If SWORD is the answer, what is the question?

Use of the Simple Web-service Offering Repository Deposit protocol

Stuart Lewis, Leonie Hayes and Vanessa Newton-Wade
The University of Auckland Library, New Zealand
Antony Corfield
Aberystwyth University, Aberystwyth, UK
Richard Davis
University of London Computer Centre, London, UK
Tim Donohue
University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, and
Scott Wilson
University of Bolton, Bolton, UK

Received 28 May 2009
Revised 6 July 2009
Accepted 10 July 2009

Abstract

Purpose – The purpose of this paper is to describe the repository deposit protocol, Simple Web-service Offering Repository Deposit (SWORD), its development iteration, and some of its potential use cases. In addition, seven case studies of institutional use of SWORD are provided.

Design/methodology/approach – The paper describes the recent development cycle of the SWORD standard, with issues being identified and overcome with a subsequent version. Use cases and case studies of the new standard in action are included to demonstrate the wide range of practical uses of the SWORD standard.

Findings – SWORD has many potential use cases and has quickly become the *de facto* standard for depositing items into repositories. By making use of a widely-supported interoperable standard, tools can be created that start to overcome some of the problems of gathering content for deposit into institutional repositories. They can do this by changing the submission process from a "one-size-fits-all" solution, as provided by the repository's own user interface, to customised solutions for different users.

Originality/value – Many of the case studies described in this paper are new and unpublished, and describe methods of creating novel interoperable tools for depositing items into repositories. The description of SWORD version 1.3 and its development give an insight into the processes involved with the development of a new standard.

Keywords Digital storage, Information management, Protocols, Standards

Paper type Technical paper

The authors would like to acknowledge the generous support of the JISC in funding both of the SWORD projects. They would also like to acknowledge the entire SWORD project team who collaboratively worked to develop and implement the SWORD standards.

1. Introduction

Modern information systems can only truly thrive when they interoperate with other systems. Institutional repositories are no exception to this rule. Repository software can typically interoperate with other systems in three ways:

(1) to get things into the repository;
(2) to get things out of the repository;
(3) to assist the repository with tasks such as preservation, storage, or authentication.

SWORD (Simple Web-service Offering Repository Deposit) is a relatively new standard that exists to fulfil the first of these interoperability types: depositing items in a repository.

The first development iteration of the SWORD standard was funded by the UK Joint Information Systems Committee (JISC) and is described in Allinson *et al.* (2008). Despite being implemented in the three leading open-source repository platforms (DSpace, EPrints and Fedora) and one commercial learning object repository (Intrallect's Intralibrary) and receiving a lot of attention in the repository world, there has been very little actual use of the standard. This can possibly be attributed to the following reasons:

- A lack of real understanding about how the standard works and where it can be used.
- A lack of tools to assist with the development of SWORD-based solutions.
- Issues with the SWORD standard that complicate its implementation.
- Only institutions running up-to-date versions of their repository software will have the SWORD functionality.

To help address some of these problems, the original SWORD development team received further funding from JISC to carry on the work of developing the standard. This paper describes the second iteration of the development of the SWORD technical standard, and aims to assist the repository and library communities in understanding the potential uses of SWORD by describing seven different cases studies where SWORD is now, or will shortly be, in use. The case studies will describe some of the scenarios where SWORD provides an ideal answer to particular types of problem. If SWORD is the answer, what is the question . . .?

2. Background

The first SWORD project developed the SWORD specification. It is a specialised profile of the Atom Publishing Protocol (2007) that facilitates the deposit of items into repositories. One of the biggest selling points for SWORD, and the first thing to note, is that rather than designing a new standard, SWORD was built upon an existing successful standard: AtomPub.

AtomPub allows the reading and writing of files and feeds to remote systems. AtomPub is already used in many different systems including GData (http://code.google.com/apis/gdata/), the data interface that powers many of Google's systems such as Google Docs. Subsequent to its use in SWORD, AtomPub has also been adopted by the Jangle interoperability specification (www.jangle.org/), which is designed to provide a common interface to library systems.

SWORD interacts with repositories in two ways:

(1) A SWORD client can request a service document from a SWORD server (the repository). The service document lists which collections the user is allowed to deposit items into, and what type of objects it accepts.

(2) A SWORD client can post a file to a SWORD server. The server will then process the deposit and respond accordingly by returning an Atom entry, which contains the URL of the new item that was created as a result of the deposit.

Users typically need to provide their login credentials when requesting a service document or performing a deposit. This allows the repository to tailor its responses and authorisations to the particular user. SWORD is a profile of AtomPub as it makes use of custom extensions that allow it to fit in with the way repositories work. One such extension is the ability for SWORD clients to deposit items on behalf of other users. For example, an administrator may deposit items on behalf of a user (mediated deposit), or a central publishing system may deposit on behalf of an author. In this scenario, the repository must check not only the credentials of the depositing user, but also that the mediating user has authorisation to deposit on behalf of the author.

The first SWORD project, which ran from March to October 2007, developed initial versions of the standard (0.1, 0.2, 0.3, 0.4, 0.5, 0.6 and 0.7, and then released version 1.0. This was subsequently replaced with versions 1.1 and 1.2, which corrected minor typographical errors in the specification and made some recommendations about how SWORD should be used. It funded the implementation of SWORD in the three major open source repository platforms DSpace, EPrints and Fedora, and in a commercial learning object repository Intrallect Intralibrary. The final outcome of the project was some case studies from people who had tried to use the standard in order to gather some real-life feedback (www.swordapp.org/sword/case-studies).

3. The evolving standard

During late 2007 and early 2008 SWORD received a lot of attention in the repository world. This was probably due to the fact that prior to SWORD there was no standardised deposit protocol for repositories, and that interoperability could now occur as SWORD has been implemented by the most widely adopted open-source repository platforms. Despite this level of attention there was very little use of the standard. Further funding was provided by JISC in 2008 to allow a second SWORD project to take place. The second project decided to address two areas: the lack of adoption of the standard, and the issues highlighted by the case studies produced by the first project.

In order to address the lack of adoption, it was decided that the project should undertake a number of steps:

(1) *Create a project web site*. The documentation for the first iteration of SWORD had all taken place on part of a wiki that was hosted by UKOLN (www.ukoln.ac.uk), the research organisation in the UK that aims to inform practice and influence policy in the areas of digital libraries, information systems, bibliographic management, and web technologies. Whilst this was an adequate tool, it was thought that, being part of a larger wiki, it was hard to find and did not carry its own identity. The text on the wiki was mainly the specification, which naturally needed to be expressed in technical language. A website was created with its own domain (www.swordapp.org) and pages

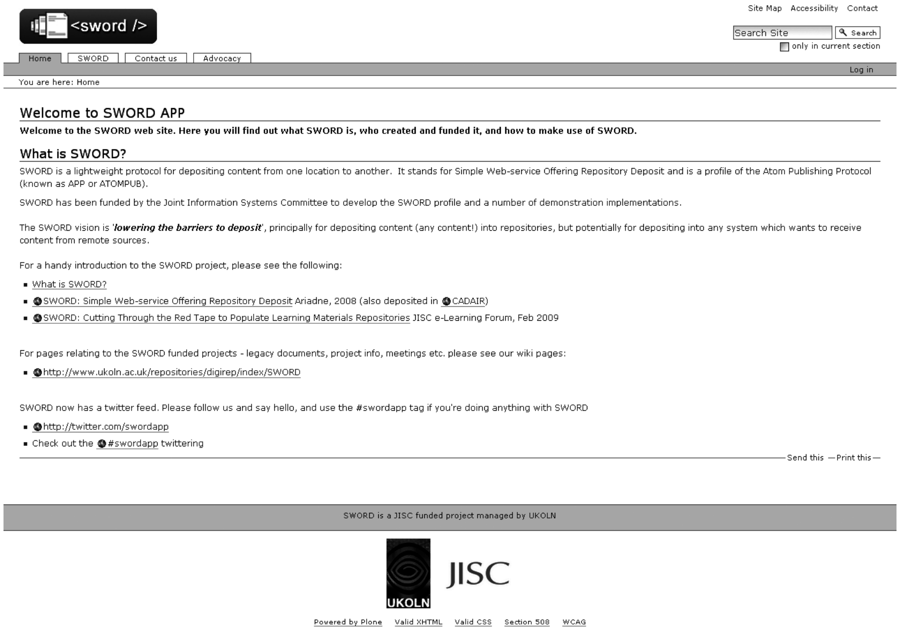
explaining SWORD in less technical language were written. The domain also allowed the other outputs of the project to be branded in a similar fashion:

- the web-based SWORD client (http://client.swordapp.org);
- the PHP code library (http://php.swordapp.org);
- the SWORD validator (http://validator.swordapp.org);
- the demonstration SWORD-enabled DSpace repository (http://dspace.swordapp.org).

Figure 1 shows the opening page of this web site.

(2) *Create a PHP SWORD client library*. As well as the repository implementations of SWORD, a client library was written in Java along with a command line interface, a graphical user interface, and a web interface (www.swordapp.org/sword/demonstrators). Whilst these demonstration interfaces worked well, it was decided to write a client library in the more lightweight and popular web scripting language PHP. It is much easier to write and host small PHP applications than Java applications, so it was hoped that by creating a PHP version of the SWORD client library (http://php.swordapp.org) we could facilitate the creation of new SWORD clients. Depositing a package using the SWORD PHP library can be as simple as using the following code:

- include (""swordappclient.php"");
- $swordappclient = new swordappclient();
- $response = $swordappclient- > deposit($depositUrl, $username, $password, $onBehalfOf, $filename, $formatNamespace, $mimeType);
- $depositId = $response- > sac_id;



You are here: Home

Welcome to SWORD APP

Welcome to the SWORD web site. Here you will find out what SWORD is, who created and funded it, and how to make use of SWORD.

What is SWORD?

SWORD is a lightweight protocol for depositing content from one location to another. It stands for Simple Web-service Offering Repository Deposit and is a profile of the Atom Publishing Protocol (known as APP or ATOMPUB).

SWORD has been funded by the Joint Information Systems Committee to develop the SWORD profile and a number of demonstration implementations.

The SWORD vision is '***lowering the barriers to deposit***', principally for depositing content (any content!) into repositories, but potentially for depositing into any system which wants to receive content from remote sources.

For a handy introduction to the SWORD project, please see the following:

- What is SWORD?
- SWORD: Simple Web-service Offering Repository Deposit Ariadne, 2008 (also deposited in CADAIR)
- SWORD: Cutting Through the Red Tape to Populate Learning Materials Repositories JISC e-Learning Forum, Feb 2009

For pages relating to the SWORD funded projects - legacy documents, project info, meetings etc. please see our wiki pages:

- http://www.ukoln.ac.uk/repositories/digirep/index/SWORD

SWORD now has a twitter feed. Please follow us and say hello, and use the #swordapp tag if you're doing anything with SWORD

- http://twitter.com/swordapp
- Check out the #swordapp twittering

Send this — Print this —

SWORD is a JISC funded project managed by UKOLN

Figure 1.
SWORD project web site opening page

(3) *Create a Facebook SWORD client*. To demonstrate the PHP SWORD client, and to show how SWORD clients can be built into other web applications, a Facebook SWORD client was written using the PHP code library. The Facebook client and the motivations behind its development are described in more detail in a case study later in this paper.

(4) *Create a SWORD validator*. When developing software that works with a given standard, it is useful if a validator is available that will compare the output generated by a piece of software with the standard. In the same way as there are validators for other web standards such as HTML, CSS and RSS, it was decided to create a validator (http://validator.swordapp.org) that would integrate with the SWORD web client to highlight any deviations from the standard in the responses generated by a SWORD server.

The case studies and the experiences of the SWORD team in implementing the SWORD standard highlighted some problems with the SWORD standard. In order to overcome these problems, some changes to the SWORD specification were required. The changes made were:

(1) *Created SWORD error documents*. It was noted in the ArXiv case study (Warner, 2008) that the method for reporting errors was deficient in two ways. First, it replied on the use of 4xx and 5xx HTTP response codes and a custom X-Error-Code header in the response. This was not always helpful, and did not allow the repository to add extra information that may be useful in explaining the cause of the error. Secondly, errors were not reported in an XML format that the client would be expecting. The case study proposed returning an error document that was an atom entry in its own right, and could therefore be easily parsed by a SWORD client that was expecting an atom entry document as the response to the deposit. This recommendation was adopted in the updated SWORD standard.

(2) *FormatNamespace and acceptPackaging*. When a client deposits an item, it can inform the server about what it is depositing in two ways. The first is to state the MIME (Multipurpose Internet Mail Extensions) type of the file. This describes the type of file such as text file, jpeg image, PDF document etc. However, sometimes this level of information is not sufficient. If a package of information is being deposited, it is often in the form of content files and metadata combined into a single file such as a zip archive. In this case, it is not enough to know that the package being deposited is a zip file. A description of the contents of that file must also be given. In the first version of the SWORD standard, servers could state which formats of package they would accept by detailing this in the formatNamespace element of service documents. The same element was used in deposits to, and responses by, the server. For clarity, this element was renamed acceptPackaging in SWORD version 1.3. In addition, a quality value mechanism was added to this element to allow servers to describe their relative preference for different package types. A typical server response might be that the server accepts packages in both the DSpace METS Submission Information Package and the Bagit package, but it shows a preference towards the former:

```
< sword:acceptPackaging    q = ""1.0""    > http://purl.org/net/sword-types/
METSDSpaceSIP < /sword:acceptPackaging >
```

< sword:acceptPackaging q = ""0.8"" > http://purl.org/net/sword-types/bagit < / sword:acceptPackaging >

(3) *Replaced "level" with "version"*. The SWORD 1.2 standard allowed SWORD servers to state their level of support for SWORD. Two levels were available. By specifying compatibility with level 0, a SWORD server stated that it knew about SWORD, but did not implement any of the SWORD extensions and was just a plain AtomPub interface. If a SWORD server specified compliance with SWORD level 1, then it stated that is was fully compliant with all of the SWORD extensions. It was agreed that these levels were to some extent redundant, as level 0 was in fact no different to a non-SWORD-compliant AtomPub interface, except for that fact that it could describe itself as such. In place of the "sword:level" extension, a new extension "sword:version" was included to allow a SWORD interface to state which version of the SWORD standard it implemented.

(4) *New "service" element.* When a SWORD client requests a service document, an XML document containing details of the repository is returned. One section of the response lists "collections" (targets) that the user can deposit items into. It may be that in a given repository there are many target collections. For example, if a repository decided to structure its collections by "subject", there could be thousands of collections. A service document describing so many collections would be very large, and may possibly be too large to transport and parse efficiently. To ameliorate the issue, a new SWORD extension element sword:service was introduced. This allows nested service documents to be listed. This would allow a client to see the top-level collections, and then traverse their way down through the levels of child collections.

(5) *New "maxUploadSize" element.* Repositories will have varying collection policies. One aspect of a collection policy may be the maximum permitted size of a new item. To allow SWORD to express this preference, a new element "sword:maxUploadSize" was added to the service document vocabulary to define the maximum size of deposit that the SWORD service would accept. The value is described in kilobytes.

(6) *User agent header.* In much the same way as web browsers identify their name and version number to web servers, it was decided to adopt the same mechanism in SWORD. When depositing an item, SWORD clients can set a HTTP header "User-Agent". This value may be used by the server to decide how to process the deposit, or it may be ignored. The value provided by the client is echoed back by the server in the atom entry it returns.

4. SWORD use cases

SWORD is an ideal solution for putting objects into a repository. There are different use cases that may be encountered where users wish to facilitate the deposit of items into a repository without having to use the native user interfaces provided by the repository. Some of these use cases are described here:

(1) *Deposit from within an authoring tool.* It may be beneficial to authors if they can upload a document directly into a repository from within the tool that they use to author their documents. This use case could be particularly useful where a given template should also be used, for example when writing a paper for a

conference. Microsoft Research has recently created such a tool for Microsoft Word 2007 called the Article Authoring Add-in (http://research.microsoft.com/en-us/projects/authoring/). It allows a template to be created that can be configured to make certain fields mandatory (e.g. title, author(s), abstract), and can have the details of a SWORD deposit URL embedded. The tool ensures that the mandatory sections are completed, and then allows the user to upload the document to the repository.

(2) *Create a custom deposit interface.* Sometimes the deposit user interface within a repository platform may not be the best interface for some groups of users. By using SWORD, custom deposit tools can be created specifically for a given purpose (e.g. submitting electronic theses).

(3) *Deposit to multiple locations.* Some items may require depositing into multiple locations. For example this paper, written by many authors from different institutions all over the world, may need to be deposited into the local institutional repository of each of the authors, a subject repository such as e-Lis (http://eprints.rclis.org) covering Library and information Science, and the funder's repository (http://ie-repository.jisc.ac.uk) covering work funded by JISC. If each of these repositories had a SWORD interface, the paper could be deposited into all of them in a single deposit action by the lead author, were such a tool to exist.

(4) *Allow non-human deposits*. It is not only human authors who have useful items to deposit into repositories. Laboratory equipment or monitoring stations may be configured to deposit items automatically.

(5) *Copy from an information system into a repository*. It could be that content held in one system (e.g. an e-learning system or blogging software) could be archived for safe-keeping into a repository where long-term preservation and curation can occur. Alternatively, a repository may wish to deposit an item into another repository automatically.

The next section details case studies where SWORD has been used in some of these situations.

5. SWORD case studies

5.1 Customised e-thesis deposit tool (Leonie Hayes, University of Auckland)

In 2007 The University of Auckland adopted a self-deposit mechanism for newly graduating doctoral students to deposit their completed PhD theses. The reason to adopt this self-deposit mechanism has been to ensure the sustainability of the service by using library staff resources to grow the acquisition of new content. This model has been very successful and over 1,300 PhD theses have been deposited by doctoral students (http://hdl.handle.net/2292/2).

We have found that the DSpace deposit system has not allowed us to be agile in changing deposit procedures or to modify text. For some time we have been seeking ways to take the deposit system out of the repository software, but still take advantage of the formal deposit checks, processes and work flows that are available within DSpace.

SWORD is able to meet all these requirements. So, we created a thesis deposit prototype for library management to assess in preparation for a formal launch to students in mid 2009. The SWORD deposit system is tailored precisely to the needs of

PhD candidates, and includes specific questions about licensing and the optional choice of applying Creative Commons licences to the thesis. The system also allows us to insert steps during the submission that reflect back to the students their licence choice to ensure that they fully understand the implications of the licence they chose. An e-mail containing the same information is sent as a receipt to the students so that they have a lasting copy of the terms that they agreed to, along with a contact address if they wish to change the licence.

Because the thesis deposit system is built upon the SWORD standard, were we to change our underlying repository platform we know that our thesis deposit system will continue to work without needing a lot of custom integration to be required for the new platform.

5.2 BibApp (Tim Donohue, University of Illinois)

BibApp (www.bibapp.org) is a campus research gateway and expert finder system used at the University of Illinois at Urbana-Champaign in the USA and written using Ruby on Rails (http://rubyonrails.org). It matches researchers with their publication data and allows users to locate subject experts as well as visualise collaborations across campus. BibApp acts as a bibliographic data repository and provides basic authority control and author disambiguation services, allowing the contained data to be cleaned. It also promotes the re-use and "remixing" of this data in other complementary systems through a variety of web services.

One such complementary system can be the local institutional repository. With most institutional repository platforms, individual authors must manually enter metadata describing their research article in order to complete a deposit. One of the goals of BibApp was to simplify this time-consuming process and allow the re-use of existing bibliographic data to create a "one step" deposit process. Using BibApp, an author needs only to upload the files associated with a publication and agree to the repository licence agreement to complete the deposit.

To accomplish this goal, we built our own SWORD Ruby on Rails plugin. This plugin allows any Ruby on Rails application to perform pre-packaged deposits into any SWORD-compliant repository. The SWORD protocol allows BibApp to keep its repository integration generic. The BibApp software itself remains unaware of the exact repository software solution. Instead, BibApp is provided with a valid repository login (for authentication purposes) and a collection in which to deposit its content via SWORD. Once the content is accepted into the repository, BibApp receives and retains a link to its final resting place.

BibApp attempts to ease the visualisation, searching and re-use of local publication data in order better to promote research on campus. The marriage of SWORD with BibApp allows for the re-use of this data into the local institutional repository, thereby easing the population of the repository and increasing the visibility of the contained research.

5.3 FeedForward (Scott Wilson, University of Bolton)

FeedForward (www.getfeedforward.org) is a desktop application, developed at the Centre for Educational Technology and Interoperability Standards (CETIS) at the University of Bolton in the UK, which allows users to aggregate and mix feeds from various sources, including academic journals, and to organise and share items using services like social bookmarking tools, blogs and Twitter. We also wanted to enable users to work with repositories, and so we implemented SWORD as an easy way to let users share both individual items and also collections of items that they had picked. For example, a lecturer could collect interesting articles in a "reading list" feed and then deposit this into the UK national online learning materials repository JORUM (www.jorum.ac.uk), for students to access.

We wanted to enable the process of sharing in this way to work transparently, and be just as easy as our implementations of Web 2.0 services such as Twitter. Thankfully, SWORD supported our need for simple drag-and-drop of items. The only difficult aspect of the implementation was the packaging of items as we didn't want to expose the details of composite formats, such as IMS Content Packaging or Open Archives Initiative – Object Reuse and Exchange (OAI-ORE), to users, rather we wanted to make the process of packaging up materials transparent. Also, the SWORD workflow meant that we had to offer a "wizard" to users to help them set up their accounts; this was slightly tricky as no auto-discovery of SWORD service endpoints was available, so users need to know the actual URL of their repository's SWORD service document. In contrast, when FeedForward users are setting the application to post to their blog, only the blog URL is needed as we can auto-discover the Atom service document. Hopefully this will be resolved in SWORD repository implementations in the future.

5.4 Computer science reports custom deposit tool (Vanessa Newton-Wade, University of Auckland)

ResearchSpace@Auckland (http://researchspace.auckland.ac.nz) is a digital repository for University of Auckland digital theses and research materials in New Zealand. It was contacted in April 2009 by staff from the university's Computer Science department as papers were already maintained in the department's own system, and were available on the departmental website. The department was initially interested in archiving only the completed report series as they did not want to add to the administrative workload by maintaining the active series in two places. However, we wanted to archive both sets of reports.

SWORD provided an effective solution to this issue. We were able to configure SWORD so that the Computer Science department enter the details of a paper once into a custom SWORD client, and it is deposited directly into ResearchSpace@Auckland. The metadata is then "bundled up" and returned to the department in the format that their system requires, and is then entered into their system via a simple cut and paste. The end result is an easy deposit into two systems, and the only extra work for the department is the final cut and paste. As a result of the simplified deposit process, the Computer Science department agreed to archive both the completed and active report series.

One of the reasons for low uptake of institutional repositories at the departmental level is the extra effort involved for depositors, particularly when a report series is already maintained elsewhere, as in the case described above. As a result, this application of SWORD is likely to provide a good marketing tool for deposit of academic research into ResearchSpace at the faculty/departmental level.

5.5 CLASM (Richard Davis, University of London)

The University of London Computer Centre (ULCC) in the UK began a six-month JISC-funded project in 2009 called CLASM: Copyright Licensing Application with SWORD for Moodle (http://clasm.ulcc.ac.uk). The aim of the project is to develop a

SWORD plugin for the open-source virtual learning environment (VLE) Moodle (http://moodle.org), so that it can interact, platform independently, with common repository platforms; and to explore and demonstrate the use of that plugin for managing copyright licensed materials in Moodle courses. ULCC hosts many Moodle VLE instances for institutions and colleges in and around London, as well as several institutional and specialist repositories, based on EPrints, DSpace and custom solutions.

The issue of managing Copyright Licensing Agreement (CLA) (www.cla.co.uk) materials has been identified by many VLE users. We are all familiar with photocopied or scanned copies of book chapters and articles that tutors use in their teaching: CLA sets out a number of conditions under which these copyrighted resources may be used in teaching, and in theory the use made of such materials by institutions should be auditable. VLEs don't necessarily easily expose all the "attachments" uploaded for use in their many courses and modules in a consistent way that might make such an audit feasible. At the same time, the superior bibliographic and resource-management features of e-repositories seemed to offer a promising approach to managing these objects more effectively for tutors, students and library staff, while making them available within a VLE, in accordance with CLA terms and conditions.

The benefits of a SWORD-based approach are that a single solution can be developed without predetermining the repository platform that must be used. The best way of utilising the repository has yet to be determined by the CLASM project: for example, a separate "dark" repository might be established, or a protected collection within an existing repository, only accessible by the VLE and repository administrators. Project outputs will include recommendations for metadata and repository configuration that should be applicable across all major repository platforms; but, by virtue of addressing the SWORD layer, rather than native repository APIs (Application Programming Interfaces), the Moodle plugins developed will interface with any SWORD-compliant repository system. Furthermore, existing examples of SWORD-based widgets for other applications (Facebook, Netvibes, iGoogle) and the availability of PHP libraries suitable for embedding in Moodle plugins, offer the project team a high degree of confidence in a successful outcome.

5.6 Facebook deposit tool (Stuart Lewis, University of Auckland)

Social networking tools such as Facebook (www.facebook.com) have become very popular over the past couple of years for disseminating information about people's personal lives and work. With many universities now having a Facebook presence, it is likely that the number of active research staff with Facebook profiles outweighs the number who have deposit accounts with their local institutional repository by several orders of magnitude. In addition, researchers probably use Facebook as a way of collaborating and sharing with other members of their research community more than they currently do with their repositories, which generally contain none of the "Web 2.0" tools that allow tagging, commenting and feedback to take place.

The Facebook deposit tool (http://apps.facebook.com/swordapp/) was developed to demonstrate how the two worlds of social networking and institutional repositories can work together and is described in Lewis (2008). It enables deposits of research items into local repositories while making use of the social networking ethos through facilities such as alerting friends of new deposits, seeing deposits made by friends, and allowing comments to be left about deposited materials.

SWORD is used as the deposit interface into repositories. Users log in with their local institutional login and password and are presented with a list of collections into which they can deposit items. They then enter some simple metadata (author names, title, abstract and citation), upload a file, and the deposit is made. A note is then posted on their profile announcing the deposit along with the item's URL from the repository.

While not a serious system designed for production quality use, it demonstrates what could be termed a "social deposit", which allows items to be deposited from within a social network allowing the benefits of both the social network and the repository to be harnessed.

5.7 ROAD simultaneous data deposit (Antony Corfield, Aberystwyth University)

The Computational Biology Group at Aberystwyth University has developed a "Robot Scientist" (King *et al.*, 2004) that is capable of automatically carrying out cycles of scientific experimentation. That is, the Robot Scientist generates its own hypotheses to explain observations, devises experiments to test these hypotheses and physically carries out the experiments. It is capable of initiating over 1,000 experiments and making over 200,000 observations per day, which generates over 1 Gigabyte of data and experiment metadata.

Between 2007 and 2009 the JISC-funded Robot-generated Open Access Data (ROAD) Project investigated the use of current open-source digital repository platforms (DSpace, EPrints and Fedora) to enable the automatic curation of the robot-generated data and metadata. Each of the repository platforms have command line tools or services for bulk ingest of Submission Information Packages (SIPs), however, the format of the metadata schemas and SIPs varies between the repository platforms. The original project plan involved writing custom software for each repository platform to allow a common SIP to be deposited in each. When SWORD was implemented in each of the repositories, it became a tool that allowed a standard method to be used across all three repositories rather than using the ingest tools specific to each repository.

Each repository platform's implementation of SWORD supports the zip packaging format with a Metadata Encoding and Transmission Standard (METS) document acting as a wrapper for the item's metadata, which reflects the EXPO ontology (Soldatova *et al.*, 2006) to describe the experiment that took place. EXPO defines over 200 concepts for creating semantic markup about scientific experiments, using the Web Ontology Language – OWL. METS is commonly used as a standardised XML encoding for transmission of complex digital library objects between systems and the Harvard Java Toolkit (http://hul.harvard.edu/mets/) was used for construction, validation, marshalling and serialising of the METS. The use of SWORD dramatically reduced the implementation time and cost of the common submission system.

6. Going forward

This paper has described some of the types of situation where SWORD can provide an answer, and has described how SWORD has changed over time in order to allow it to more easily work in the repository environment. What is next for the SWORD standard?

The standard will have to continue to evolve at a technical specification level. It will need to evolve to keep up with the changes in the AtomPub protocol of which it is an extension, and it will need to keep up with any changes in the repository environment. One example of such a change might be the introduction of support for "multipart" deposits (http://tools.ietf.org/html/draft-gregorio-atompub-multipart-04). Multipart deposits allows multiple files to be uploaded in one transaction rather than having to combine the files into a package such as a zip file.

Another important area with which SWORD will need to engage is the discussions around the area of packaging standards. At present there is no real consensus on a set of standard packing formats that repositories should all support. Until this happens, and the repositories all implement the same standard package formats, it will not be possible to harness the full power of interoperability that could be delivered by SWORD. It is clear that work is needed within the repository community to ensure the implementation and support of existing packaging standards, or the development of new common standards. An alternative interim route to package standardisation could be to develop and use “packaging proxy” services that convert between package types. Such proxies could work dynamically to convert to suitable packaging standards based upon the preferences of a SWORD server as described in its service document.

At the practitioner level, more advocacy about SWORD, how it works, and what it can do, is required. To facilitate this, more tools and SWORD code libraries will be required to make the process of creating SWORD-enabled applications easier for developers.

Whatever the future holds for SWORD, it looks like the standard is here to stay. With the number of projects using SWORD growing month by month, and the recent addition of the award for “most innovative JISC repository project” presented to the SWORD project at the 2009 Repositories and Preservation programme meeting (www.ukoln.ac.uk/news/get/2009/05/11/sword-wins-innovation-award-at-jisc-event/), the future looks bright for SWORD.

What questions do you have that SWORD can answer?

References

Allinson, J., François, S. and Lewis, S. (2008), “SWORD: Simple Web-service Offering Repository Deposit”, *Ariadne*, No. 54, available at: www.ariadne.ac.uk/issue54/allinson-et-al/ (accessed 30 June 2009).

Atom Publishing Protocol (2007), “Internet engineering task force trust”, available at: http://bitworking.org/projects/atom/rfc5023.html (accessed 30 June 2009).

King, R.D. *et al.*, (2004), “Functional genomic hypothesis generation and experimentation by a robot scientist”, *Nature*, No. 427, pp. 247-52, available at: http://dx.doi.org/10.1038/nature02236 (accessed 30 June 2009).

Lewis, S. (2008), “Launched today: the Facebook repository deposit tool”, available at: http://blog.stuartlewis.com/2008/11/17/launched-today-the-facebook-repository-deposit-application/ (accessed 30 June 2009).

Soldatova, L., Clare, A., Sparkes, A. and King, R.D. (2006), “An ontology for a robot scientist”, *Bioinformatics*, Vol. 22 No. 14, pp. 464-71.

Warner, S. (2008), “Case study: SWORD implementation for arXiv.org”, available at: www.swordapp.org/sword/case-studies/sword-case-study-arxiv (accessed 30 June 2009).

Corresponding author
Stuart Lewis can be contacted at: s.lewis@auckland.ac.nz