



Issues surrounding syndicated feed deposit into institutional repositories

Authors: Gareth Waller

Date: October 2009

Version: Final

Status: For publication

Authorised: Jackie Carter/Peter Burnhill

Version Changes to this File

REVISION	DATE	CHANGE	DETAILS
0.1	Nov 09	Revised in line with comments from PB	Agreed publication

Related Documents

RELATED DOCUMENT	DETAILS

Table of Contents

OUTLINE	3
REFERENCES	3
BACKGROUND	3
SYNDICATED FEED FORMATS	4
WHAT IS A FEED?	4
HOW DO FEEDS WORK?	5
REPOSITORY IMPLEMENTATION	6
ITEM IDENTIFICATION	6
ITEM UPDATES	6
ITEM DELETIONS	7
MISSING ITEMS	7
POLLING PERIOD	7
FEED FORMATS	7
METADATA FORMATS	7
REPOSITORY REQUIRED METADATA PROFILE	7
LICENSING CONTENT	8
LINKS OR RESOURCE DOWNLOAD	8
CONCLUSION	9

Outline

Repositories offer various ways of depositing resources. This paper examines the issues surrounding the potential offered by a syndicating feed standard such as Really Simple Syndication (RSS) and the ATOM protocol. The distinction is made between notification of metadata, for the purpose of registration and supply of metadata (to support search and subsequent onward linking to the object described being hosted elsewhere) and deposit of the object with metadata (to support its release for others to use).

References

1. <http://swordapp.org/>
2. <http://www.dspace.org/>
3. <http://www.fedora-commons.org/>
4. <http://www.eprints.org/>
5. <http://www.intrallect.com/>
6. <http://creativecommons.org/>
7. <http://www.openarchives.org/OAI/openarchivesprotocol.html>
8. <http://www.loc.gov/standards/sru/>
9. <http://en.wikipedia.org/wiki/RSS>
10. [http://en.wikipedia.org/wiki/Atom_\(standard\)](http://en.wikipedia.org/wiki/Atom_(standard))
11. <http://www.w3.org/TR/xhtml-rdfa-primer/>

Background

Institutional repository software is designed as a device storing and finding user submitted resources and their associated metadata. There are various offerings both commercial and open source and some common examples are listed below:

- DSpace [2]
- Fedora [3]
- EPrints [4]
- intraLibrary [5]

In terms of allowing a user to deposit content, there are 3 common approaches:

1. Deposit of a single physical resource via a web interface
2. 'Deposit' of a web link pointing to a physical resource somewhere on the internet, again via a web interface – (referred to as notification)
3. A machine to machine (M2M) deposit interface e.g. using a standard such as SWORD [1]

For the first with a human makes a choice on which resource to deposit, where this resource should be stored (i.e. in the repository or somewhere else on the internet) and usually how the resource should be licensed e.g. under a Creative Commons licence

[6]. For the second, again a human makes a decision to deposit a web link, although this 'deposit' of metadata does not imply that the repository is used in decision making about where the content is stored nor does the repository assist in the assignment of licence and subsequent release of the content for others to use. The term notification is a better description of what is taking place, in order to register the existence of a resource through appropriate metadata for identification, description and availability – including the URL. For the third, the actions are not taken by humans.

Many repositories also offer numerous ways of allowing users to find resources, namely:

1. Direct search via a repository web interface
2. Metadata harvesting using a harvesting standard such as OAI-PMH [7]
3. Web service interface offering search based on a specific search term such as SRU [8]

It is also now a common feature of repositories to offer syndicated feeds informing end users of additions to the repository e.g. a new journal article.

As more and more institutions decide to use institutional repositories to store digital media, an interesting question can be posed – is it possible for one repository to “subscribe” to another repository, learn about new content and offer this content to its own end users? At first glance using syndicated feeds to achieve this would be an obvious choice and the remainder of this paper will discuss the possible issues to consider with this approach.

Syndicated Feed Formats

There are three main formats to consider when discussing syndicated feeds, namely:

RSS 1.0 [9]
RSS 2.0 [9]
Atom 1.0 [10]

These formats differ in their syntax and as such for a repository to support deposit of material via subscribing to a syndicated feed, a choice must be made as to which format to support (if not all).

What is a feed?

In its simplest term a feed is an XML document (text document). An example can be seen below which was taken from the BBC News RSS feed:

```
<?xml version="1.0" encoding="ISO-8859-1" ?><?xml-stylesheet title="XSL_formatting" type="text/xsl" href="/shared/bsp/xsl/rss/nolsol.xsl"?>
<rss version="2.0" xmlns:media="http://search.yahoo.com/mrss">
  <channel>
    <title>BBC News | News Front Page | UK Edition</title>
    <link>http://news.bbc.co.uk/go/rss/-/1/hi/default.stm</link>
```

Issues Surrounding Syndicated Feed Deposit into Institutional Repositories

```
<description>Visit BBC News for up-to-the-minute news, breaking news,
video, audio and feature stories. BBC News provides trusted World and UK news as well
as local and regional perspectives. Also entertainment, business, science, technology
and health news.</description>
<language>en-gb</language>
<lastBuildDate>Tue, 13 Oct 2009 10:34:40 GMT</lastBuildDate>
<copyright>Copyright: (C) British Broadcasting Corporation, see
http://news.bbc.co.uk/1/hi/help/rss/4498287.stm for terms and conditions of
reuse</copyright>
<docs>http://www.bbc.co.uk/syndication/</docs>
<ttml>15</ttml>
<image>
  <title>BBC News</title>
  <url>http://news.bbc.co.uk/nol/shared/img/bbc_news_120x60.gif</url>
  <link>http://news.bbc.co.uk/go/rss/-/1/hi/default.stm</link>
</image>
<item>
  <title>Pay up or quit, Cameron warns MPs</title>
  <description>David Cameron says that if MPs are asked to pay
back expenses and do not do so, they will not be able to stand as Tory
candidates.</description>
  <link>http://news.bbc.co.uk/go/rss/-
/1/hi/uk_politics/8304125.stm</link>
  <guid
isPermaLink="false">http://news.bbc.co.uk/1/hi/uk_politics/8304125.stm</guid>
  <pubDate>Tue, 13 Oct 2009 10:25:39 GMT</pubDate>
  <category>Politics</category>
  <media:thumbnail width="66" height="49"
url="http://newsimg.bbc.co.uk/media/images/46517000/jpg/_46517272_cameronnew_getty.jpg
"/>
</item>

</channel>
</rss>
```

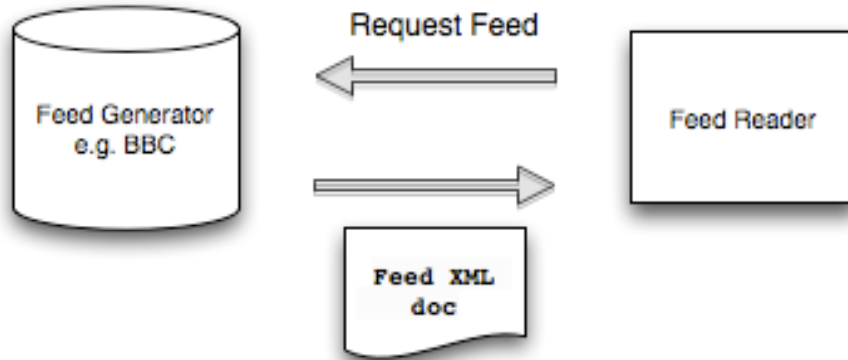
Feeds simply describe updates to a service e.g. website, blog etc and users who are interested in that service can “subscribe to” the feed. Subscribing to a feed allows a user to be informed of updates without having to go to the actual service e.g. a user subscribing to the BBC News RSS feed could be informed of new BBC headlines without having to go to the BBC website and searching for news.

How do feeds work?

A user subscribes to a feed they are interested in via a “feed reader”. A “feed reader” (sometimes referred to as a “feed aggregator”) is a piece of software which manages users subscribed feeds and automatically checks for new updates from feeds on behalf of the user e.g. Google Reader, NetNewsWire. A feed reader can keep track of which items within a feed a user has read and which items are considered “new” and unread.

A feed reader can automatically request the feed document on a regular basis e.g. every hour, check for new items and inform the user accordingly.

The actual feed (remember simply an XML document) is generated by a server process on the remote server e.g. the BBC News service. Normally the feed contains a finite number of items and does not contain every update since the service started. For example the BBC News feed does not include every headline since the BBC News service started, it merely contains the most recent headline e.g. the 40 most recent headlines. This is an important point to note when considering using a feed for deposit into a repository, which will be discussed later.



Repository Implementation

In order for a repository to support automatic ‘deposit’ of resource by subscribing to a remote syndicated feed, the repository must act as a feed reader and manage feed subscriptions on behalf of the repository administrator. The implementation must find a mechanism of determining which items in a feed have already been ‘read’, i.e. deposited into the repository, and also which items have been updated or deleted (so that the repository remains in sync with the feed).

The following issues should be considered if attempting to implement a feed reader solution within an institutional repository:

Item identification

How can a unique identifier be assigned to an item within a feed and how can this be linked with the corresponding resource in the repository? The feed reader component of the repository must know which items have been previously processed so that duplicate submissions are prevented.

Item updates

How can the feed indicate if an item has been updated? Does there need to be an agreed string term within the feed that a repository implementer could search for e.g. “Jorum:update”? Could resource check summing or signatures provide a solution?

Drawbacks – how can a term be agreed with potentially numerous repository implementers. Check summing and resource signatures require a single physical file, what if the item in the feed points to an entire website with numerous pages, links, images etc.

Item deletions

How can a feed indicate if an item has been deleted? Does simple omission from a feed indicate an items deletion?

Drawbacks – simply omitting the item from a feed cannot be used to mark a deletion as a feed should not contain the entire contents of a repository. It should only contain a finite number of updates.

Missing items

If a feed contains a finite number of items, there is the possibility that a subsequent request of the feed will not contain every update since the last time the feed was requested. E.g. a feed reader could read the BBC news headlines on a Monday, and then again on Tuesday but it is not guaranteed to receive every news headline that happened between Monday and Tuesday. The BBC news feed generator may be configured to only send the 50 most recent headlines, if there were 60 headlines between Monday and Tuesday, 10 would be “lost”. This would be a problem for using a feed for “harvesting” content from a remote repository as the feed would never again contain the “lost” updates and as such the resource would never appear in the subscribing repository.

Polling period

How often should the feed document be requested from the feed generator i.e. how frequently should the subscribing repository check for updates? Depending on the frequency of additions/updates/deletions to the remote service this may need to be very regular in an attempt to limit “missing items” (see earlier).

Feed formats

Which feed formats should be supported?

Metadata formats

Which metadata formats should be supported within feeds e.g. Dublin Core, LOM etc. It is highly likely that different feed generators may supply different metadata formats.

Repository Required Metadata Profile

It is common for a repository to implement a minimum metadata a profile which each resource submitted to the repository should meet e.g. must have title, description,

keywords, licence etc. How can this be managed when a user is not submitting the item and has the opportunity to add metadata if necessary – harvesting is an automatic process.

Licensing Content

How can items within a feed indicate they are bound by a specific license? Is this using a standard such as RDFa[11] or non-standard strings e.g. “Licensed under Creative Commons 2.0 Attribution”. Does the repository support the license specified in the metadata? More fundamentally, there is presumption that deposit does not need to be followed up by human intervention to assign an appropriate licence to give authority for the subsequent release of the content for others to use.

Links or resource download

The feed will contain links to resources, should the repository simply store the link or should the link be “followed” and the content downloaded and stored in the repository? This is of course first and foremost a policy question but it has technical and operational implications.

Conclusion

Support for deposit of resources via subscription to repository feeds raises a complex set of issues which require prior attention. It is worthwhile re-thinking what purposes are to be served by use of feed technology. This technology is geared to notification, and its extension for use of notification of structured metadata (including the URL pointer to the remote location of the described resource) may have value. However to use this technology to harvest resources from a remote repository would be to go beyond what feed technology was designed for.

Feed technology provides incremental updates about a service not a complete catalogue of the contents of a service. If the intention is to replicate the contents of one repository in another, alternative technologies such as OAI-PMH would be a much more suitable alternative. OAI-PMH allows harvesting of all metadata records from a repository and support facilities such as resumption tokens but of course must be supported by the remote service.

Also technologies such as SRU allow remote repository searching based on keyword thereby negating the need for the contents of one repository to be mirrored in another – the remote repository could simply be searched on demand. This of course has its own drawbacks e.g. remote repository may not be responding when a search happens due to load, service outage etc.

It is doubtful that a single repository solution could be used to support any feed from any source given the issues outlined earlier, a dialogue between feed providers would have to be established and the issues worked through.

It is also worth noting that to the authors knowledge, no repository platform supports deposit of resources via subscription to a syndicated feed (that includes both open source and commercial platforms). This is therefore a topic for R&D not for immediate service deployment.