This is a repository copy of SWORD : simple web service offering repository deposit.

White Rose Research Online URL for this paper:
http://eprints.whiterose.ac.uk/3782/

Conference or Workshop Item:

Reuse
Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.
Simple Web-service Offering Repository Deposit
Quick introduction

SWORD?
SWORD

- Simple Web service Offering Repository Deposit
- JISC funded project 2007
- To provide a standard mechanism for 'doing deposit' into repositories
- Small amount of continuation funding for SWORD II (pending)
SWORD : what it is

- A lightweight protocol for deposit
- A profile of the Atom Publishing Protocol
- Implementations of the SWORD deposit interface in IntraLibrary, Fedora, DSpace and EPrints
- Three java deposit clients – web-based, command-line and desktop
- NOT a solution to metadata/packaging decisions
Before SWORD there was deposit API
Deposit API

- Discussions at the JISC-CETIS Conference 2005 focused on the lack of a deposit ‘standard’

- Rachel Heery and Repositories Research Team at UKOLN facilitated a working group of repository developers

- to address this requirement for a standard interface for deposit
Motivations

- no standard interface for tagging, packaging or authoring tools to upload objects into a repository
- no standard interface for transferring digital objects between repositories
- no way to deposit into more than one repository with one ‘click’
- no way of initiating a deposit workflow from outside a repository system
Deposit API - achievements

- Common agreement
- Scope and definitions
- Requirements and parameters
- Scenarios
- Outline approach
- Draft XML serialisations
But

- Progress ground to a halt after July 2006 meeting
- Difficult to keep momentum without a formal project and without money!
- November 2006, JISC funding call explicitly mentioned ‘deposit’ as an area for funding proposals ...
JISC to the rescue

Repositories and Preservation Programme,
Tools and Innovations strand
The project, the funding

Six months (plus slight extension) funded under the tools and innovations strand of the JISC Repositories and Preservation Programme, in March 2007

SWORD partners:

- UKOLN, University of Bath (Project management and dissemination) – Julie Allinson
- University of Southampton (EPrints) – Les Carr, Seb Francois
- University of Aberystwyth (DSpace, Fedora, reference client) – Stuart Lewis, Neil Taylor, Glen Robson, Richard Jones
- Intrallect (IntraLibrary) – Martin Morrey, Sarah Currier
- Plus some friendly advisors – Jim Downing, Richard Green
SWORD – the acronym

- **Simple** – lightweight, agile and fit-for-purpose
- **Web service** – independent of proprietary software, supports standard interfaces
- **Offering**
- **Repository** – or any system which wants to put or receive content
- **Deposit** – or put, or post, or register, or add – a little step in the ingest workflow
But why ... ?

slicing up the scenarios
Use cases

- Deposit from a Desktop/Online tool - more on next slide
- Multiple deposit - e.g. deposit to institutional and (mandated) funders’ repository with one action
- Machine deposit - e.g. automated deposit from a laboratory machine
- Migration/transfer - e.g. to a preservation service
- Mediated deposit - e.g. deposit by a nominated representative, to additional repositories
Desktop tools

- Desktop ‘smart’ deposit client
  - reference client deposit through SWORD interface
- Feedforward personal information organiser
  - [http://legolas.cetis.ac.uk](http://legolas.cetis.ac.uk)
- Potential for a Flickr-style batch uploader
- Facebook plug-in?
- Deposit as ‘save as’
  - e.g. from within word processing software
Making scenarios happen

the work
Parameters – mandatory

- Mandatory (level 0)
  - deposit any type of content
  - repository or collection id
  - identifier
  - deposit status (accepted, rejected, error), error codes, error description
Parameters - optional

Optional (level 1 mandatory)

- mediated deposit
- repository / collection name
- collection policy, description
- accepted formats
- format namespace
- source repository
- checksum
- compliance level
- additional identifiers
Future proofing with layers

- Layered approach

- Two levels of compliance
  - Level 0 compliance requires a set of mandatory elements
    - and a set of optional elements
  - Level 1 offers a set of additional elements for richer functionality
    - Mandatory at level 1
Offering Services

- Explain/Discover service
  - can I deposit? to which collections?
  - what can I deposit? what are the policies?

- Deposit service
  - deposit accepted
  - receipt returned to depositor
Existing standards

- WebDAV (http://www.webdav.org/)
- JSR 170 (http://www.jcp.org/en/jsr/detail?id=170)
- SRW Update (http://www.loc.gov/standards/sru/)
- Flickr Deposit API (http://www.flickr.com/services/api/)
- Fedora Deposit API (http://www.fedora.info/definitions/1/0/api/)
- OKI OSID (http://www.okiproject.org/)
- ECL (http://ecl.iat.sfu.ca/)
“the Atom Publishing Protocol is an application-level protocol for publishing and editing Web resources”

- **benefits**
  - supports many of our parameters and requirements, in particular file deposit
  - it already exists and has growing support
  - it is well-used in popular applications
  - it has an extension mechanism
  - Google have created their own profile (gdata)
  - good fit with the Web architecture

- **drawbacks / risks**
  - too much of a retrofit?
  - it is designed for a single package/file OR an atom document — this means that we need to package up metadata and files
SWORD profile of APP

- ‘POST’ only - SWORD does not deal in update/delete
- ‘POST’ binary files only - SWORD does not (currently) specify how to post ATOM documents
- Categories not used
- SWORD extensions
  - HTTP Header extensions
  - APP / ATOM extensions
- Recommendations for discovery
  - accessed from /sword-app/
  - service document at /sword-app/servicedocument
  - use of <link> header to point to sword implementation
How it works ...

- APP works by issuing HTTP requests (GET, POST)
  - GET Service Document (explain/discover)
  - POST ATOM document or file to collection URI
- HTTP response and ATOM document is returned
- HTTP basic authentication is required
Parameters - mandatory

- Mandatory (level 0)
  - deposit any type of content - APP yes
  - repository or collection id - APP yes
  - identifier - APP yes
  - deposit status (accepted, rejected, error), error codes, error description - APP yes (and extension)
  - compliance level - extension
Parameters - optional

- Optional (level 1 mandatory)
  - mediated deposit - extension
  - repository / collection name - APP yes
  - collection policy, description - extension
  - accepted formats - APP yes
  - format namespace - extension
  - source repository - APP yes
  - checksum - extension
  - compliance level - extension
  - additional identifiers - APP yes
Examples – GET (explain)

GET /sword-app/servicedocument
HTTP/1.1
Host: www.myrepository.ac.uk
X-On- Behalf-Of: lcarr
Examples - GET (explain)

status: The status is: Code: 200, Message: 'OK'
<?xml version="1.0" encoding="UTF-8"?>
  <sword:level>1</sword:level>
  <sword:verbose>true</sword:verbose>
  <sword:noOp>true</sword:noOp>
  <workspace>
    <atom:title type="text">Fedora SWORD Workspace</atom:title>
    <collection href="http://glen.dnsalias.org/sword/deposit/collection:open">
      <atom:title type="text">Open Collection</atom:title>
      <accept>text/xml</accept>
      <accept>application/zip</accept>
      <accept>application/x-zip-compressed</accept>
      <accept>application/atom+xml</accept>
      <accept>image/gif</accept>
      <accept>image/jpeg</accept>
      <accept>image/jpg</accept>
      <sword:collectionPolicy>This collection accepts any deposit from anyone</sword:collectionPolicy>
      <dcterms:abstract>This is a collection of objects which can be freely deposited to. This is avialable for the SWORD test project</dcterms:abstract>
      <sword:mediation>true</sword:mediation>
      <sword:treatment>Preservation actions may occur on submitted deposits</sword:treatment>
      <sword:formatNamespace>uri</sword:formatNamespace>
    </collection>
  </workspace>
</service>
Examples - POST (deposit)

POST /burning-collection HTTP/1.1
Host: www.myrepository.ac.uk/sword-app
Content-Type: application/zip
Authorization: Basic ZGFmZnk6c2VjZJldA==
Content-length: nnn
Content-MD5: md5-digest
Content-Disposition: filename=mydeposit.zip
X-On-Behalf-Of: lcarr
X-Format-Namespace: METS
Examples - POST (deposit)

```xml
<entry xmlns="http://www.w3.org/2005/Atom"
    xmlns:sword="http://purl.org/net/sword/"
    xmlns:"http://www.w3.org/2005/Atom"
    xmlns:sword="http://purl.org/net/sword/">
    <title>My Deposit</title>
    <id>info:something:id</id>
    <updated>2007-05-14T14:27:08Z</updated>
    <author><name>mmorrey</name></author>
    <summary type="text">A summary</summary>
    <content type="application/zip"
        src="http://www.myrepository.ac.uk/my_deposit.zip"/>
    <link rel="edit-media"
        href="http://www.myrepository.ac.uk/lcarr/workflow/my_deposit"/>
    <link rel="edit"
        href="http://www.myrepository.ac.uk/lcarr/workflow/my_deposit.atom"/>
    <contributor><name>lcarr</name></contributor>
    <source>
        <generator uri="http://www.myrepository.ac.uk/sword/"
            version="1.0">
            SWORD & My Repository
        </generator>
    </source>
    <sword:treatment>Description</sword:treatment>
    <sword:verboseDescription>Description</sword:verboseDescription>
    <sword:noOp>true</sword:noOp>
    <sword:formatNamespace>http://www.loc.gov/METS_Profile/</sword:formatNamespace>
</entry>
```
The Technical
implementing the profile
Implementation

Repository implementations
  - DSpace
  - EPrints
  - IntraLibrary
  - Fedora

Client implementations
  - Java client library
    - command-line, desktop and web clients
status: Requesting the document from http://cakeordeath.ecs.soton.ac.uk/cgi/servicedocument

<?xml version="1.0" encoding="UTF-8"?>
  <sword:level>1</sword:level>
  <sword:verbose>true</sword:verbose>
  <sword:noop>true</sword:noop>
  <workspace>
    <atom:title type="text">DSpace at My University</atom:title>
    <collection href="http://sword.aber.ac.uk/dspace-sword/deposit/123456789/28">
      <atom:title type="text">Collection with workflow step 1</atom:title>
      <sword:collectionPolicy>NOTE: PLACE YOUR OWN LICENSE HERE This sample license is provided for informational purposes only. NON-EXCLUSIVE DISTRIBUTION LICENSE By signing and submitting this license, you (the author(s) or copyright owner) grants to DSpace University (DSU) the non-exclusive right to reproduce, translate (as defined below), and/or distribute your submission (including..."/></sword:collectionPolicy>
    </collection>
  </workspace>
</service>
User testing

Four case studies, implementations in:

- SPECTRa tool
- arXiv
- White Rose Research Online
- SOURCE project
The future ...

where seamless deposit actually happens?
Is anybody using SWORD?

In addition to the case study implementations:

- Feedforward has already implemented
- ICE project is looking at SWORD
- DSpace and EPrints installations already exist
- Microsoft eChemistry work
- OAI-ORE interest
- more are planned
- NISO activity around deposit – hopefully this will recognise SWORD
SWORD II and beyond

- Small amount of continuation funding
- What for? up for discussion/scoping
  - additional APP support (update/delete)
  - more clients (.net / php etc)
  - more tools (for desktop and web-based deposit)
  - ORE testing (deposit of Resource Map as ATOM document, deposit of Resource Map as XML file)
- ongoing support for code and test installations
Questions?

www.ukoln.ac.uk/repositories/digirep/index/SWORD

Julie Allinson <j.allinson@gmail.com>