Intrallect intraLibrary 3.7: Integration Guide - IntraLibrary Connect

Revision: 6
Created: 1st August 2005
Last Revised: 28th August 2012
Contact: support@intrallect.com
Company: Intrallect Ltd
Product: intraLibrary, Digital Object Repository, intraLibrary Connect
Copyright: Copyright Intrallect Ltd 2003-2012. All rights reserved.

This document is made available to support Intrallect's customers and users of Intrallect's software. The text of these documents and the design of the intraLibrary software are both the intellectual property of Intrallect Ltd. Intrallect does not provide this document for any other purpose, and offer no warranty nor accept any liability for its use in any other context.

Table Of Contents

1. Introduction

This integration guide explains how to connect intraLibrary with other applications and services. This is achieved through a series of web services collectively described as intraLibrary Connect. IntraLibrary Connect is not an additional product, it is a standard part of intraLibrary. These web services do not need to be "switched on" – they are always available.

IntraLibrary Connect services are divided into four groups as shown in the diagram below. This manual is divided into the same four groups.

These four groups are:
- **Discover:** The ability to use a web service for search and discovery of resources in intraLibrary. This web service can be used within authoring tools, portals, learning environments, web sites, etc.
- **Gather:** The ability to extract metadata from intraLibrary in bulk and incrementally over time as it is updated. This can be useful for creating aggregated metadata catalogues or for integration with library systems.
- **Inform:** The ability to get information "pushed" from intraLibrary in the form of RSS news feeds and Apple podcasts.
- **Store:** The ability to deposit resources into intraLibrary individually or in bulk from desktop clients or from other applications such as web sites, authoring tools, or programmatically through scripts.
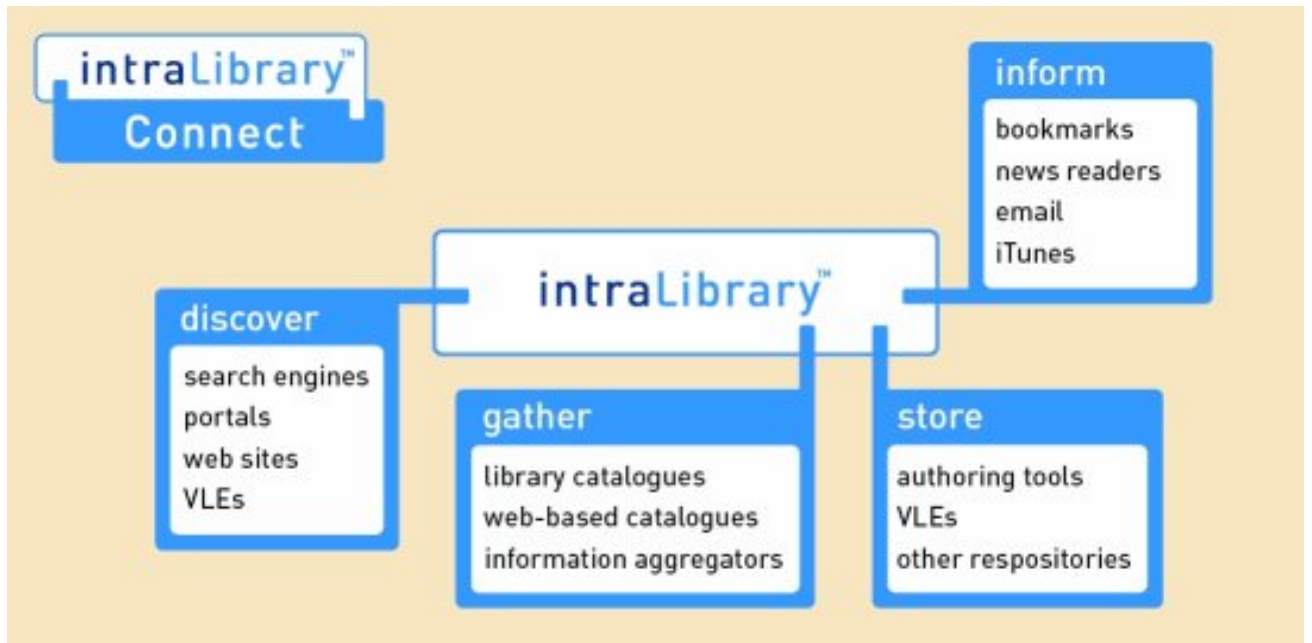
The intraLibrary Ecosystem, shown in the diagram below, is the infrastructure that can be achieved when intraLibrary is integrated with other applications through the intraLibrary Connect web services and supported by the many other services that Intrallect can provide.
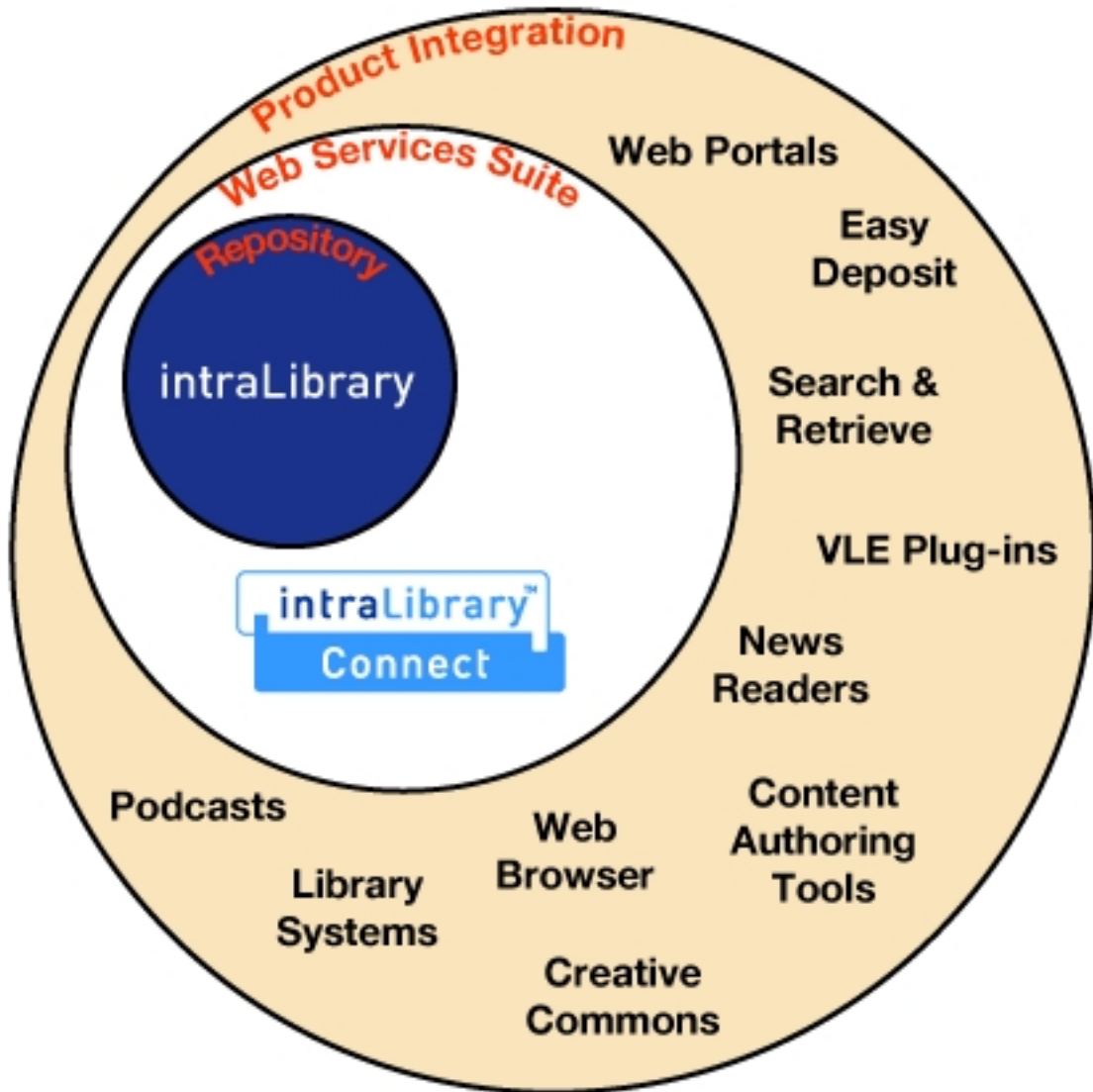
2. Discover

The "discover" web service allows remote users to search for and discover resources held in intraLibrary.

2.1. Benefit to users

The discover component of intraLibrary Connect enables users to discover and retrieve digital resources from intraLibrary without having to log into the intraLibrary interface. There are many ways in which the discover component can be used, such as:

- Open access search portals
- E-learning tools including:
- Web sites such as project, departmental, library or institutional web sites
- Using the web browser search box

The next part of this section will describe the web services on which the discover component is based. This is followed by examples of the use of the discover component.

2.2. Discover Web Service

The discover web service is based on SRU/SRW. SRU means Search and Retrieve by URL and is the REST-based version of the web service while SRW is Search and Retrieve Web Service which is the SOAP version of the same service. SRU/SRW is an international standard managed by the US Library of Congress which now recommends the use of SRU over SRW.

SRU and SRW both operate in a similar way. A query is constructed. The query can be general or specific, for example specific metadata fields can be searched. The metadata fields that can be queried are defined by the "context set". IntraLibrary supports Unqualified Dublin Core and IEEE Learning Object Metadata as context sets – more detail later. The query is submitted to intraLibrary (the submission method is where SRU differs from SRW). The metadata records that match the query are returned. The schema of returned metadata can be specified – Dublin Core or IEEE LOM. The metadata format returned need not be the same as the context set in the query. IntraLibrary has a number of metadata extensions (see below for details) and these will be returned in full if, and only if, the IEEE LOM format is selected for returned records. A style sheet (XSL) may be specified in the query to be applied to the returned results.

2.3. Opening Collections for External Searching

Any intraLibrary repository may contain multiple collections of digital objects. The collections in an intraLibrary repository are closed by default, which means resources in these collections are not visible to the outside world. An administrator-class user of

intraLibrary can open the collection for searching by changing a single property of the collection using a checkbox in the collection properties.

Allow published
content in this
collection to be
searched by
external systems

2.4. Entry Point

The entry point for searching intraLibrary using SRU is the root URL for the intraLibrary installation, with "/IntraLibrary-SRU" appended to it:

http://<intralibrary-root>/IntraLibrary-SRU

2.5. SRU - IntraLibrary-SRU

SRU queries are submitted through http and take the form:

<target repository service>?<query>

where <target repository service> is the URL for the SRU service of your repository. This will be the Application URL of your repository (set in the admin section of intraLibrary in the System tab, System Options section) followed by IntraLibrary-SRU. So if the Application URL of your repository is:

http://demonstrator.intralibrary.com/

then the target repository service is

> http://demonstrator.intralibrary.com/IntraLibrary-SRU

Note that if your Application URL includes a port number that must also be included, for example:

> http://repository.intralibrary.com:8080/IntraLibrary-SRU.

The <query> part contains a query constructed using the Contextual Query Language (CQL). An example of a CQL query is:

> query=intrallect&maximumRecords=4&operation=searchRetrieve&
>   version=1.1&recordSchema=lom

See the section on Queries below for more detail.

So a complete SRU query might look like:

> http://demonstrator.intralibrary.com/IntraLibrary-SRU?query=intrallect&
>   maximumRecords=4&operation=searchRetrieve&version=1.1&
>   recordSchema=lom

If you put this into the address bar of a browser you will have returned to you the XML metadata records that match this query – at least those records which are in collections for which access is permitted (see Access Control below for more details). This ability to put SRU queries in the address bar of a browser makes it very easy to test queries without developing any code.

Sending a query-less request, that is of the form

http://demonstrator.intralibrary.com/IntraLibrary-SRU

will return an XML document explaining the service capabilities of the web service including metadata formats.

2.6. Queries

SRU/SRW uses the Contextual Query Language (CQL) to create queries. Some examples of simple queries are shown below.

- fish
- dc.title=metadata
- dc.creator="john smith"
- dc.keyword=snow
- dc.subject="Natural history of organisms"

There are a number of other ways to qualify the search, including:

Mandatory – these must always be included for a query to be valid

- operation=searchRetrieve
- version=1.1

Optional

- maximumRecords=N
- recordSchema=lom
- startRecord=N
- recordPacking=xml
- stylesheet=URL

IntraLibrary supports no other standard SRU parameters.

2.7. Metadata Context Sets

Specific metadata fields from any of the supported context sets can be used in queries. The context sets are:

- Dublin Core
- Record Metadata
- IEEE LOM
- Intrallect

A complete list of specific fields and the formats to use when querying them is provided in the appendix at the end of the Discover section.

To search using a specific context the usual search query format, for example:

query=fish

is modified to include the context:

query=dc.title=fish

Context searches can be combined using logical *and* and *or* and by using brackets. Examples are shown in the section Defining Collections to Search.

2.8. Defining Collections to Search

From the Record Metadata Context Set intraLibrary supports two indices

Index Name	Description
rec.collectionName	Name of a collection which contains the record. This is set by editing the name of a collection in the "admin tools" interface of intraLibrary.
rec.collectionIdentifier	Identifier for a collection which contains the record. This is a property of a collection, which can be set by editing the properties of a collection in the "admin tools" interface of

	intraLibrary.
rec.userName	Username is the identifier for a user found in the username column on the "admin tools/users" section.

The index rec.collectionName, rec.collectionIdentifier or rec.userName can be used as part of a query to specify particular collections or users. For example a query can:

- constrain a results set to objects in one collection only
  - dc.title = "cat" and rec.collectionName = "e-Prints"
- constrain results to a single user
  - rec.userName = ID12345
- return the entire contents of a collection
  - rec.collectionIdentifier = "uk.ed.epr1"
  obtain results from more than one collection
  - dc.title = "cat" and (rec.collectionName = "images" or rec.collectionName = "learning objects")

2.9. Passing Authentication Tokens

Basic security can be achieved by using authentication tokens as an additional index. Athentication tokens should be in the form:

    x-info-2-auth1.0-authenticationToken=string

- "string" is a string which has been set in intraLibrary to act as an authentication token for one or more collections. The value of "string" is only obtainable from the owner of the repository. Only one authentication token can be included in each query.

See later for more on access control, including examples of the use of authentication tokens.

2.10. Structure of the returned XML

When a query is submitted to the SRU service a response is provided in the form of an XML file.

If no style sheet has been applied the XML returned will have the form:

Returned XML	Description
<SRW:searchRetrieveResponse>	
<SRW:version>1.1 </SRW:version>	
<SRW:numberOfRecords>20 </SRW:numberOfRecords>	Returns total number of records discovered. Note that this may not be the number of records returned as this could be limited by the number of records maximumRecords=N setting
<SRW:records>	Contains all the returned records
<SRW:record>… </SRW:record>	Contains each returned record. See below for details.
</SRW:records>	
<SRW:echoedSearchRetrieveRequest>	
<SRW:version>1.1 </SRW:version>	
<SRW:query>health </SRW:query>	Echo of original query
<SRW:startRecord>1 </SRW:startRecord>	Echo of Start record
<SRW:maximumRecords>4 </SRW:maximumRecords>	Echo of maximumRecords
<SRW:recordPacking>XML </SRW:recordPacking>	Echo of recordPacking
<SRW:recordSchema>lom </SRW:recordSchema>	Echo of recordSchema
</SRW:echoedSearchRetrieveRequest>	
</SRW:searchRetrieveResponse>	

Each of the metadata records returned will have the form shown below. Note that all fields will not always be available as some depend on the context.

Returned XML	Description
<SRW:record>	
<SRW:recordSchema>lom	

</SRW:recordSchema>	
<SRW:recordPacking>XML </SRW:recordPacking>	
<SRW:recordData>… </SRW:recordData>	Contains all available metadata in IEEE LOM format, including all extensions to IEEE LOM supported by Intrallect. Metadata entries are supplied only when they are present.
<SRW:recordPosition>1 </SRW:recordPosition>	recordPosition is the number of this record, starting at 1, relative to the total number of records. This may be used, for example, to specify the startRecord in a subsequent search.
<SRW:extraRecordData>	Contains the extra information that is Intrallect specific.
<package:packagePreviewLocator> … </package:packagePreviewLocator>	Contains a URL which can be used to preview content. If the resource is a single file this will deliver the file. If the resource is a package it will deliver the package in the Intrallect preview environment. If the resource is a web resource it will redirect to the URL of the web resource.
<package:packageType>imscp </package:packageType>	Defines the type of content package
<package:packageTypeVersion>1.1.2 </package:packageTypeVersion>	Defines the version of content package type
<package:packageDownloadLocator> … </package:packageDownloadLocator>	Contains a URL which can be used to download content packages containing both resources and their metadata. Note that if a resource is a "web resource" no package download URL will be present.
<package:packageManifestLocator> … </package:packageManifestLocator>	Contains a URL to the manifest file for the content package.
<package:packageResourceId> ... </package:packageResourceId>	Contains a number which is a unique ID for the resource.
<record:record>	

`<record:lastModified>` 2009-09-07T13:03:10 `</record:lastModified>`	The date the metadata record was last modified.
`<record:created>` 2007-07-29T21:12:56 `</record:created>`	The date the metadata record was created.
`</record:record>`	
`<review:meanStarRating>`4 `</review:meanStarRating>`	Mean star rating for this resource
`<review:numberOfComments>`1 `</review:numberOfComments>`	Number of comments on this resource. Note that each actual comment is included in the LOM part of the record.
`<review:numberOfStarRatings>`0 `</review:numberOfStarRatings>`	Number of star ratings for this resource. Note that the actual star rating for each item is included in the LOM part of the record.
`<review:numberOfStarRatings>`0 `</review:numberOfStarRatings>`	Number of star ratings for this resource. Note that the actual star rating for each item is included in the LOM part of the record.
`<intralibrary:type>` ... `</intralibrary:type>`	The type of file as defined in intraLibrary. Possible values are: *scorm1.2; scorm2004; singleFile; singleFilePackage; imscp; web; physical; kaltura.*
`<intralibrary:thumbnailLocation>` ... `</intralibrary:thumbnailLocation>`	A URL that will deliver a thumbnail image. This is only available for currently supported thumbnail files (jpeg, png, gif and Kaltura files).
`<rights:rights>`	
`<rights:rightsLink>`	
`<rights:rightsURL>` … `</rights:rightsURL>`	A URL to a statement of rights or licence conditions.
`<rights:rightsIcon>` … `</rights:rightsIcon>`	A URL link to an image of a rights icon.
`</rights:rightsLink>`	
`</rights:rights>`	
`</SRW:extraRecordData>`	

2.11. Modifying the Metadata Returned

IntraLibrary can be configured to overwrite parts of the metadata returned under an SRU search. In the "Metadata Options" section of the "system" page there is an option to select a metadata template to apply to metadata returned by a search. The values and macros in the selected template will overwrite the values stored in the relevant fields of each record returned in a search operation.

Possible uses for the feature include:
* consistently exposing useful properties, such as PublicURLs in specific metadata fields
* blanking the contents of certain fields for privacy or security reasons
* ensuring consistency in important statements, such as those describing copyright or usage policy

2.12. Access Control

Within intraLibrary internal access control is managed through collections and you also manage external access through collections. It is possible to determine three levels of access:

* Open: An administrator can make a collection accessible externally by checking "Allow published content in this collection to be searched by external systems" in the "external access" section of the Collections information. (See Administration Guide for more details).
* Closed: If the "Allow published content in this collection to be searched by external systems" is not checked then no external system will be able to discover resources in that collection. If a resource is in more than one collection and any of its collections are open while the others are closed then the resource will be externally accessible through the open collection.
* Authenticated: It is possible to make a collection open but require an authentication token to be provided to obtain access. This allows external access but only to those who know the token for that collection. Authentication tokens can be set and modified by the intraLibrary administrator.

It is important when using SRU searches with collection names or IDs or authentication tokens to understand how these work.

- Collection names or IDs should be used when a search is to be limited to only those collections. These are EXCLUSIVE searches.
- Authentication tokens are used to protect collections from open access. A query that does not include an authentication token will not discover resources within the authentication-token-protected collection. However, it should be noted that a search using an authentication token will not only discover resources in the authentication-token-protected collection but also any open collections. Using authentication tokens provides an INCLUSIVE search.
- If you want to protect a collection from open access but also to search exclusively within that collection then it is necessary to use BOTH Collection names (or IDs) AND authentication tokens.

These options are summarised in the table below:

Collection is ...	Not searchable	Not searchable and has authentication token	Searchable but collection not specified	Searchable and collection specified	Searchable and authentication token provided
Simple query	Hidden	Hidden	Shown	Shown	Hidden
Query with authentication token	Hidden	Hidden	Shown	Shown	Shown
Query with collection named or IDed	Hidden	Hidden	Hidden	Shown	Hidden
Query with collection named or IDed and authentication token	Hidden	Hidden	Hidden	Shown	Shown

In the bottom right box only resources in the named/id-ed and authenticated collection are returned.

2.13. Examples

Examples of a range of SRU queries are provided below. In the first example the target URL is included but in subsequent examples this is omitted to avoid unecessary repetition.

General	
Purpose	Query
Get the full LOM metadata record for a single resource and only that resource	http://repository-intralibrary.leedsmet.ac.uk/IntraLibrary-SRU ?query=dc.identifier=oai:com.intralibrary.leedsmet:430& maximumRecords=4&operation=searchRetrieve& version=1.1&recordSchema=lom
Search for all metadata records and return the results in LOM format – this will include Intrallect extensions so that virtually all metadata is available.	/IntraLibrary-SRU?query=cql.allRecords=1& maximumRecords=4&operation=searchRetrieve& version=1.1&recordSchema=lom

Collection-related	
Purpose	Query
Search for "test" in collection 1	/IntraLibrary-SRU?operation=searchRetrieve&version=1.1& query="test" AND rec.collectionName="collection1"
Search for "test" in collection 1 or collection2	/IntraLibrary-SRU?operation=searchRetrieve&version=1.1& query="test" AND (rec.collectionName="collection1" OR rec.collectionName="collection2")
Search for "test" in both collection 1 and collection2	/IntraLibrary-SRU?operation=searchRetrieve&version=1.1& query="test" AND (rec.collectionName="collection1" AND rec.collectionName="collection2")

Authentication token related	
Purpose	Query
To search for resources, including those in an authentication token	/IntraLibrary-SRU?query=EdReNe& x-info-2-auth1.0-authenticationToken=1234& maximumRecords=4& operation=searchRetrieve&version=1.1&

protected collection. Note that this is also discovers resources that are in collections which have no authentication tokens and are open for searching.	recordSchema=lom
To search for resources in a specific collection that is authentication token protected collection.	/IntraLibrary-SRU? query=intrallect%20 AND%20 rec.collectionName=%22Research%22& x-info-2-auth1.0-authenticationToken=1234& maximumRecords=4&operation=searchRetrieve& version=1.1&recordSchema=lom

2.14. SRW

SRW is the SOAP-based version of the Search and Retrieve Web service. As SRW is no longer recommended by the standard maintainer this section has been removed from this Guide. For advice on using SRW please contact Intrallect directly.

2.15. Tools using SRU

A simple SRU client using only HTML and Javascript is provided by Intrallect and is freely available for anyone to use and adapt.

Java implementations of an SRU Blackboard Powerlink and SRU Blackboard Building Block are also provided by Intrallect under an open source licence.

A PHP implementation of a Moodle plugin is also provided by Intrallect under an open source licence.

In addition a PHP implementation of an open access portal based on SRU has been created by IRISS (The Institute for Research and Innovation in Social Services) and is available under an open source licence.

All of the tools are available for download through the downloads section of the Intrallect

web site.

3. Discovery: Appendices

In this section all the appendices relating to the Discovery web services are collected together.

3.1. Dublin Core Context Set

The following Dublin Core fields can be searched using CQL in intraLibrary. The IEEE LOM fields that these Dublin Core fields map from are also indicated:

Dublin Core Context	IEEE LOM fields used
dc.title	1.2 Title
dc.creator	2.3.2 Contributor (when 2.3.1 Role is author)
dc.subject	1.6 Keyword (all entries) and 9.2 Classification location (taxonpath) (only when "9.1 Purpose" is 'idea' or 'discipline')
dc.description	1.5 Description
dc.publisher	2.3.2 Contributor (when 2.3.1 Role is publisher)
dc.contributor	2.3.2 Contributor (in all roles including Metadata contributor)
dc.date	2.3.3 Date of contribution (when 2.3.1 Role is author or publisher)
dc.type	5.2 Type of resource
dc.format	4.1 Technical format
dc.identifier	4.3 Location of resource
dc.source	No mapping
dc.language	1.4 Language of resource
dc.relation	7.2.2 Related resource description
dc.coverage	1.7 Geographic / Time Period Coverage
dc.rights	6.3 Statement of Copyright and Restrictions

3.2. Record Metadata Context Set

Two fields from the Record Metadata Context set are also supported by intraLibrary

rec.collectionName
rec.collectionIdentifier
rec.userName

3.3. IEEE LOM Context Set

The following LOM fields can be searched using CQL in intraLibrary:

lom.general_title
lom.general_catalogentry_catalog
lom.general_catalogentry_entry
lom.general_language
lom.general_description
lom.general_keyword
lom.general_coverage
lom.general_structure
lom.general_aggregationlevel
lom.lifecycle_version
lom.lifecycle_status
lom.lifecycle_contribute_role
lom.lifecycle_contribute_centity
lom.lifecycle_contribute_date_datetime
lom.lifecycle_contribute_date_description
lom.metametadata_catalogentry_catalog
lom.metametadata_catalogentry_entry
lom.metametadata_contribute_role
lom.metametadata_contribute_centity
lom.metametadata_contribute_date_datetime

lom.metametadata_contribute_date_description
lom.metametadata_metadatascheme
lom.metametadata_language
lom.technical_format
lom.technical_size
lom.technical_location
lom.technical_requirement_type
lom.technical_requirement_name
lom.technical_requirement_minimumversion
lom.technical_requirement_maximumversion
lom.technical_installationremarks
lom.technical_otherplatformrequirements
lom.technical_duration_datetime
lom.technical_duration_description
lom.educational_interactivitytype
lom.educational_learningresourcetype
lom.educational_interactivitylevel
lom.educational_semanticdensity
lom.educational_intendedenduserrole
lom.educational_context
lom.educational_typicalagerange
lom.educational_difficulty
lom.educational_typicallearningtime_datetime
lom.educational_typicallearningtime_description
lom.educational_description
lom.educational_language
lom.rights_cost
lom.rights_copyrightandotherrestrictions
lom.rights_description
lom.relation_kind

lom.relation_resource_description
lom.relation_resource_catalogentry_catalog
lom.relation_resource_catalogentry_entry
lom.annotation_person
lom.annotation_date_datetime
lom.annotation_date_description
lom.annotation_description
lom.classification_purpose
lom.classification_taxonpath_source
lom.classification_taxonpath_taxon
lom.classification_taxonpath_taxon_id
lom.classification_taxonpath_taxon_entry
lom.classification_description
lom.classification_keyword

3.4. Intrallect Context Set

The following Intrallect-specific fields can be searched using CQL in intraLibrary:

intrallect.annotationextension_averagerating
intrallect.tag_tagstring
intrallect.tag_taggedby_identifier
intrallect.tag_taggedby_person

3.5. IntraLibrary Context Set

The following IntraLibrary-specific fields can be searched using CQL in intraLibrary:

Name	Notes
intralibrary.type	The possible values that can be used when searching with this context are: *scorm1.2; scorm2004; singleFile; singleFilePackage; imscp; web; physical; kaltura.*

4. Gather: Metadata Harvesting

An IntraLibrary repository can expose metadata for harvesting by other systems, using the Open Archive Initiative - Protocol for Metadata Harvesting (OAI-PMH). This section describes how intraLibrary supports OAI-PMH to enable external systems to harvest metadata from intraLibrary. It is also possible to use intraLibrary to harvest metadata from other data sources but that is covered in the *Administrator's Guide.* For full details of the protocol visit the OAI web site. IntraLibrary supports version 2.0 of OAI-PMH.

4.1. Opening Collections for Harvesting

Any intraLibrary repository may contain multiple collections of digital objects. The collections in an intraLibrary repository are closed by default, which means records contained in the collection are not harvestable. An administrator-class user of intraLibrary can open the collection for harvesting by changing a single property of the collection. Individual collections are exposed as "sets" under OAI-PMH.

Allow published
records in this
collection to be
harvested by
external systems

4.2. Entry Point

The entry point for harvesting metadata under OAI-PMH from intraLibrary is the root URL for the intraLibrary installation, with "/IntraLibrary-OAI" appended to it:

http://<intralibrary-root>/IntraLibrary-OAI

4.3. Metadata formats

The metadata formats that can be harvested from intraLibrary are summarised in the

following table.

Name	Prefix	Specification
Dublin Core	dc	http://dublincore.org/documents/dces
IMS Metadata 1.2	imsmd	http://www.imsglobal.org/metadata/index.html
IEEE LOM	lom	http://ltsc.ieee.org/wg12/materials.html
ODRL (version 1.1)	o-ex	http://www.w3.org/community/odrl/

4.4. Testing

You can test the harvestability of a repository using the OAI Registry Explorer. This allows you to send commands to intraLibrary. Type in the URL including */IntraLibrary-OAI* and then you can select commands such as "Identify", "List Metadata Formats", etc. Note that for some commands you need to use parameters.

It is also possible to carry out a simple test of OAI Harvesting by typing a URL into a browser. For example:

  http://demonstrator.intralibrary.com/IntraLibrary-OAI?verb=Identify

or

  http://demonstrator.intralibrary.com/IntraLibrary-OAI?verb=ListMetadataFormats

or

  http://demonstrator.intralibrary.com/IntraLibrary-OAI?verb=ListRecords&
        from=2008-01-01&metadataPrefix=imsmd

Note that in each case the OAI command must be preceded by *verb=* and in the case of ListRecords it will usually be qualified by dates and the required metadata format.

4.5. OAI Verbs and Parameters

Complete details of the verbs that can be used and the qualifying parameters that can be applied to each verb are not provided here as they are available at the OAI web site

4.6. Modifying the Metadata Returned

IntraLibrary can be configured to overwrite parts of the metadata harvested under OAI-PMH. In the "Metadata Options" section of the "system" page there is an option to select a metadata template to apply to harvested metadata. The values and macros in the selected template will overwrite the values stored in the relevant fields of each record returned in a harvest operation.

Possible uses for the feature include:
• consistently exposing useful properties, such as PublicURLs in specific metadata fields
• blanking the contents of certain fields for privacy or security reasons
• ensuring consistency in important statements, such as those describing copyright or usage policy

5. Store

The Store web service permits users, who have an intraLibrary contributor-level account, to deposit resources through an external web service rather than through the intraLibrary upload or import facilities.

5.1. Benefits of the Store web service

Sometimes it is not practial or desirable to use the "upload" facility to put files and packages into intraLibrary. For example, a file/package may be too big to upload in a reasonable time over a restricted network connection, or there may be a large set of files or records that need to be transferred from an existing system.

On other occasions it may be desirable to be able to deposit into intraLibrary without leaving another application or web site, or to create a simple, and fast, means of depositing without having to log into the intraLibrary interface.

These are some of the many scenarios in which the Store web service is useful.

5.2. SWORD

The Simple Web-service Offering Repository Deposit (SWORD) is a REST web service based on the Atom Publishing Protocol (AtomPub). SWORD was developed by the JISC-funded SWORD Project, to which Intrallect was a founding contributor.

The SWORD web site at swordapp.org is the main centre for information on SWORD including the specification, a short course and downloads including clients for SWORD.

The following is a summary of how you can use Intrallect's implementation of SWORD to deposit resources and/or metadata into intraLibrary from other systems, desktop programs, or migration tools.

5.3. Entry Point

The entry point for the SWORD service provided by intraLibrary is:
- http://<intraLibrary-root>/IntraLibrary-Deposit

5.4. Using SWORD

Users of the Deposit web service are likely to fall into one of two groups:
- Those who want to use any existing tools to make depositing into intraLibrary quick and easy.
- Those who want to develop applications which can deposit into intraLibrary (for example adding a deposit function to a learning management system.)

The first of these is described in more detail below. Those who want to develop their own applications should refer to the SWORD AtomPub Profile version 1.3

5.5. SWORD Version

From version 3.3 intraLibrary supports SWORD version 1.3. All previous intraLibrary versions supported SWORD version 1.2. Please note that SWORD 1.3 is *not* backward compatible with SWORD 1.2. Any user who has previously used SWORD 1.2 with an earlier version of intraLibrary should read the section Migrating Between SWORD Versions. Anyone implementing SWORD for the first time can ignore that section.

5.6. Migrating Between SWORD Versions

Previous versions of intraLibrary supported SWORD 1.2. As SWORD 1.3 is not backward compatible to SWORD 1.2 anyone using SWORD will need to upgrade their SWORD client before using intraLibrary 3.3. If you have been using the desktop java client available on Sourceforge the client should be upgraded from version 1.0 to version 1.1. To clarify which versions are required refer to the table below:

	IntraLibrary v3.0 or v3.1	IntraLibrary 3.3 or above
SWORD specification	1.2	1.3
SWORD Java client on Sourceforge	1.0	1.1

After updating with the new version of the client no additional work should be required.

5.7. SWORD Clients

On the SWORD web site are a number of clients. At the time of writing the only client that supports IMS Content Packaging, used by intraLibrary, is the Desktop Client (Version 1.1 last modified 2009-06-15). This client can be downloaded and used very effectively with intraLibrary.

There are also Java and PHP libraries available for those who wish to develop their own clients.

5.7.1. Using the SWORD Desktop Client

The remainder of this section will focus on the use of the SWORD Desktop client that is downloadable from http://sourceforge.net/projects/sword-app/files/. This client has both a graphical user interface (GUI) and a command-line interface. By using the command line interface it is possible to create scripts that can carry out large scale deposit of resources into intraLibrary.

Download the SwordClient and unzip it into a suitable location on the user's computer.

Modify the SwordClient.properties file to identify the source of your intraLibrary service document. The serviceurls parameter should be a comma-separated list of urls that offer SWORD deposit services. For intraLibrary this url will be of the form: http://*repository_url*/IntraLibrary-Deposit/service, where *<repository_url>* should be set to the base address of the repository including, if relevant, a port number. If you wish you can remove all other serviceurls. Save your file in SwordClient.properties.

Now that you have everything set up you are ready to start. Since this client supports both graphical user interface (GUI) and command-line interaction both are described separately. You may choose to try only one, or both, as they each have different benefits. In general the GUI is easier to use, particularly at the testing stage, while the command-line interface is easier to modify to make the whole process more automated.

5.7.1.1. Graphical User Interface (GUI)

In the folder you unzipped you will find a file called sword-common-1.1.jar. Run this file. In Windows this is simply a matter of double-clicking on it. As this client is written in Java you will need to have Java installed on your computer.

The first thing to do is connect to the service (repository) and get the service document. To do this you can click on the icon on the top left or choose "Add service" from the File menu. When prompted you will need to supply your intraLibrary username and password. The location should display the serviceurl you added to the properties file. There is no need complete the "on behalf of" section.
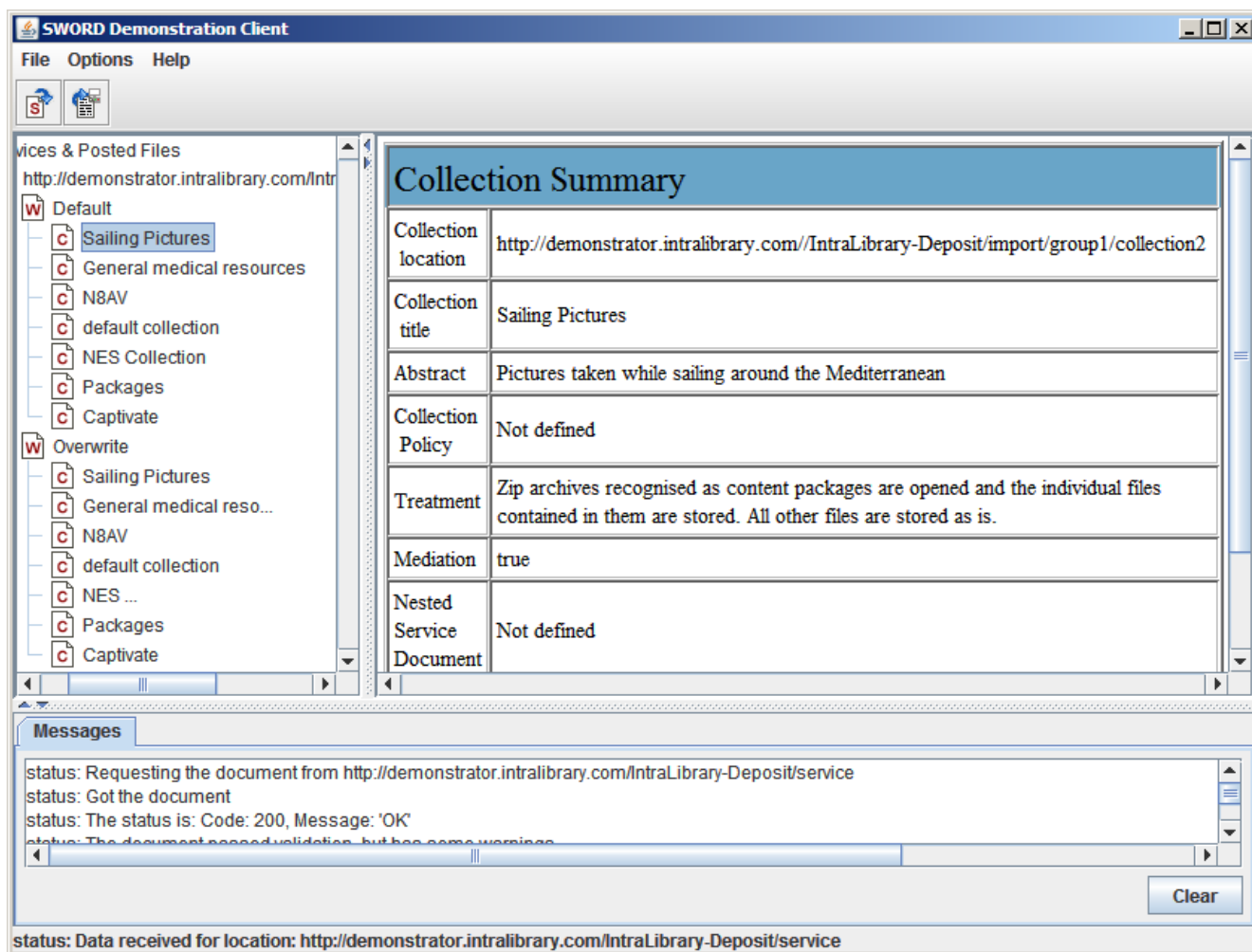
Once the service document is successfully loaded the GUI should look similar to the diagram above. The top left panel shows the workflows available to the user who logged in and within each workflow the collections to which that user can contribute. In this case clicking on one of the collections "Sailing Pictures" displays details of that collection in the panel on the right. The most important part of this is the Collection location which is the URL that must be used as the target destination for depositing a resource into that collection under that workflow. The bottom panel "Messages" shows the detailed message returned by the repository. In most cases this will not be needed as the same information is displayed in a more user-friendly way in the left and right panels.

Note that when a service document is returned it is the service document for *that particular*

user. A different user may have access to different workflows and collections and will therefore receive a different service document. The "Collection location" URL should be treated as a URL for that workflow and collection for that particular user.

Choose a workflow/collection in the left panel to which you want to deposit a resource by clicking on it. Then click on the right-hand icon at the top or select "Post" from the File menu. To deposit a file click "Add" in the dialog box and select the Collection location from the list available and complete the username and password. Then click Browse and select the file, content package or metadata record you want to deposit. All other entries are optional and unnecessary for intraLibrary, so click on "Post File" to complete the deposit. You will receive a confirmation message in the messages panel and in the status bar at the bottom of the screen. In addition your posted file will appear within the specified collection on the left-hand panel. You can continue to deposit additional files.

The next time you log into intraLibrary the files will be in the workflow stage you specified in your work area. See later for details of a workflow that will automatically publish resources as soon as they are deposited.

5.7.1.2. Command Line Interface

Since a command line interface is less user-friendly than a GUI it may not be obvious that this approach can be both useful and powerful. In this section the command line interface is introduced and two examples of its use are described:

- Including a command line in a Windows batch file to produce a desktop icon which can act as a drag-and-drop method of depositing resources into intraLibrary.
- Using a script including command lines to carry out bulk deposit of resources into intraLibrary.

When using the command line interface it is necessary to obtain the service document manually and use that to identify the destination URL for the workflow and collection into which the deposit is to be made. (Or, of course, you can use the GUI to get the deposit target URLs.)

To obtain your service document simply put the url http://*repository_url*/IntraLibrary-Deposit/service into your browser and respond with depositor user name and password when requested. The service document you will receive in return is an XML file of the form:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<service xmlns="http://www.w3.org/2007/app"
   xmlns:atom="http://www.w3.org/2005/Atom"
   xmlns:sword="http://purl.org/net/sword/"
   xmlns:dcterms="http://purl.org/dc/terms/">
  <workspace>
   <atom:title>Default</atom:title>
   <collection href="http://repository.mydomain.edu/IntraLibrary-Deposit/import/group1/collection1">
     <atom:title>open</atom:title>
     <accept>*/*</accept>
     <sword:collectionPolicy>This is the policy statement</sword:collectionPolicy>
     <dcterms:abstract>the default collection</dcterms:abstract>
     <sword:treatment>Zip archives recognised as content packages are opened and the
     individual files contained in them are stored.  All other files are stored as is.</sword:treatment>
     <sword:mediation>true</sword:mediation>
     <sword:acceptPackaging>http://www.imsglobal.org/xsd/imscp_v1p1</sword:acceptPackaging>
   </collection>
  </workspace>
  <workspace>
   <atom:title>sword</atom:title>
   <collection href="http://repository.mydomain.edu/IntraLibrary-Deposit/import/group7/collection8">
     <atom:title>sword</atom:title>
     <accept>*/*</accept>
     <sword:collectionPolicy>This is the policy statement</sword:collectionPolicy>
     <dcterms:abstract>for sword testing</dcterms:abstract>
     <sword:treatment>Zip archives recognised as content packages are opened and the
     individual files contained in them are stored. All other files are stored as is.</sword:treatment>
     <sword:mediation>true</sword:mediation>
     <sword:acceptPackaging>http://www.imsglobal.org/xsd/imscp_v1p1</sword:acceptPackaging>
    </collection>
   <collection href="http://repository.mydomain.edu/IntraLibrary-Deposit/import/group7/collection1">
     <atom:title>open</atom:title>
     <accept>*/*</accept>
     <sword:collectionPolicy>This is the policy statement</sword:collectionPolicy>
     <dcterms:abstract>the default collection</dcterms:abstract>
     <sword:treatment>Zip archives recognised as content packages are opened and the
     individual files contained in them are stored. All other files are stored as is.</sword:treatment>
     <sword:mediation>true</sword:mediation>
     <sword:acceptPackaging>http://www.imsglobal.org/xsd/imscp_v1p1</sword:acceptPackaging>
```

```
    </collection>
  </workspace>
  <sword:verbose>true</sword:verbose>
  <sword:version>1.3</sword:version>
  <sword:noOp>true</sword:noOp>
  <sword:maxUploadSize>209714176</sword:maxUploadSize>
</service>
```

The basic concepts featured in this file are:

- **group**: A group of users in intraLibrary.
  - The *<workspace>* element indicates a group.
  - All the groups through which a user has deposit access, will be listed using separate *<workspace>* elements.
- **collection**: A collection in intraLibrary.
  - The *<collection>* element indicates a group.
  - Within each workspace (i.e. group), the collections for which the user has deposit access will be listed.
  - **The "href" attribute of each collection contains the URL to be used for any deposit request that targets the relevant group and collection.**
- **metadata**: Brief information about the groups and collections.
  - The element *<atom:title>* contains the title of the collection or group.
  - The element *<dcterms:abstract>* contains the description of the collection.
  - The element *<accept>* contains a list of mime-types that can be deposited into the collection.
    - Normally any mime-type can be deposited, as indicated by *<accept>*/*</accept>*
  - The element *<sword:acceptPackaging>* contains the XML namespace of the packaging format supported for ingest by the repository.
    - For intraLibrary the value of this element is normally *http://www.imsglobal.org/xsd/imscp_v1p1*, which means the repository supports IMS and SCORM content packages.
  - The element *<sword:treatment>* contains a description of how a package of content will be handled when deposited into the repository.

To carry out a command line deposit on Windows (other operating systems have similar approaches) open a command window and type the command

CALL java -jar *SWORDHOME*\sword-common-1.1.jar -cmd -t post -u *USERNAME*
        -p *PASSWORD* -href *DEPOSIT_TARGET*

-filetype *FILE_TYPE* -file *FILENAME*

where the parts in italics should be replaced by the relevant parameters:

- **SWORDHOME** the full path name of the location where SWORD was installed and the sword-common-1.1.jar file resides.
- **USERNAME** the intraLibrary user name of the depositor.
- **PASSWORD** the intraLibrary pasword of the depositor.
- **DEPOSIT_TARGET** the target URL for the selected workspace and collection, for example http://repository.mydomain.edu/IntraLibrary-Deposit/import/group7/collection8 from the above service document.
- **FILE_TYPE** the mime type of the file being deposited. In fact intraLibrary ignores this and it can be set to application/zip in all cases.
- **FILENAME** the name of the file to be deposited, including the full path from the current location and any file extension.

Clearly it would be tedious to have to type such a command every time you want to deposit a file. A much simpler option is to create a batch file. In the batch file all the parameters that do not change can be set and only the parts which do change need to be provided. In fact if we assume that the batch file is always being used by the same user to deposit into the same workspace and collection then the only parameter that changes is the FILENAME. Below is an example of suitable batch file that you can cut and paste.

```
@echo off
setlocal
rem -----------------------------------------------------------
rem          Drag files onto shortcut to this to
rem    deposit into a SWORD Compliant Repository
rem -----------------------------------------------------------
rem  Copyright Intrallect Ltd 2007, 2008
rem  Released under Creative Commons Attribution licence.
rem  http://creativecommons.org/licenses/by/2.5/scotland/
rem -----------------------------------------------------------
rem      You can edit the following properties
rem -----------------------------------------------------------
set SWORDHOME=c:\sword-client-1.1
set DEPOSIT_TARGET=http://localhost:7070/intralibrary3p3/
                           IntraLibrary-Deposit/import/group4/collection1
set FILE_TYPE=application/zip
set USERNAME=sword
```

```
set PASSWORD=sword
rem -----------------------------------------------------------
rem      Don't change anything below here
rem -----------------------------------------------------------
:start
echo depositing %1
CALL java -jar %SWORDHOME%\sword-common-1.1.jar -cmd -t post -u %USERNAME% -p
%PASSWORD% -href %DEPOSIT_TARGET% -filetype %FILE_TYPE% -file %1 > log.txt
shift
if not "%~1"=="" goto start
echo Finished
pause
endlocal
```

Note that the line containing "DEPOSIT_TARGET" and the following line ending "collection1" should all be on a single line. Similarly the line beginning "CALL" and the following line ending "log.txt" should all be on a single line.

To use this save it as a batch file, in Windows save it with a .bat extension. Create a shortcut to this file and place the shortcut on the desktop. Now any file that is dropped on the icon is treated as a parameter to the bacth file (%1 in the script) and will be passed for deposit. If multiple files are selected then they will be deposited one after the other.

If one user needs to submit to several different/workspaces/collections then several different batch files can be created as drag-and-drop targets.

To create a script that would deposit a large number of files all that is required is to gather the names of all the files to be deposited and cycle round all the files calling this command each time with the next file name on the list.

5.8. Types of files to be deposited

SWORD enables only one file to be deposited. This means that when multiple files are to be deposited they should be packaged in an IMS (or SCORM) content package. If metadata is to be included when the deposit is made then, again, the metadata and file(s) need to be in an IMS Content Package. The options for depositing are summarised below:
- **Single files:** Any single file (for example, Powerpoint, PDF, image, Word, MP3, or

video clip) can be deposited through SWORD. No metadata is included in the upload but a metadata template can be applied for the contrubutor which will generate metadata as the file is deposited. Only if the metadata created by the template meets the application profile requirements of the workflow will the file be automatically publishable.

- **Single metadata records:** A metadata record (for example to describe a "web resource") can be deposited through SWORD. The file should be an XML file with the same structure as you obtain when you export a LOM metadata record from intraLibrary. In fact this is a good way to create a template for a metadata record. Note that metadata fields which are not used can be omitted from the XML file.
- **IMS Content Packages:** If you wish to deposit more than one file at a time, either a file and associated metadata or several files (for example a web page and the images displayed in the web page) then they must be in an IMS Content Package. Various tools are available for content packaging, such as Reload.

5.9. Workflows for SWORD deposit

A user can deposit, using SWORD, into any workflow to which they have access. The deposited resource will be placed in the stage of the workflow selected by the deposit URL. In many cases that is all that is required and the normal workflow process can proceed after the deposit.

However, one benefit of using SWORD is that it is possible to deposit and have a resource published in a single step. To do this it is useful to create a specific workgroup intended for SWORD deposit which has its own workflow. One characteristic of this workflow should be that it contains two stages and that "Publish" should occur between the first and second stage.

Simple SWORD
The simplest possible example of a SWORD workflow
Workflow Status: *In use*

Stage

Dummy | 1

Process

Dummy process

Group Roles

Actions

Publish

Stage

Deposit | 2

Process

Deposit and publish

Group Roles

Resource owner

Actions

Delete

Edit Metadata

Upload

The screenshot above shows a workflow in intraLibrary that can support immediate publishing and deposit through SWORD. This workflow is very, very basic and you may want to include other actions and/or roles. It is intended as a demonstration of the minimum required to achieve immediate publication. Please note that this workflow alone is not sufficient to achieve publication of resources. In addition the following are required:

- Application profile: the workgroup must use a metadata application profile in which all mandatory fields can be completed from the metadata template.
- Metadata template: A metadata template must be used which is capable of completing all mandatory metadata fields. While fields such as the file size and type and upload date and contributor are easily created it is more difficult to create title, description and keywords. A common approach is to use the file name as the title. The description and keywords can be left as optional or can have fixed values included in the metadata template if all the resources are of similar origin. Also, the template can be used to provide a fixed classification for all resources using this mode of deposit or the classification must not be mandatory.
- Profile: The user must have their profile set to the required metadata template so that it is applied to all resources uploaded in this way. For this reason it may be advantageous to use a username and password specifically for sword deposits in some cases.

6. Inform

The Inform web service provides intraLibrary's capabilities for "pushing" information and updates from intraLibrary to exteral systems. It has two components, RSS news feeds and Apple podcasts, both of which are created within intraLibrary in a similar way.

6.1. Benefits of Inform web service

The first benefit of using RSS feeds and podcasts is the ability to create a set of resources that you can view in any another application that supports news feeds or podcasts. You can create this set of resources by browsing to any node of any taxonomy, searching using advanced or simple searches, or by using a node of your own My Favourites. The feed could be about a particular topic and it might be placed in a learning management system so that a group of students can access the resources, or it might be in your own news reader or iTunes app.
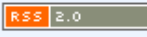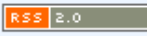
The second major benefit is that when new resources are added that meet the conditions you have specified for your feed you will be told without needing to log into intraLibrary.
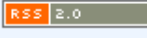
Whether you are monitoring a particular node of a taxonomy or searching for resources on a particular topic in a particular format and that other people have awarded a 3 star or more rating, your news reader or podcasts system will be updated when new resources are added that meet the criteria.

The frequency with which the new resource updates are fed through to applications can be set by the administrator in the System page under RSS Options. Possible values are 1 minute, 1 hour, 1 day, 1 week or 1 month. The default is 1 day.

6.2. Creating news feeds and podcasts

There are many opportunities to create news feeds and podcasts within intraLibrary. Every time a list of results appears on screen, whether from a simple or advanced search, from browsing the nodes of a classification or in My Favourites there is always a symbol which you can click on to create a news feed or podcast. After selecting a name, and (optionally) a description of your feed you are presented with the option to choose an RSS feed or a podcast. To pass either of these feeds to another system you can either drag and drop or copy and paste them. When creating a feed from a list of results in intraLibrary it should be recognised that the feed contains all items in the search results not just those on the specific pagenated page.

Name	Description	Language			
blended	blended learning resources	English	View results	RSS 2.0	RSS PODCAST
5-star resources	average star rating equal to 5	English	View results	RSS 2.0	RSS PODCAST
5 top rated, most recent sailing items		English	View results	RSS 2.0	RSS PODCAST

Each user has a section in their intraLibrary Profile in which they can manage their feeds as shown above. Here you can delete any feeds that are no longer required and preview the results from any feed. You can also get the RSS and Podcast icons provided when you first created the feed.

6.3. Using RSS News Feeds

There are far too many RSS news feed applications to mention here. In many cases the feeds are used to bring information to the user in tools that they use on a regular basis. However, it may also be that the feed is be transformed in some way for another purpose.

One way to transform feeds is to use Yahoo Pipes, a useful way to mashup data from RSS feeds.

News feeds generated from intraLibrary conform to the "RSS 2.0" specification. This specification states that an item in an RSS feed has three required elements:
- **title:** the title of the item (taken from the intraLibrary Title metadata)
- **link:** the URL of the item (taken from the intraLibrary publicURL)
- **description:** the item synopsis (taken from the intraLibrary Description metadata - if more than one despripton is present only the last description field is included.)

Of the optional elements intraLibrary supports:
- **pubDate:** the publication date of the resource (taken from the date uploaded into intraLibrary, not from any date metadata)
- **guid:** stands for globally unique identifier. It is a string that uniquely identifies the item. When present, an aggregator should choose to use this string to determine if an item is new. (taken from the OAI identifier for the item whether or not this is present in the metadata)

RSS items can contain extended data as long as the elements and attributes are defined in a namespace. IntraLibrary supports only one item of extended data:
- **dc:date** the same as pubDate but provided in Dublin Core/IEEE LOM format.

The following is an example of an RSS feed produced by intraLibrary:

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:content="http://purl.org/rss/1.0/modules/content/"
     xmlns:taxo="http://purl.org/rss/1.0/modules/taxonomy/"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd"
     xmlns:dc="http://purl.org/dc/elements/1.1/" version="2.0">
  <channel>
   <title>test2</title>
   <link>http://localhost:7070/intralibrary3p3</link>
   <description>Enter news feed description here (optional)</description>
    <item>
      <title>Intrallect</title>
```

```
    <link>http://localhost:7070/intralibrary3p3/IntraLibrary?command=open-preview&
    learning_object_key=i513n67627t</link>
    <description>Intrallect web site</description>
    <pubDate>Thu, 26 Aug 2010 10:11:45 GMT</pubDate>
    <guid isPermaLink="false">oai:com.intrallect.poptart:966</guid>
    <dc:date>2010-08-26T10:11:45Z</dc:date>
  </item>
  <item>
    <title>Sabbatical Yacht</title>
    <link>http://localhost:7070/intralibrary3p3/IntraLibrary?command=open-preview&
    learning_object_key=i171n5949t</link>
    <description>A photograph of the yacht Sabbatical in Croatia</description>
    <pubDate>Thu, 26 Aug 2010 07:25:25 GMT</pubDate>
    <guid isPermaLink="false">oai:com.intrallect.poptart:66</guid>
    <dc:date>2010-08-26T07:25:25Z</dc:date>
  </item>
  </channel>
</rss>
```

When an RSS feed is produced by intraLibrary the order in which items appear is the same as the default order in which search results appear, that is, they are sorted by relevance with the most relevant first. Anyone wishing to transform the feed, for example to another order, may find Yahoo Pipies useful.

6.4. Using Podcasts

There are software applications that support podcasts for almost every operting system avaialble including mobile devices. Probably the most common application is iTunes.

Podcast are, in fact, news feeds that include an enclosure which points to the relevant file.

Podcasts generated from intraLibrary include the following elements:
- **title:** the title of the item (taken from the intraLibrary Title metadata)
- **description:** the item synopsis (taken from the intraLibrary Description metadata - if more than one description is present only the last description field is included.)

- **enclosure:** describes a media object that is attached to the item (intraLibrary provides a direct URL to the media file, the file size and the file mime/type)

Of the optional elements intraLibrary supports:

- **guid:** stands for globally unique identifier. It is a string that uniquely identifies the item. When present, an aggregator should choose to use this string to determine if an item is new. (taken from the OAI identifier for the item whether or not this is present in the metadata)

In podcasts IntraLibrary supports two items of extended data:

- **itunes:explicit** used by itunes to identify explicit content (always set to "no" in intraLibrary)
- **itunes:keywords** a list of comma separated words or phrases (taken from the IntraLibrary keywords metadata)

The following is an example of a podcast produced by intraLibrary:

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:content="http://purl.org/rss/1.0/modules/content/"
    xmlns:taxo="http://purl.org/rss/1.0/modules/taxonomy/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd"
    xmlns:dc="http://purl.org/dc/elements/1.1/" version="2.0">
  <channel>
   <title>4 or more</title>
   <link>http://localhost:7070/intralibrary3p3</link>
   <description>Enter news feed description here (optional)</description>
   <language>en</language>
   <dc:language>en</dc:language>
    <item>
     <title>Rob sailing</title>
     <description>Rob at the helm of Sabbatical. The boat has two wheel, one on each side.
     Which wheel you use depends on the position of the sails.</description>
     <enclosure
     url="http://localhost:7070/intralibrary3p3/open_virtual_file_path/i1026n4355t/DSCF1208.JPG"
     length="69706" type="image/jpeg" />
     <guid isPermaLink="false">oai:com.intrallect.poptart:87</guid>
     <itunes:explicit>no</itunes:explicit>
     <itunes:keywords>Sailing, Yacht</itunes:keywords>
    </item>
    <item>
```

```
            <title>Navigating towards Ston</title>
            <description>Navigating a fairly narrow channel towards Ston in Croatia.
            Had to motor for this section.</description>
            <enclosure
            url="http://localhost:7070/intralibrary3p3/open_virtual_file_path/i1368n4805t/Tim steers.jpg"
            length="76124" type="image/jpeg" />
            <guid isPermaLink="false">oai:com.intrallect.poptart:96</guid>
            <itunes:explicit>no</itunes:explicit>
            <itunes:keywords>Sailing, Yacht</itunes:keywords>
        </item>
    </channel>
</rss>
```

7. Additional Notes

Notes in this section define the status of any other integration methods offered by intraLibrary.

7.1. Authenticated IntraLibrary REST Services

This Guide is intended for those developing products or applications based on intraLibrary's open services. IntraLibrary also has a set of REST-based web services that enable external application developers to work closely with intraLibrary using services available only to authenticated intraLibrary users. Please contact Intrallect if you wish to use these authenticated web services.

7.2. IMS DRI

The IMS Digital Repository Interoperability (IMS DRI) specification is no longer supported by intraLibrary. It is believed that this interface has not been used for some time.

7.3. Repository Java-API

IntraLibrary's Java-API is now deprecated for external use and support for the Java-API is expected to be removed at the next release.

7.4. Authentication

IntraLibrary supports various methods of external authentication including: LDAP, Active Directory, EduServ OpenAthens, and Shibboleth (both native and through the UK Access Federation. For more details of any specific authetication method please contact Intrallect directly through support@intrallect.com.

7.5. Using Events

IntraLibrary previously exposed an "Event Handling" framework in which customized actions could be taken by customer-defined "event handlers". This has now been replaced by the IntraLibrary Reporting Service. Please contact Intrallect directly for more details.

8. References

- Open Archive Initiative - Protocol for Metadata Harvesting (OAI-PMH): http://www.openarchives.org/pmh/
- OAI Repository Explorer: http://re.cs.uct.ac.za/
- Search and Retrieve URL (SRU): http://www.loc.gov/standards/sru/index.html
- Dublin Core Metadata: http://dublincore.org/documents/dces
- IMS Metadata: http://www.imsglobal.org/metadata/index.html
- IMS Content Packaging: http://www.imsglobal.org/content/packaging/
- IEEE LOM: http://ltsc.ieee.org/wg12/materials.html
- Open Digital Rights Language (ODRL): http://www.w3.org/community/odrl/
- Simple Web-service for Ordering Repository Deposit (SWORD): http://swordapp.org/