints underpinned by a OAI-PMH data set of approximately 50-60Gb (which is key to ROAR's feature set) indexing individual DC records from all archives. Despite close links with EPrints the service is still very much focussed on tracking open access literature across all different software, institutional and geographic boundaries.

There is also tangential support via ROARMAP (Registry of Open Access Repository Material Archiving Policies) [24] which allows institutes to register their Open Access Mandate. ROAR also provides a range of support and tools for users to interface with the data and analyze various statistics via a graphical analysis.

4.2 Growth

ROAR has seen a sustained level of growth since 2004 growing from less than 500 records to well over 2000 currently. As *Figure 11* shows ROAR is growing at a rate of roughly 300 new repositories per year and currently covers approximately 2250.

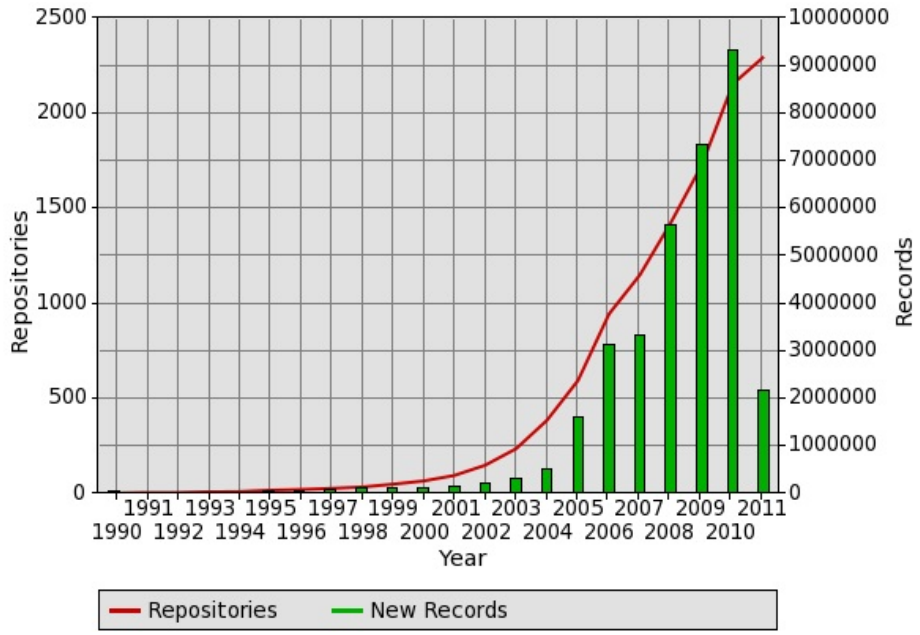

Figure 11 *Growth of the ROAR Database - Worldwide*

In line with the analysis of OpenDOAR we consider how this growth compares to maximum throughput and rate of growth of the space in general. In comparison with the DOAJ for example, growth is subdued (DOAJ is growing at a rate of approximately 2500 journals per year, with a dataset of 8617 journals) however focussing on bibliographic registrations the situation looks to be inline with the rate of growth in the space.

The trend for ROAR very much reflects the outlook that in mature countries registrations are reaching saturation point. For the UK, ROAR has just under 200 entries indicating that most bibliographic registrations are already complete and growth is therefore driven by other assets, such as data sets. *Figure 12* shows the rate of growth of ROAR for different source countries over time.

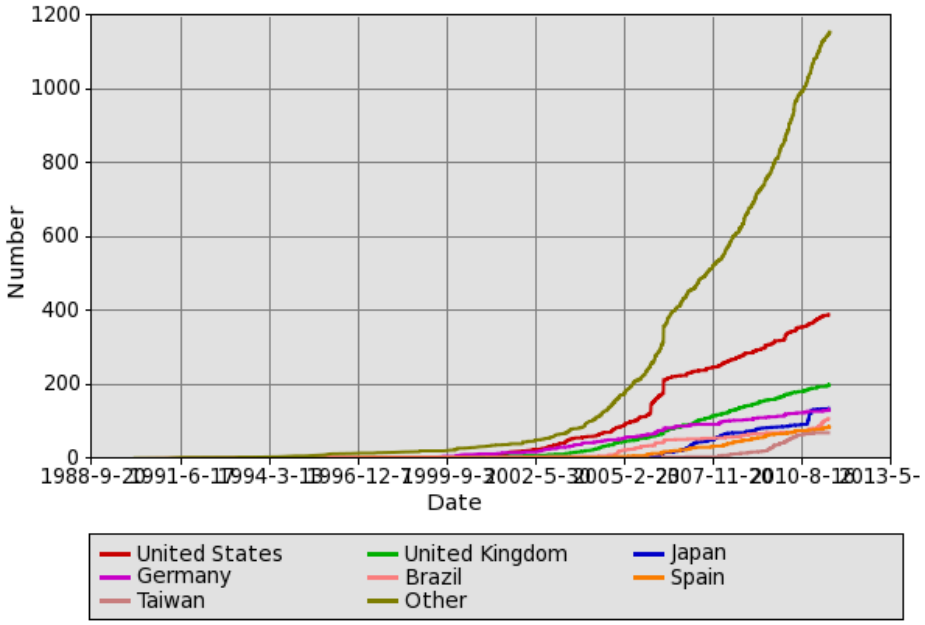

Figure 12 Growth of the ROAR Database - by country

In terms of usage statistics there is limited information available. There are a number of services

which are known to use the ROAR API, though:

- Repository66 [8]
- Open Access Repository Junction [9]
- Institutional Repository Search [10]
- BASE - Bielefeld Academic Search [11]

This would indicate at least some uptake.

4.3 Coverage

ROAR is global in scope cataloguing repositories from every continent. It is open to all new registrations contingent on the repository meeting its relatively open collection policy[2]. In this section we look at various breakdowns of the coverage areas as provided by ROAR via its graphical analysis section [25].

The main aim of this section is to determine if there is any significant bias in coverage that may skew the results. To do this we consider geographical spread, repository type, content type and usage. We demonstrate that coverage is not *heavily* skewed towards any country, software, repository type, etc; the statistical distributions of the coverage appear to be approximately in-line with what you would expect globally. What it does *not* show is whether the coverage is complete. See the section **Growth** for details about the size of the dataset; without further authoritative data about the actual number of repositories, it is impossible to verify the completeness of the coverage.

4.3.1 Geographical

Of the key countries broken out by ROAR it was clear that the register was slightly more European oriented with European based organizations accounting for 28.6% of the leading countries as opposed to North America which accounted for 20.09%. The data is shown in *Figure 13*.

The country breakdown indicates a large ‘other’ category accounting for 29.71% of repositories. It is not clear from the data available if this is due to a cataloguing issue or due to the presence of significant geographical spread.

[2]ROAR does not have a formal collection policy, but its objectives are to aid in providing a catalogue of all OA publishing. See **Appendix B** for some details.

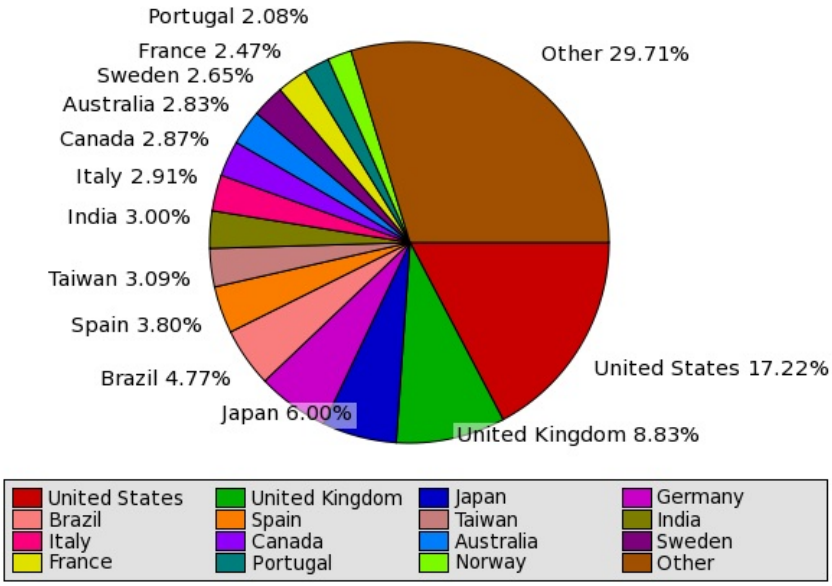

Figure 13: Proportion of Repositories by Country

An analysis of total known repositories worldwide suggests that at least some of those listed as 'Other' are indeed from smaller unlisted countries that do not have significant numbers of repositories. Repository66 - a graphical representation of repositories throughout the world - is based on data from ROAR and OpenDOAR (see *Figure 14*), and we can clearly see there are many repositories which likely make up the 'Other' category.

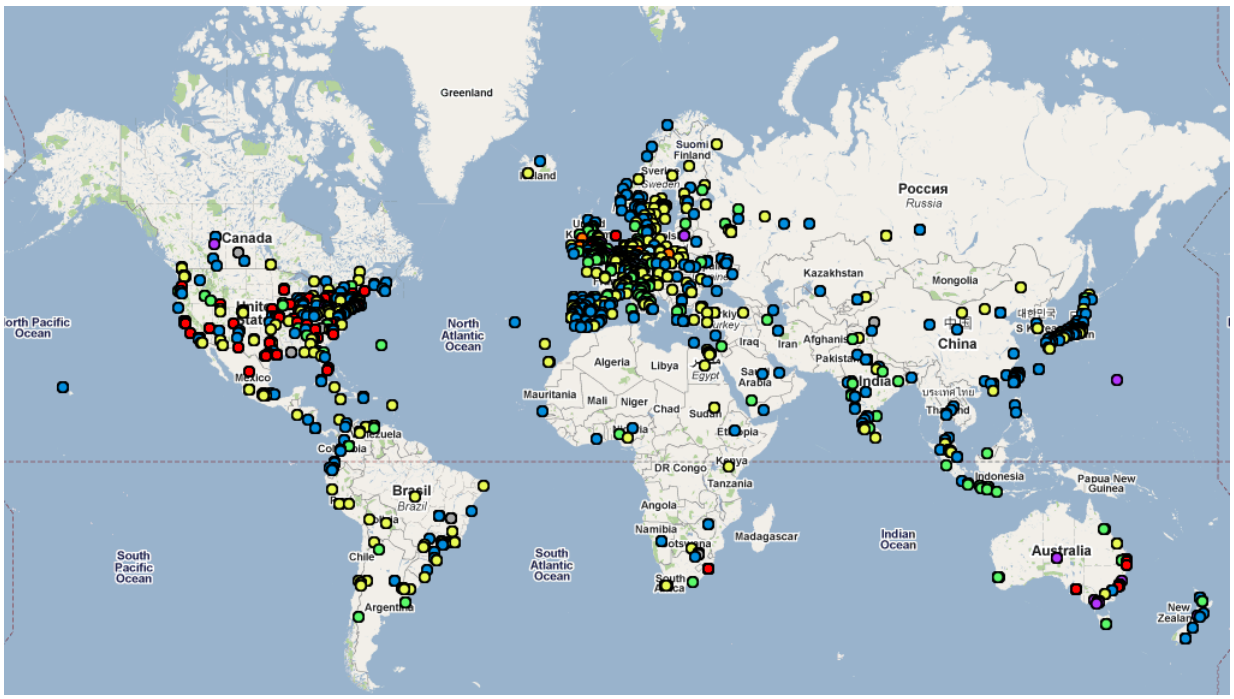

Figure 14 (reproduced from Figure 5): Repository 66 World Map - data from ROAR and OpenDOAR representing all listed repositories

4.3.2 Repository Type/Content Type

ROAR lists repositories according to its own categorization, concentrating mainly on the content held in the repository or supposed intent behind the repository mandate. There is some scope for confusion with this schema as there is overlap between organizational type and content type.

As can be seen from *Figure 15*, 'Research Institutional or Departmental' repositories dominate with 62.22%, reflecting the dominance of this strand within the space of repositories. The third largest segment however is 'e-Theses' which makes no mention of institutional type. While this is a useful category in indicating content type held in the repository it is not relevant in comparison to organizational structure.

While this categorization has more segmentation than other OARRs the relevance depends on the robustness of the categorization. It is not clear how exclusive the categories are with the potential that for some repositories several categories may be applicable.

The clipped text at the upper left of *Figure 15* is an artifact of the ROAR UI.

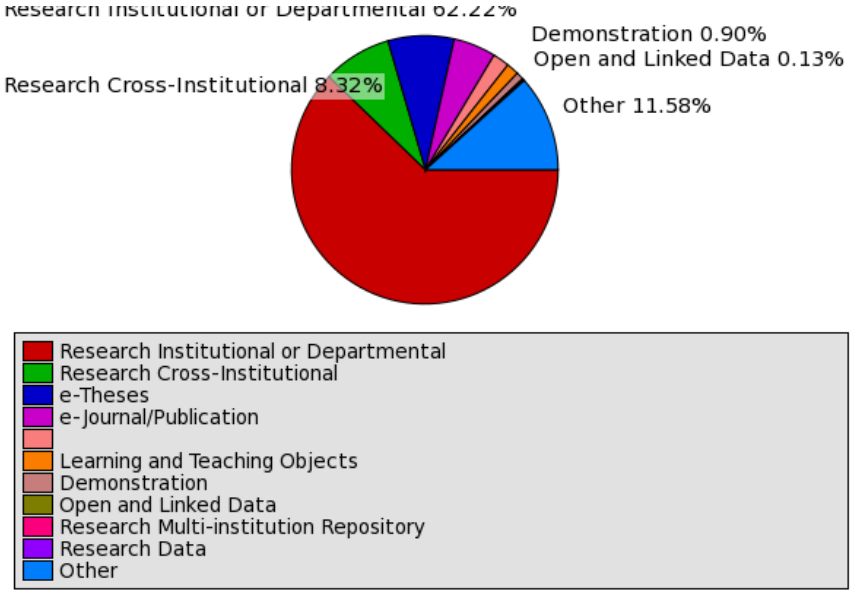

Figure 15: Open Access Repository Types

4.3.3 Usage

ROAR does not provide publicly accessible usage statistics, so it is not possible to do an accurate analysis of the kinds of usage coverage that it receives.

The development team noted that while there is currently no analysis of usage statistics, the data, stored in archived weblogs, could be pulled together should the need arise to seek funding.

4.3.4 Software

Figure 16 shows the distribution of repository software for ROAR. As with OpenDOAR we can use this to create a representation of adjusted repository software types as shown in *Figure 17*. This

indicates an excellent match between software types represented in the register and the general landscape; *Figure 17* shows the proportions of these well-known and widely used repository platforms, and indicates a proportional spread which is close to what we would expect for an unskewed register.

The existence of a fairly substantial 'Other softwares' category raises some concerns over the robustness of meta data handling. At present this segment accounts for nearly a quarter of repositories indexed by ROAR. While many of these repositories may be based on more obscure software a significant number are likely to be due to misrecognition.

Additionally it is difficult to tell whether the total repositories on the individual software pages are real, so what the total coverage is is hard to estimate. To detail this accurately would be quite time consuming amounting to a project in itself.

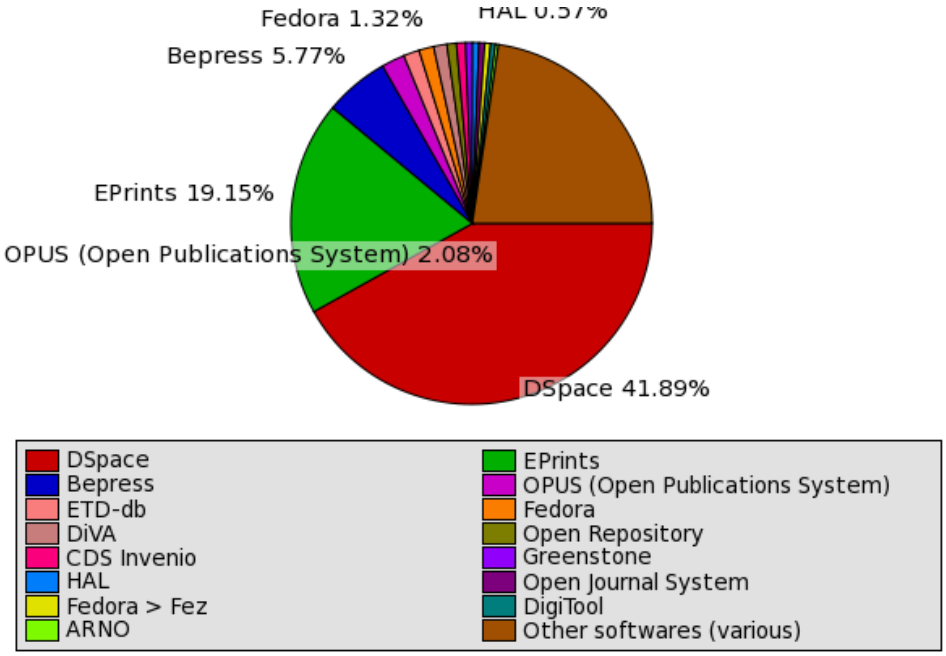

Figure 16: Usage of Open Access Repository Software

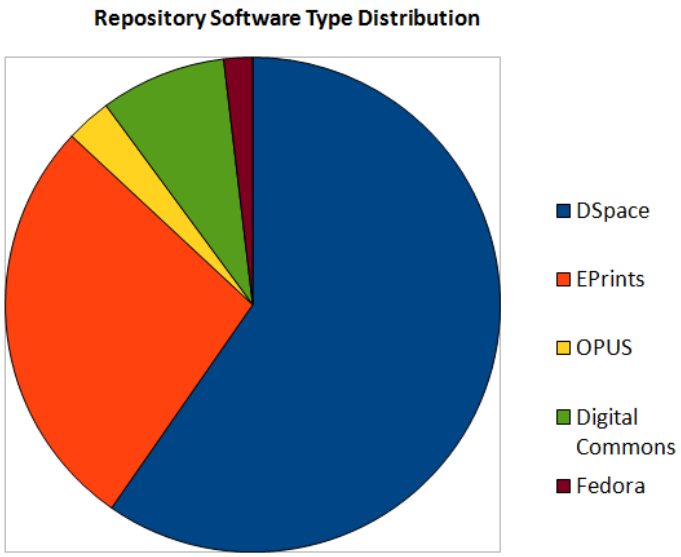

Figure 17: Adjusted Repository Types for ROAR

4.4 Functional Analysis

ROAR has an end-user interface, an administrative interface (which is not publicly available) and an API.

4.4.1 End-User Interface

The web interface provides a range of meta data information to end users about the various repositories indexed. *Table 4* provides a non-exhaustive list of fields available via the web interface:

Repository URL	Country	ROAR ID	Record Count	History
Organisation	Description	Birth Date	Digital Objects	Software
Address	Repository type (<domain (e.g. institutional)>)	Public Fulltext Formats	<objects> <documents>	<records with documents>
Daily Deposit Activity	OAI-PMH Interface	Location (<lat> & <long>)	<records with object>	

Table 4: Non-exhaustive list of fields available to end users through the ROAR UI

There is no documentation on the site regarding what the total set of fields looks like or what the exact meaning of each field is, so the above should be considered indicative only.

The User Interface offers relatively comprehensive search and query capabilities mostly well internationalised. Users can filter results by a wide range of options as shown in *Table 5*

ROAR ID	*Home Page*	*OAI-PMH Interface*

Record Count	*Registry Title*	*Software Subjects*
Description	*Repository Type*	*Birth Date Country*

Table 5: Example search filters available to end users through the ROAR UI

The UI also offers the facility to download the result set in a number of formats including Google Custom Search Engine XML, JSON, and RDF.

The feature set for the end-user to interact with the directory contents is quite rich, and includes the following 3 general areas:

- **Search**
  The end-user can search [26] the repository using free-text field, and/or by specifying constraints on: ROAR ID, Home Page URL, OAI-PMH Interface URL, Present in which other registry, Title, Description, Repository Type, Birth Date (although it is unclear what field this refers to), Country, Software, Subjects, Record Count, Exemplar (although it is unclear what field this refers to)

- **Browse**
  The end-user can browse through the full directory listing which is broken down to allow users to browse by: Country, Type, and Software

- **Statistics**
  A limited amount statistical data is available via the web interface [27]. A graphical analysis section on the website allows users to access statistics and ready built graphics from the following dimensions: Known repositories over time, records per year, Repository type, Country Software

4.4.2 Administration

Registered users of the service may also suggest new repositories for the directory. The following information is required:

Homepage URL	OAI-PMH	Repository Type	Title
Description	SWORD deposit endpoint (submission not service document)	RSS	Twitter
More than 75% full-text (boolean)	More than 75% OA full-text (boolean)	OA mandate (boolean)	Organisation
Location	Software	Subjects	Contact
Other registries	Additional info	Comments and suggestions	

Table 6: fields for creating a record in ROAR

Once a user has added a new repository it first goes to editorial review before reaching the

registry. New records go into the buffer, and are manually reviewed. The editorial review cycle is approximately weekly but this can be variable depending on workloads and number of submissions. Most rejections are spam and e-journals.

Updates are effected under the following circumstances:

1. People own their own records so can update whenever they choose (ROAR wants to move towards this as the standard model)
2. By email to the ROAR developer (which occurs at a rate or around one or two emails per week)

An OAI-PMH harvester is run every 1 to 2 weeks with a secondary process examining the DC in the OAI-PMH feed to locate any attached fulltext. ROAR does *not* look at the EPrints CGI counter, which does contain this information for that platform only.

Repositories which cease to exist are removed only if and when someone notices; it is difficult to tell what is permanently dead and what is just not working at the moment, so this information is not acted upon until there is some certainty.

4.5 Data Analysis

The ROAR data curation policy is to encourage the repository owers to maintain their own records within the system (as a consequence updates to records are variable in their regularity). Nonetheless, data held in ROAR is approved via a manual review process which offers some small amount of validity checking; in addition it prevents spam and incorrectly registered content such as e-journals (which are redirected to DOAJ). They do not, as a general rule, run technology such as link-checkers on the data.

4.5.1 What data can be held

The data that can be held by ROAR are as indicated in *Table 7*, and represents a fairly extensive set of metadata about the repository, its contents, APIs and contact methods.

rev_number	datestamp	lastmod	status_changed
type	home_page	title	software
geoname	version	date	recordcount
recordhistory	fulltext	open_access	mandate
description	rss_feed	exemplar	note
contact_email	sword_endpoint	suggestions	twitter_feed
succeeds	oai_pmh	submit_to	subjects
activity	fulltexts	webometrics	location

registry	organisation	item_issues	

Table 7: Fields available via the API

It is lacking in the following areas:

- It cannot hold multilingual data
- It's record history is a simple list of numbers, and there is little information on what these numbers mean
- Repositories which are owned by more than one Organisation are difficult to represent (e.g. White Rose [22])
- It lists SWORD deposit endpoints but not SWORD service documents, which is limiting.

In addition, it is worth noting that the data format is not of any pre-existing standard - it has been developed in response to developing requirements placed upon the system. The ROAR team note that since its inception there has been no need to perform any significant metadata migration during schema changes, so it has proved very stable; most metadata enhancements have been additive such as the addition of support for SWORD endpoints.

4.5.2 How well populated are the fields?

Figure 18 shows how well populated the fields in the ROAR schema actually are. This is based on data extracted via the API; the maximum value is 2321 instances of the field being populated corresponding to the full number of records in the dataset at the time of download.

While a good number of fields have high population there is a significant lack of completion for some important fields. Some of these fields may not be available for all repositories which may account for some instances of less than full population. In particular, potentially useful fields such as the SWORD endpoint or the Twitter feed are extremely sparse. In the case of Twitter this is likely to be because not that many repositories make announcements via Twitter, but in the case of SWORD, which is a standard part of DSpace, EPrints and Fedora, this is a critical omission. It is likely that omissions like this are due to ongoing schema changes introducing such fields *after* many of the repositories have been registered (there are no SWORD endpoints at all in the oldest 1600 records, for example, and it is clear from the data where support for this field was introduced).

At present ROAR do not have a strategy for encouraging users to come back and update their data after schema changes which might introduce more fields.

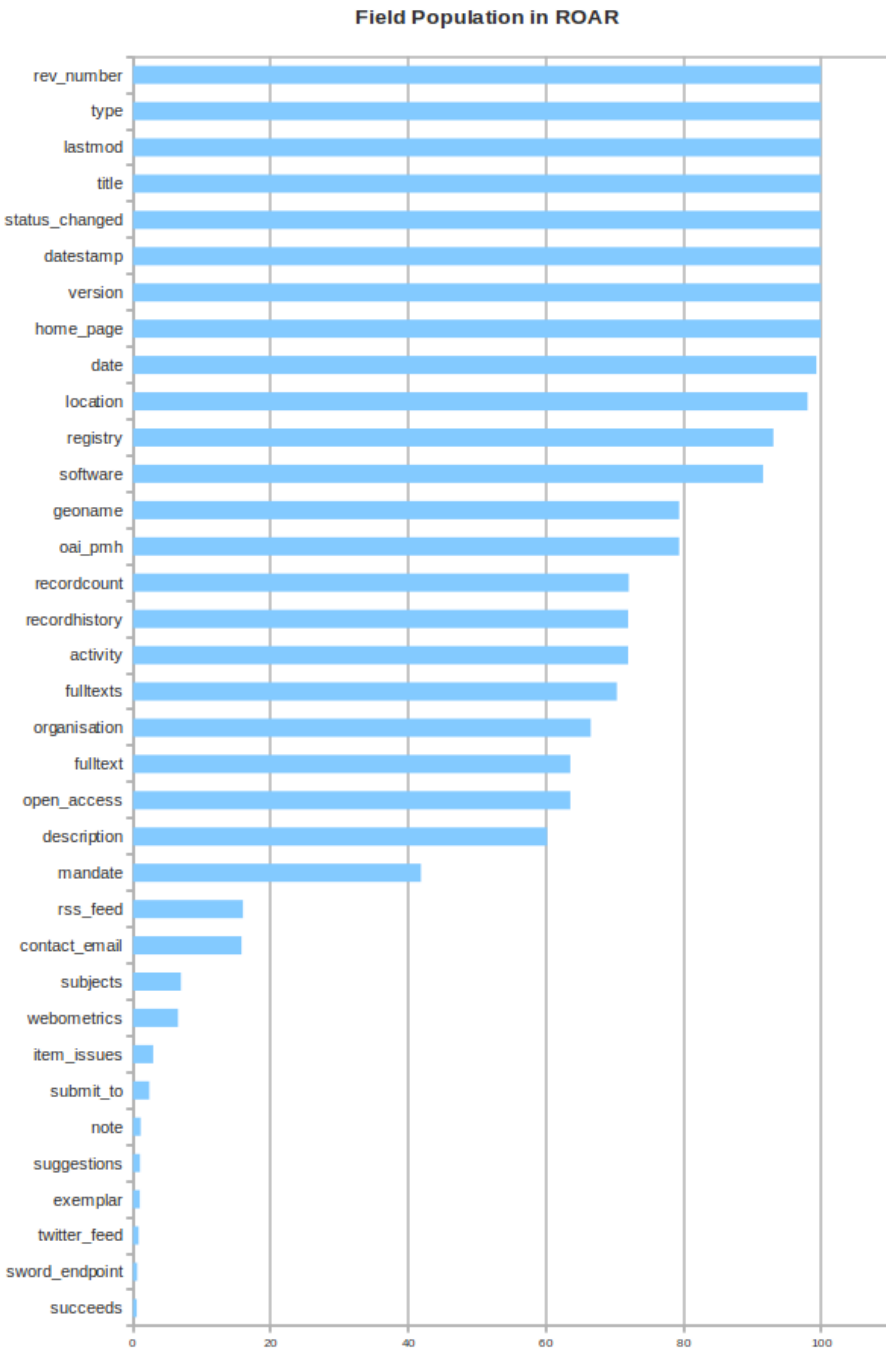

Figure18: Field Population in ROAR (% complete)

4.5.3 How accurately populated are the fields?

Inspection of the dataset converted to a regular spreadsheet indicates that the data is relatively accurately populated, with few fields filled in with content which should be in a different field. It is estimated that the incorrectly assigned values account for less than 1% of the total content, which is a very low error rate.

There is little obviously wrong with the data format itself, although there is a reasonably high incidence of unnecessary or erroneous punctuation in certain fields (hanging commas, empty parentheses, etc) such as *title* and *organisation*.

4.5.4 How accurate is the data?

Without considerably more work it is difficult to tell how accurate all the data is. ROAR do not run link-checking software on the server, so there is a high degree of uncertainty in the quality of the URLs provided. Independently verifying these would be of value. A number of the URLs in the data are observed to contain IP addresses, which is likely to be fragile.

4.6 API Analysis

The API in ROAR is divided into two parts: an OAI-PMH feed and a full EPrints XML export of the dataset. The OAI-PMH interface is an implementation of the standard EPrints OAI-PMH, and presents the data as described in *Table 8*, which is a non-exhaustive list of fields which appear in the output feed; no documentation is available as to what the full list of fields are, or how they map to the core ROAR data model, although in most cases this is self-evident.

dc:title	dc:date	dc:type	dc:relation
dc:coverage	dc:description	dc:publisher	

Table 9: Non-exhaustive lists of DC fields available in ROAR's OAI-PMH feed

The fields available in the full EPrints XML export are listed in *Table 7*. There is limited documentation regarding the fields; there is an XML schema document [28] which formally lays out the structure of the data in the API, but it is very long and complex and lacks clear user guides to the meaning of fields. Evidence from the API Consumer Feedback indicates that this document was largely useless in actually understanding the API and the data.

Upon retrieving a record from the EPrints XML export, the developer will see XML much like this (some information omitted for brevity):

```
<eprint id="http://roar.eprints.org/id/eprint/1">
    <eprintid>1</eprintid>
    <rev_number>630</rev_number>
    <documents>...</documents>
    <eprint_status>archive</eprint_status>
    <userid>1</userid>
    <datestamp>2010-01-06 13:43:48</datestamp>
    <lastmod>2011-02-03 20:19:10</lastmod>
    <status_changed>2010-01-06 13:43:48</status_changed>
    <type>subject</type>
    <metadata_visibility>show</metadata_visibility>
    <item_issues_count>0</item_issues_count>
    <home_page>http://archivesic.ccsd.cnrs.fr/</home_page>
    <title>@RCHIVESIC</title>
    <oai_pmh>
        <item>http://archivesic.ccsd.cnrs.fr/oai/oai.php</item>
    </oai_pmh>
    <location>
        <item>
            <country>fr</country>
```

```
            <city></city>
            <latitude></latitude>
            <longitude></longitude>
        </item>
    </location>
    <software>hal</software>
    <geoname>geoname_2_FR</geoname>
    <version>other</version>
    <date>2002-05-17 19:24:41</date>
    <activity>
        <low>63</low>
        <medium>0</medium>
        <high>0</high>
    </activity>
    <recordcount>1272</recordcount>
    <recordhistory>
        0,0,0,0,0,0,0,0,0,0,0,0,75,285,482,606,756,895,991,1105,1240,1272
    </recordhistory>
    <fulltexts>
        <total>0</total>
        <docs>0</docs>
        <rtotal>0</rtotal>
        <rdocs>0</rdocs>
    </fulltexts>
    <registry>
        <item>
            <name>opendoar</name>
            <id>58</id>
        </item>
        <item>
            <name>celestial</name>
            <id>669</id>
        </item>
    </registry>
</eprint>
```

The API for accessing the data is very straight forward; it uses either a standard, well understood, OAI-PMH interface or simply provides a link to download the entire dataset. The limitations of the API are really around the overall limitations of the OAI-PMH interface: this exposes only simple DC metadata which does not contain the most interesting parts of the ROAR data. As such it is largely useless, and the developers of client systems simply download the entire dataset from the provided link. Since the full data export is just that, there is no real full API to speak of for ROAR.

The data itself is of a custom XML format (albeit well defined with an XML schema, but which is too long and complex to be of value), and there is no documentation as to the field usage (although many are self-explanatory, some are not). The subject classifications used by EPrints are Library of Congress, but the other classifications for repositories are ad hoc and ROAR specific. The same is true of the metadata schema, which was developed based on requirements as they arose during development.

There is no client software library available which can consume the API, although any OAI-PMH code could be used against that interface were that to be of value. It is unlikely that a client library is necessary for ROAR in its current form, as there is no queryable data API which would need supporting. Some client code was developed during the creation of this report [20].

Overall the data is extensive and the EPrints XML export is easy enough to work with. The whole system would benefit significantly from improved documentation of the data, though.

4.6.1 API Consumer Feedback

There are 4 formally known users of the API:

- Repository66 [8]
- Institutional Repository Search [10]
- Open Access Repository Junction [9]
- BASE search [11]

This section summarises feedback from developers with Repository66 and Open Access Repository Junction on how they interacted with the API; see the Method section for introductions to those systems.

ROAR does not have any client libraries, and there are currently no plans to produce client libraries. As a result the developers were required to work from scratch however, existing OAI-PMH clients can be used for limited effect.

4.6.1.1 Documentation

Both developers found it very quick to get up and running with the API, although ROAR was poorly documented.

4.6.1.2 Data

Multilingual data was an issue: the ideal set up is to have names and other data in the native language of the repository, with translations as alternatives where appropriate, but this configuration is not supported.

An API user from OA-RJ commented: " The repo name should be the repo name in its native language. Translations should be better handled as this is an international world. A Korean repository record is most likely useful for a Korean researcher."

Better handling of repositories which have ceased trading would also have been useful: to be able to know when a repository ceased and whether it was replaced by another.

It was felt that overall the quality of the data was quite low, possibly as a result of a lack of curation beyond that by the repository managers submitting the data. Repository 66 were given access to the administration side of the system so that quality could be enhanced where necessary for the map development.

4.6.1.3 API

Both developers used the full data download and did not interact extensively with the OAI-PMH interface. The OAI-PMH interface was not found to be useful and the resumption tokens were a nuisance given the size of the dataset. Following the initial data download, the Repository 66 developer then subsequently queried for each item to get its usage statistics.

OA-RJ found a number of features that would have been nice additions. Their use case is to regularly refresh their own cache and interpretation of the data, and so full data dumps are time consuming for them to process. Instead an incremental feed interface which included all the data in the raw data dump would have been valuable, but which could also be configured to omit all the administrative data (such as file references for graphs, etc) for rapid client side processing.

4.6.1.4 Data Schema

API users were satisfied with the data schema but it was noted that due to the limited use case, "any structured data format would have done". The development team at Repository66 were positive on the fact that "ROAR was able to influence changes for a CSV text dump of content statistics"

Neither developer felt that a standard set of metadata elements would have been of significant value. The schema used by OpenDOAR is sufficiently self-explanatory that the important fields were easy to identify. If the API were to be extremely heavily used, though, by a variety of consumers it was suggested that a standards-based approach might be worthwhile. The existing XML schema definition was found to be of no value due to its extreme complexity.

The format of the data is XML, but the developers would have been happy with any consistent, structured format such as JSON.

4.6.1.5 Support

Both developers contacted ROAR during their work for informal support, and were given that support outside of the normal roles of the individuals concerned. There is no ticketing system or support address. It was noted that "EPrints has a technical discussion forum, which may have sufficed, but it was unclear what information this could provide."

During some of the work, improvements were made to the ROAR data which could have been useful to feedback, but there is no formal route by which this can be done. For example, Open Access Repository Junction identified equivalences with OpenDOAR which could have been useful to either service.

4.7 Future/Technical Roadmap

The technical roadmap for ROAR is not a well defined list of objectives, and due to the fact that it is currently unfunded there are no specific project goals to meet. Therefore, if something interested is proposed to the developers then it will get done depending on its percieved value and time to implement. ROAR does have the opportunity to take advantage of developments going on in the main track of EPrints development.

The development team noted that better stats on full-text statuses would be a valuable addition.

5 The shape of a new registry

5.1 OpenDOAR and ROAR: The best bits, limitations and opportunities

This section details the bits of OpenDOAR and ROAR which worked well from a technical evaluation point of view and from the perspective of the domain experts involved in and/or interviewed during the creation of this report.

In creating the list of Best Bits and Limitations, we have attempted to normalise the comments so that all features are considered for both systems.

5.1.2 OpenDOAR

5.1.2.1 Best Bits

- The manual curation of the data is seen to add value and some trust to the data
- The coverage is unskewed, and therefore likely to be quite good; any future endeavours in this sector would be able to start with this dataset
- The graphical data analysis is extremely flexible, easy to use and illuminating
- Categorisation is of value, although it's uncertain that the specific categories are the right ones
- The different kinds of result layouts in the UI (maps, tables, charts) provide a good end-user experience
- The search and browse functions are important end-user features
- The ability for an end-user (not just the repository owner) to suggest a modification/new record is a basic attempt to crowd-source the data
- Providing Repository Cross-Search is a good value-add
- The Policies Tool (and related assistive information), although not directly related to OpenDOAR is f interest
- The use of the OAI-PMH Identify verb to add metadata is correct use of the technology
- Running a link checker on the dataset increases confidence in it
- Nothing ever gets deleted, which is good from a provenance point of view (although options to retire and supercede records would be valuable)
- Quite accurately populated (although anecdotally, there are many spelling errors in the data)
- A Full, detailed, data dump is available
- Most data in the schema is self-explanatory (although there are bits that are not)
- Periodic campaigns are run to improve parts of the metadata in the system
- It is easy enough for the limited number of client developers to engage OpenDOAR for support and provide feedback
-

5.1.2.2 Limitations

- The collection policy is extremely focussed on full-text Open Access, and some scenarios have therefore been passed over
- OpenDOAR does not include bibliographic only repositories, but the existing focus in the community on open bibliographic data would suggest that this data is useful

- No publicly available usage stats for the service
- The data schema is relatively rigid and does not accommodate all appropriate relationships between entitied
- The Search/Browse interface has the appearance of being faceted, but isn't true faceting
- The repository content counts are only collected once at registration time and are therefore of no value
- The API only provides XML data - it could be useful to provide JSON as well
- No support for multilingual data in UI, API or data schema
- No revisions history is maintained
- Better tools needed to check data quality (e.g. spellchecker)
- Anecdotally, OpenDOAR does not work so well in a machine-to-machine capacity
- There is poor support for discontinued repositories
- Large scale use of API should be better supported by standards usage
- No support for Linked Data; entities should be clearly separated and identified with URIs
- No use of standard vocab/categories (e.g. Dublin Core/Library of Congress)
- Although the API is quite complex, there is no standard client library
- Although publicly funded, the code is not available as Open Source
- There is no clear policy for getting new fields added to the schema populated, although OpenDOAR do run periodic campaigns to improve areas of the system
- The data is full of many small errors (e.g. spelling errors)
- There is poor support for asserting when one repository has superceded another
- There is no official feedback route or ticketing system

5.1.3 ROAR

5.1.3.1 Best Bits

- It is build on software which is part of a major OSS platform (EPrints)
- All OAI-PMH requests are archived in a large underlying dataset
- ROARMAP, the OA Mandates list is a good piece of added value
- The coverage is largely unskewed, and therefore likely to be quite good; any future endeavours in this sector would be able to start with this dataset
- Categorisation is of value, although it's uncertain that the specific categories are the right ones
- Usage statistics are archived (although not publicly available)
- The advanced search features are very detailed
- The self-submission process delegates the management of the records effectively to the repository owners
- The OAI-PMH harvester records any errors which occur
- A record count history is maintained
- No significant metadata migration has been performed, ever, which indicates stability in the schema
- Quite accurately populated (although anecdotally, there are many spelling errors in the data)
- A Full, detailed, data dump is available
- Most data in the schema is self-explanatory (although there are bits that are not)
- It is easy enough for the limited number of client developers to engage ROAR for support and provide feedback
- The result set of a search can be exported in a large variety of formats through the UI
- Providing Repository Cross-Search is a good value-add
- A revision history of the records is maintained, although it is not clear how to access that

from outside the system

5.1.3.2 Limitations

- It is possible, looking at the data, that ROAR is skewed towards European coverage
- The categorisation of content and repositories is confusing and contradictory
- The graph interface is awkward to use and there are rendering issues with the charts produced
- No publicly available usage stats
- ROAR is lacking in both UI and API documentation
- The Search/Browse interface has the appearance of being faceted, but isn't true faceting
- The graphs/charts are quite limited in scope
- The editorial review process provides a single point of failure
- No link-checking is run on the stored dataset
- There is no ongoing data curation at the ROAR end
- There is poor support for discontinued repositories
- No support for multilingual data in UI, API or data schema
- The record count history is difficult to use (although it is of value that it exists at all)
- The data schema is relatively rigid and does not accommodate all appropriate relationships between entitied
- There is no policy for getting new fields added to the schema populated
- There is a very high incidence of barely used fields
- The data is full of many small errors (e.g. spelling errors)
- The OAI-PMH is of no use
- The XML Schema which describes the data structure is too complex to be used as documentation
- There is poor support for asserting when one repository has superceded another
- There is no incremental API (other than OAI-PMH, which doesn't carry enough data)
- There is no official feedback route or ticketing system
- ROAR makes limited use of standards
- The API only provides XML data - it could be useful to provide JSON as well
- The many result set formats available to the UI are not available via the API
- Better tools needed to check data quality (e.g. spellchecker, link-checker)

5.2 Opportunities for a new system

This section describes the opportunies for a new system which could be derived from the best practices of OpenDOAR and ROAR as well as addressing their limitations. This could be viewed as a partial set of requirements, which should be taken along with the stakeholder requirements to specify a new kind of OARR.

5.2.1 Administrative aspects

1. Ensure there is an option for administrator manual intervention. This means the administrators get to see all new and updated records (even if there is no explicit approval process prior to being added to the registry) and being alerted to any data issues by link-checkers, spell checkers and other integrity checking software (discussed below).

2. Ensure easy support for all variations: software, region, content type, repository type. That is, to continue the work of the existing registries of maintaining an unskewed, wide coverage dataset. Ensuring that the data model and interfaces do not descriminate against certain types or users, will maximise potential for coverage; this explicitly includes being multilingual in the UI and the data

3. It should take as much advantage from crowd-sourcing as possible. End users should be able to suggest modifications of existing records, or the creation of new records, as well as asserting relationships between records in the system (such as that one repository replaces another, or that two records are actually about the same thing).

4. The collection policy should be very open, and ideally should deal with all kinds of content-bearing scholarly systems, not just repositories. There are a number of use cases which could be serviced by a registry of not just OA repositories but also full-text vs bibliographic repositories, and dark archives, or administrative archives.

5. The software must be open source. The ideal would be that anyone who wanted to could set up and run their own registry of repositories. Further that the software could be of value for other kinds of registries (for example, e-journals), and the software should be constructed in a modular way such that it can easily be extended for whatever means.

6. The administrators should have a strategy for keeping records up to date with changes to the core metadata schema

7. The service should have some formal mechanism for support of end users and developers; this would not be built into the registry software, but would be offered by the service provider.

8. Public usage stats ought to be available, although this would be at the discretion of the service provider.

5.2.2 End user functionality

9. Flexible reporting on the data. This would include chart based results and faceted browse of the datasets, for example.

10. Support and provide policy information such as collection and curation policies. This should be machine readable.

11. The API and the UI should provide all the information available to the system (with the exception of any information which may come under data protection), so that they are both fully functional and technically equivalent

12. The system must be multilingual in every way: the User Interface should be internationalised (with as many translations as possible), the data schema should support the same field in multiple languages and the data itself should be in its native language and English (if possible). The API would then offer features to filter by language and allow the developers to only retrieve the content which is relevant to them

13. Self-Submission should be a primary route for data into the system, combined with as much auto-discovery of data as possible.

14. The data should be available in a variety of formats, including, for example Google Cross Search. Note that this should be possible via both the UI and the API.

5.2.3 Data model

15. Use of standards where appropriate; in particular the use of standard categorisations for repositories and vocabularies in the API.

16. The data model should support entities and relationships, such as for organisations, people and repositories, and construct its records through sets of relationships like isPartOf, isReplacedBy, and so forth.

17. The data model should be as self-explanatory and human understandable as possible, but the system should also be properly documented.

18. Incoming content should be archived. This would include all repository records, but also all OAI-PMH feeds, usage logs, and so on. Note that this would take a lot of storage space, so there may need to be an archiving and retention policy.

19. The data schema should be trivially extensible. This may be via and extension method such as that used by Atom or a more fundamental approach like RDF.

5.2.4 Technology

20. Run data curation tasks constantly on the dataset. This includes common tasks such as link-checking and spell-checking, but there is scope for more sophisticated services: for example, flagging URLs that are "at risk" because they contain IP addresses or URL query parameters (which are likely, therefore, to be fragile); or retrieving OAI-PMH feeds and web pages and verifying the metadata (embedded in web page meta tags, say) to track changes to the services, and help identify possibly deceased systems. In the event of issues the administrators of the system as well as the owners of the record should be alerted.

21. Repository content counts should be collected regularly and recorded in an easily re-usable way.

22. The API should provide Content Negotiation, so that API developers can retrieve XML (in a variety of schemas) or JSON, or any other appropriate format

23. The system should maintain a full revisions history of its items

24. The API should leverage existing standards and approaches such as REST, ATOM, JSON, etc. It is not clear that there's a value of OAI-PMH in the system.

25. The API should come with at least one client library written in a common programming language to aid access to the content by 3rd party developers.

26. The API must support incremental requests

5.2.5 3rd Part Features

There is a lot of opportunity for 3rd party service providers to build off the data provided by a system which met the above requirements. OpenDOAR and ROAR already provide tangential services such as Repository Cross-Search and the Policy Tool.

Further to this it is possible to imagine domain specific interfaces and search features built from the API, and custom reporting services based on the data, as well as much better servicing existing 3rd party systems such as OA-RJ and Repository 66.

6 References

[1] http://www.opendoar.org/about.html
[2] http://www.sherpa.ac.uk/
[3] http://www.sherpa.ac.uk/romeo/
[4] http://www.sherpa.ac.uk/juliet/index.php
[5] http://www.opendoar.org/tools/en/policies.php
[6] http://www.opendoar.org/documents/BergenPresentation20060512Handouts.ppt
[7] http://www.doaj.org/doaj?func=byCountry&uiLanguage=en
[8] http://maps.repository66.org/
[9] http://oarepojunction.wordpress.com/
[10] http://irs.mimas.ac.uk/
[11] http://www.base-search.net/
[12] http://www.opendoar.org/find.php?format=charts
[13] http://arxiv.org/
[14] http://www.opendoar.org/search.php
[15] http://www.opendoar.org/tools/emailservice.html
[16] http://www.opendoar.org/suggest.php
[17] http://www.opendoar.org/tools/api13dtd.html
[18] http://www.opendoar.org/tools/api13manual.html
[19] http://www.opendoar.org/api13.php
[20] https://bitbucket.org/richardjones/oarr/src
[21] http://linkeddata.org/
[22] http://eprints.whiterose.ac.uk/
[23] http://roar.eprints.org/information.html
[24] http://roarmap.eprints.org/
[25] http://roar.eprints.org/cgi/roar_graphic?cache=973936
[26] http://roar.eprints.org/cgi/search/advanced
[27] http://roar.eprints.org/cgi/roar_graphic?cache=984627
[28] http://roar.eprints.org/cgi/schema