Giving Researchers Credit for their Data - Phase 3

Implementation guide for repository managers and publishers on communicating with the *data paper companion* application

Anusha Ranganathan

anusha@digitalnest.co.uk

1. Introduction

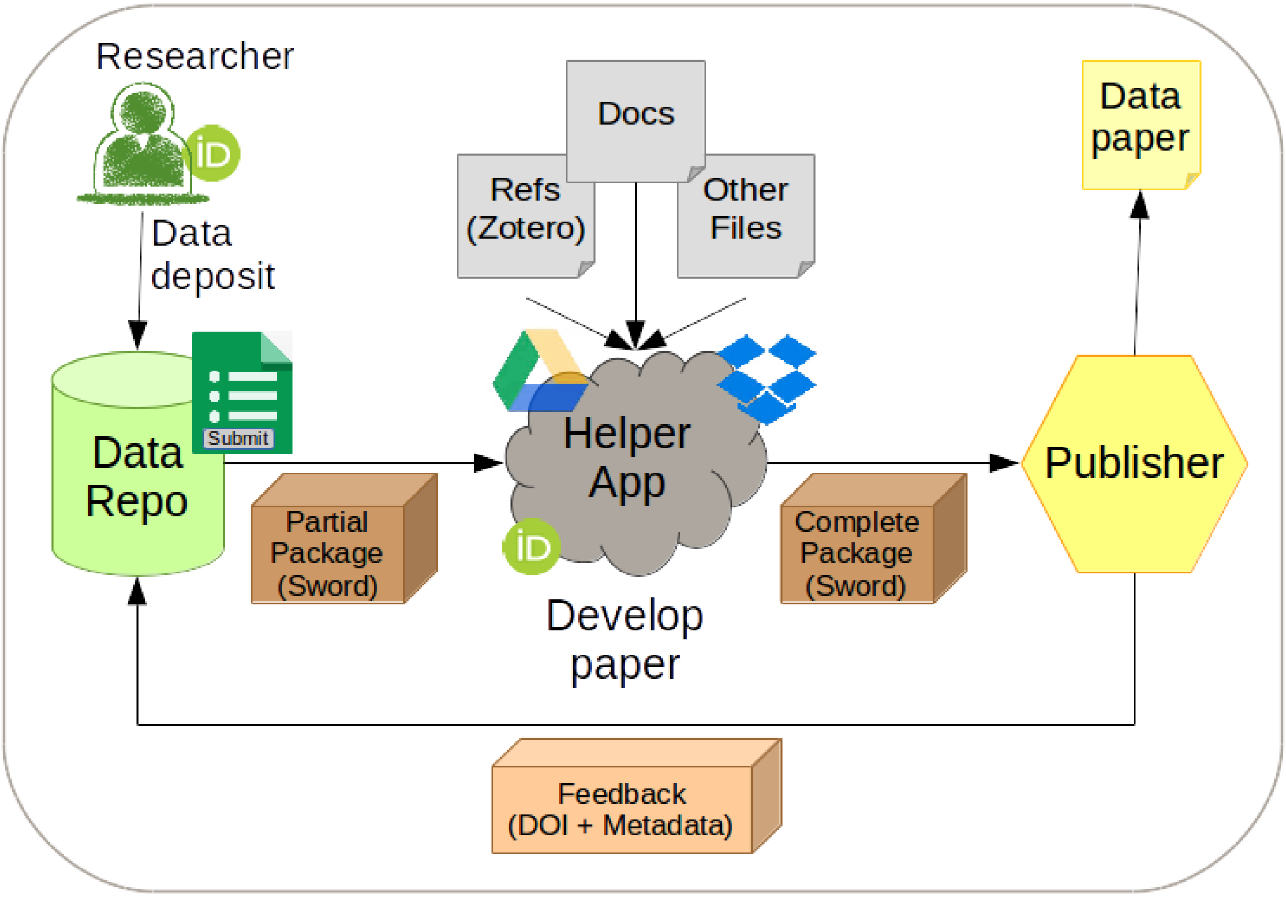

Fig1: Schematic

This is a simple schematic of the process developed where metadata and methods detail are transferred from an institutional repository to a relevant publisher platform for publication as a data paper, collaborating with all of the stakeholders of the project - publishers, data repository managers and software developers.

The above schematic, as outlined in phase 1 and developed in phase 2 of the project identifies the communication between 3 main systems and the researcher. The 3 systems are:

The data repository
The helper app (*data paper companion)*
The publisher

It also hints at protocols that can be used for communication between the systems, namely the SWORD protocol (Simple Web-service Offering Repository Deposit) for transferring data and integration with ORCID for user authentication using OAuth2.

2. SWORD protocol

2.1 SWORD[1]: Current state and overview

SWORD is a lightweight protocol for depositing content from one location to another. It stands for Simple Web-service Offering Repository Deposit and is currently in its second version. The SWORD protocol Is a profile of the Atom Publishing Protocol (known as APP or ATOMPUB) and has been extended from the ATOMPUB format.

The purpose of SWORD is to provide an interoperable interface that allows deposits to be easily made into repositories. Details such as contents of the package being transported, the method of packaging, metadata formats to be used to describe the package and authentication and authorization between the client and server are out of the scope of SWORD.

SWORD supports the idea of collections, whereby SWORD servers can setup multiple collections to serve different needs. For example, collections to support different packaging requirements or collections to support different groups based on authorization rules and user membership or to support different content workflows.

2.2 SWORD usage in the *data paper companion*

The *data paper companion* application has both a SWORD server and a SWORD client implemented within the application in order to be able to receive content from institutional repositories and serve content to publisher submission systems respectively.

At present the *data paper companion* application doesn't group content within it's system into different collections and so the client submitting data packets into the *data paper companion* application needs to submit all data packets into a single default collection. Institutional repositories require a SWORD client to be integrated with their system to be able to communicate with the *data paper companion* and submit data packets

Publisher paper submission systems require a SWORD server to be able to accept data packets sent by the *data paper companion.* Publisher SWORD systems can support the idea of collections if required and the SWORD client in the *data paper companion* will currently submit to the first collection from the list of collections available for submission.

---

[1] http://swordapp.org/.
Presentations: http://swordapp.org/sword-v2/sword-v2-presentations/
Sword profile: http://swordapp.github.com/SWORDv2-Profile/SWORDProfile.html
Code containing client libraries and reference implementations –
http://sourceforge.net/p/sword-app/code/HEAD/tree/, https://github.com/swordapp

2.3 Available SWORD resources

SWORD version 2 specifications are available at
http://swordapp.org/sword-v2/sword-v2-specifications/

The SWORD profile web page has a good overview and explanation of the specifications at http://swordapp.github.io/SWORDv2-Profile/SWORDProfile.html

The SWORD related repositories in github https://github.com/swordapp have a few example client and server implementations in different languages.

- Eprints[2] Dspace[3], Fedora 3[4] and Dataverse contain SWORD implementations (as plugins) to accept deposits.
- SWORD clients are available in Java[5], Ruby[6], PHP[7] and Python[8] .
- Reference SWORD server implementations are also available in Python[9] and Java [10]

Sourceforge also has some of the code used for building repository plugins https://sourceforge.net/p/sword-app/code/HEAD/tree/

---

[2] http://wiki.eprints.org/w/SWORD_2.0
[3] https://wiki.duraspace.org/display/DSDOC3x/SWORDv2+Server
[4] https://wiki.duraspace.org/display/FCSVCS/SWORD-Fedora+1.2
[5] https://github.com/swordapp/JavaClient2.0
[6] https://github.com/swordapp/sword2ruby
[7] https://github.com/swordapp/swordappv2-php-library/
[8] https://github.com/swordapp/python-client-sword2
[9] https://github.com/swordapp/Simple-Sword-Server
[10] https://github.com/swordapp/JavaServer2.0

3. Interactions between the data repository and the *data paper companion*

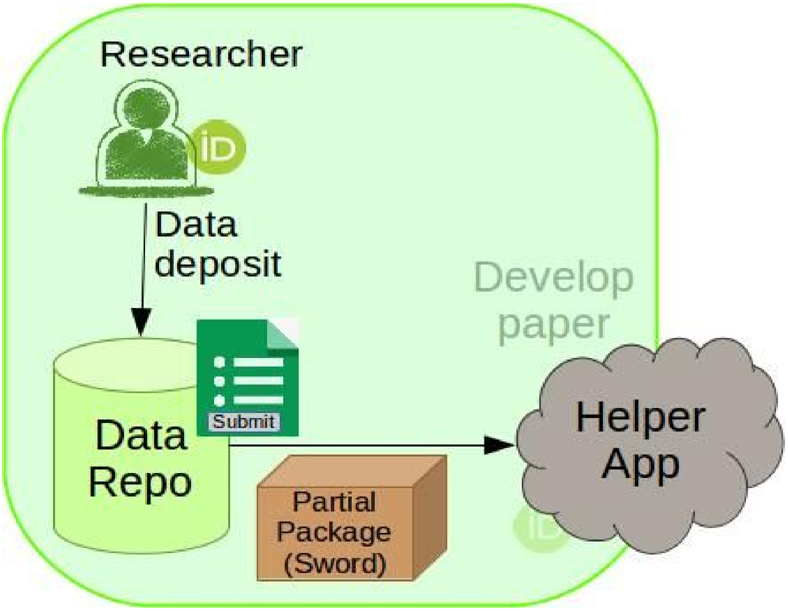

Fig 2: The interaction between the data repository and the helper application

3.1 Data repository requirements for interacting with the *data paper companion*

- DOI / Persistent URI for the dataset
  - Note: only data packages that are openly accessible and not under embargo or dark to be deposited to the *data paper companion*
- Repository user should have a valid ORCID ID used for authentication and included in the metadata
- Metadata in one of two formats - Datacite compatible xml metadata file or metadata in Dublin core xml
  - If a DOI is available, just providing the DOI as metadata would be sufficient
  - The metadata file is to be titled *metadata.xml*
- If other domain or repository specific metadata files are available, add them as additional files
- Package all files including metadata file into a zip file
- Information to be submitted to helper application by means of a SWORD client
  - For this, the repository would need to know the SWORD endpoint of the *data paper companion* (IRI of the SWORD service document)

A reference implementation is available for Hydra users as a Ruby Gem

Note: Edits from the data repository are not supported in this version of the application

3.2 User interactions with the data repository for submitting a data paper to the *data paper companion*

A user will ideally be able to select a data package for which they would like to publish a data paper. Once the package is selected (either from a list of packages or from the landing page of the data package), the user needs to click on a button to submit the package to the *data paper companion*.

Clicking on a link available next to the button, will take them to the *data paper companion* application where they will be able to login through ORCID, using their ORCID credentials (if they aren't already logged in) and view their recent submission, along with past submissions.

3.3 SWORD client interactions with the *data paper companion*

Once the user has clicked on submit, the SWORD client in the data repository will need to perform the following essential interactions with the SWORD server in order to submit a data package (usually as a zip file) to the *data paper companion*

1. Retrieve the service document from the SWORD server

   The client sends a GET request to the IRI of the Service Document (for example: http://datapapercompanion.com/swordv2/service-document/ note: url not active)

   The server responds with a Service Document enumerating the IRIs of a group of collections and the capabilities of those collections supported by the server. In the case of the *data paper companion*, all deposits are made into a single default collection

   An example service document is below

```
<service
xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:sword="http://purl.org/net/sword/terms/"
xmlns:atom="http://www.w3.org/2005/Atom"
xmlns="http://www.w3.org/2007/app">
<sword:version>2.0</sword:version>
<sword:maxUploadSize>1048576</sword:maxUploadSize>
<workspace>
  <atom:title>Data Paper Companion</atom:title>
  <collection href="http://datapapercompanion.com/swordv2/collection-deposit/">
    <atom:title type="text">Data Paper Companion deposit collection</atom:title>
    <accept>application/zip</accept>
    <collectionPolicy>Submission should be related to one data paper and should
```

```
      contain the metadata file and the submitter details</collectionPolicy>
     <sword:mediation>false</sword:mediation>
     <sword:treatment>The zip file will be unpacked and the metadata
      extracted</sword:treatment>
     <sword:acceptPackaging>
      http://purl.org/net/sword/package/SimpleZip</sword:acceptPackaging>
   </collection>
 </workspace>
 </service>
```

2. The client should create a resource (submit the zip file) by POSTing a zip file containing the data packet as the body of an HTTP request to the collection IRI stated in the service document with the Content-Type, Content-Disposition, Content-MD5 and Packaging headers.

   An example collection IRI is http://datapapercompanion.com/swordv2/collection-deposit/.

   An example HTTP request is shown below

   ```
   POST http://datapapercompanion.com/swordv2/collection-deposit/ HTTP/1.1
    Host:  datapapercompanion.com
    Authorization: Basic ZGFmZnk6c2VjZXJldA==
    Content-Type: application/zip
    Content-Length: [content length]
    Content-MD5: [md5-digest]
    Content-Disposition: attachment; filename=[filename]
    Packaging: http://purl.org/net/sword/package/SimpleZip
   ```

   The *data paper companion* will respond with a HTTP 201 created response along with an optional deposit receipt (See appendix 2 – example deposit receipt)

A full SWORD client will have further functionality to support sending edits made to a data package to a SWORD server, but as the application doesn’t yet have support for edits, but just takes submissions, the above functionality is what is essentially required from the SWORD client

4 Interactions between the *data paper companion* and the publisher submission system

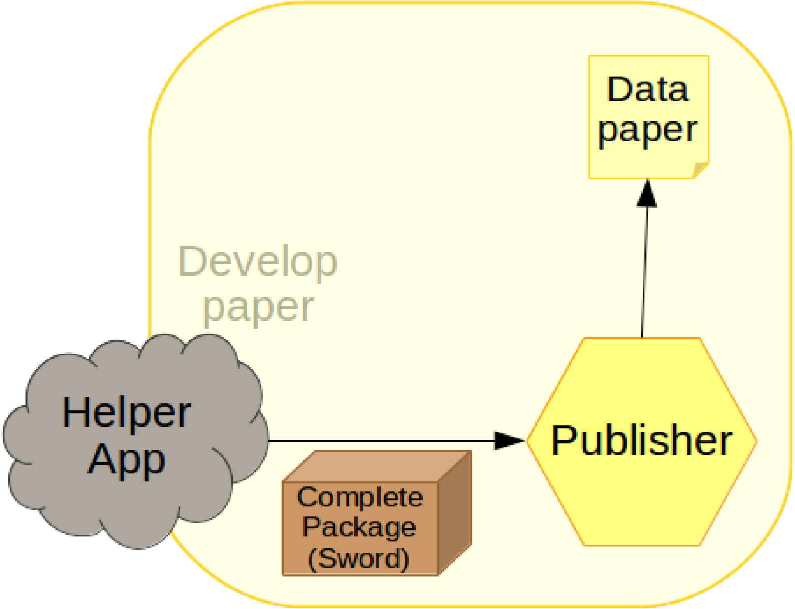

Fig 3: The interaction between the helper application and the publisher

4.1 Information needed from the publisher for each journal by the *data paper companion*

For each journal, the *data paper companion* will register a journal configuration entry. The journal configuration entry contains the following information

- Title of journal
- Website
- Name of publisher
- Contact information for the publisher
- Preferred metadata format – DC / Datacite XML / publisher's format
- Choice of licences for the author to choose from
- Declarations needed by the author
- Links to the data paper template file
- Authorization keys
- SWORD endpoint

4.2 User interactions with the *data paper companion* for submitting a data paper to the publisher

A user will ideally be able to select a data package for which they would like to publish a data paper and go to the landing page of that data paper. From here, the user can

- Select a journal
- Download the pre-filled data paper template
- Upload the data paper and any supplementary files from their local computer or their cloud storage (Google Drive, Dropbox, Onedrive...)
- Choose a license for their work from the list of available licences from the publisher
- Accept the publisher's terms and conditions and sign a declaration
- Review their data paper, files and selections

When the user is ready to submit their data paper to the journal, they will click on the submit button. The *data paper companion* will then package all of the data into a zip file and send it to the journal's submission system using the SWORD protocol.

4.3 Submission package sent from the *data paper companion* to the publisher's system

The SWORD package sent to the publisher's SWORD endpoint from the SWORD client in the *data paper companion* will contain the following files

- The metadata file in the chosen format (Dublin core XML, Datacite XML or publisher's XML format) *metadata.xml.* The metadata will include information about the following
    - Data repository the package originated from
    - User submitting the package
    - The list of all the files in the package and what the file is (metadata files received from the data repository, the data paper, supplementary files uploaded by the user)
- Any additional domain / repository specific metadata files sent by the data repository
- The data paper (the template paper file pre-filled by the application and completed by the user)
- Any supplementary files added by the author

4.4 SWORD server interactions with the *data paper companion*

Once the user has clicked on submit, the SWORD client in the *data paper companion* will need to perform the following essential interactions with the SWORD server in the publisher's system in order to submit a data paper

1. Retrieve the Service Document from the SWORD server

   The client sends a GET request to the IRI of the Service Document (for example: http://publisher.com/swordv2/service-document/)

   The server responds with a Service Document enumerating the IRIs of a group of Collections and the capabilities of those Collections supported by the server.

   The *data paper companion* can authenticate with the publisher SWORD endpoint using API keys or basic-auth if required.

   An example service document is below

```
<service
xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:sword="http://purl.org/net/sword/terms/"
xmlns:atom="http://www.w3.org/2005/Atom"
xmlns="http://www.w3.org/2007/app">
<sword:version>2.0</sword:version>
<sword:maxUploadSize>1048576</sword:maxUploadSize>
<workspace>
  <atom:title>Name of Publisher</atom:title>
  <collection href="http://publisher.com/swordv2/collection1/">
    <atom:title type="text">Publisher collection of a specific type</atom:title>
    <accept>application/zip</accept>
    <sword:mediation>false</sword:mediation>
    <sword:treatment>The zip file will be unpacked and the files
      extracted</sword:treatment>
    <sword:acceptPackaging>
      http://purl.org/net/sword/package/SimpleZip</sword:acceptPackaging>
  </collection>
</workspace>
</service>
```

2. The *data paper companion* will pick the first collection from the list of collections in the service document and submit the data paper by POSTing the zip file containing the data packet as the body of an HTTP request to the collection IRI stated in the

service document (http://publisher.com/swordv2/collection1/), with the Content-Type, Content-Disposition, Content-MD5 and Packaging headers

An example HTTP request is shown below

```
POST http://datapapercompanion.com/swordv2/collection-deposit/ HTTP/1.1
Host:  datapapercompanion.com
Authorization: Basic ZGFmZnk6c2VjZXJldA==
Content-Type: application/zip
Content-Length: [content length]
Content-MD5: [md5-digest]
Content-Disposition: attachment; filename=[filename]
Packaging: http://purl.org/net/sword/package/SimpleZip
```

The publisher SWORD endpoint will need to respond with a HTTP 201 created response along with an optional deposit receipt (See appendix 2 – example deposit receipt). The deposit receipt will need to contain IRIs for editing a resource. Since we are not going to make use of any edit functionality, this can be a dummy URL.

Once the SWORD server receives the packet, the treatment of the packet and the workflow required to integrate the contents of the data packet with the publisher system is left to the local system to implement a solution that meets their requirements.

A full SWORD server implementation will have further functionality to support allowing edits to a packet, but as the application doesn't yet have support for edits, but just takes submissions, the above functionality is what is essentially required from the SWORD server.

Appendix 1

Example service document sent by the *data paper companion* to the repository

```
<service
  xmlns:dcterms="http://purl.org/dc/terms/"
  xmlns:sword="http://purl.org/net/sword/terms/"
  xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns="http://www.w3.org/2007/app">
  <sword:version>2.0</sword:version>
  <sword:maxUploadSize>1048576</sword:maxUploadSize>
  <workspace>
    <atom:title>Data Paper Companion</atom:title>
    <collection href="http://datapapercompanion.com/swordv2/collection-deposit/">
      <atom:title type="text">Data Paper Companion deposit
        collection</atom:title>
      <accept>application/zip</accept>
      <collectionPolicy>Submission should be related to one data paper
        and should contain the metadata file and the submitter
        details</collectionPolicy>
      <sword:mediation>false</sword:mediation>
      <sword:treatment>The zip file will be unpacked and the metadata
      extracted</sword:treatment>
      <sword:acceptPackaging>
        http://purl.org/net/sword/package/SimpleZip</sword:acceptPackaging>
    </collection>
  </workspace>
</service>
```

Appendix 2

Example deposit receipt sent by the *data paper companion* to the repository

```xml
<entry xmlns="http://www.w3.org/2005/Atom"
  xmlns:sword="http://purl.org/net/sword/"
  xmlns:dcterms="http://purl.org/dc/terms/">

  <title>My Deposit</title>
  <id>info:something:1</id>
  <updated>2016-05-01T14:27:08Z</updated>
  <summary type="text">A summary</summary>
  <generator uri="http://datapapercompanion.com//sword-plugin/" version="1.0"/>

  <!-- the item's metadata -->
  <dcterms:abstract>The abstract</dcterms:abstract>
  <dcterms:accessRights>Access Rights</dcterms:accessRights>
  <dcterms:alternative>Alternative Title</dcterms:alternative>
  <dcterms:available>Date Available</dcterms:available>
  <dcterms:bibliographicCitation>Bibliographic Citation</dcterms:bibliographicCitation>
  <dcterms:contributor>Contributor</dcterms:contributor>
  <dcterms:description>Description</dcterms:description>
  <dcterms:hasPart>Has Part</dcterms:hasPart>
  <dcterms:hasVersion>Has Version</dcterms:hasVersion>
  <dcterms:identifier>Identifier</dcterms:identifier>
  <dcterms:isPartOf>Is Part Of</dcterms:isPartOf>
  <dcterms:publisher>Publisher</dcterms:publisher>
  <dcterms:references>References</dcterms:references>
  <dcterms:rightsHolder>Rights Holder</dcterms:rightsHolder>
  <dcterms:source>Source</dcterms:source>
  <dcterms:title>Title</dcterms:title>
  <dcterms:type>Type</dcterms:type>

  <sword:treatment>Unpacked. contents with additional metadata
    from Datacite</sword:treatment>

  <link rel="edit-media" href="http://datapapercompanion.com/col1/mydeposit"/>
  <link rel="edit" href="http://datapapercompanion.com/col1/mydeposit.atom" />
  <sword:packaging>http://purl.org/net/sword/package/SimpleZip</sword:packaging>

  <link rel="http://purl.org/net/sword/terms/originalDeposit" type="application/zip"
    href="http://datapapercompanion.com/col1/mydeposit/package.zip"/>
  <link rel="http://purl.org/net/sword/terms/derivedResource" type="application/pdf"
    href="http://datapapercompanion.com/col1/mydeposit/file1.pdf"/>
  <link rel="http://purl.org/net/sword/terms/derivedResource" type="application/pdf"
    href="http://datapapercompanion.com/col1/mydeposit/file2.pdf"/>
</entry>
```