

Item Versioning Support

DSpace Item Level Versioning Support

- [DSpace Item Level Versioning Support](#)
 - [Introduction](#)
 - [Stakeholders](#)
 - [Business Requirements / Goals](#)
 - [User Scenarios](#)
 - [Higher Priority Requirements](#)
 - [Lower Priority Requirements](#)
 - [Mockups](#)
 - [Item View](#)
 - [Versioning edits to an Item](#)
 - [Examples of Items with an additional Item Version Attached](#)
 - [Version 1](#)
 - [Version 2](#)
 - [Edit Item and Version Management](#)
 - [Technical Requirements and Existing Implementation](#)
 - [Administration](#)
 - [Services that are effected by user scenarios](#)
 - [Work Areas for Implementation of Version History](#)
 - [Dryad Solution for Versioning](#)
 - [DSpace Data Model and Versioning](#)
 - [DSpace Item Objects:](#)
 - [DSpace Bundle Objects:](#)
 - [DSpace Bitstream Objects:](#)
 - [Services to support Versioning and Alternative Identifiers.](#)
 - [Versioning Service](#)
 - [Identifier Service](#)
 - [Application IdentifierService Interface](#)
 - [Backend IdentifierProvider Interface](#)
 - [Example of Spring Configuration for Default HandleProvider Support](#)
 - [Support for resolving identifiers via DSpace User Interface](#)
 - [The Resource Path "/resource/..."](#)
 - [Basic XMLUI HTML Representation](#)
 - [RIS Representation](#)
 - [BibTex Representation](#)

Newer information available



This page contains an initial requirements analysis for the Item Level Versioning Support, contributed to DSpace 3.0. Refer to the [official DSpace 3 documentation](#) for the most up to date information on Item Level Versioning.

Introduction

DSpace Item Level Versioning Support is a funded open source activity to bring work done by NESCent and @mire into the DSpace 3.0 Codebase. Additional partners have signed onto this activity to assure that resources are available to drive Item Versioning Support forward and make it available to the larger DSpace community in the next release.

Stakeholders

Key Stakeholders involved with Item Versioning Support:

- Mark Diggory - @mire
- Holly Miller - MBL
- Diane Rielinger - MBL
- Lisa Raymond - WHOI
- Ryan Scherle - NESCent / Dryad

Business Requirements / Goals

The history of an item should be available, and it should be possible to cite a particular version of the item.

User Scenarios

General user actions that would generate a new Version of a new Item and their overall impact on creation of a new DSpace item.

1. Release of a new scientific dataset with additional data
2. Release of a corrected scientific dataset with altered data
3. Release of a revised or corrected journal article, technical report, ...

Higher Priority Requirements

1. What is Versionable
 - a. Versioning is at the level of an Individual Item
 - b. Should preserve the current state of metadata, bitstreams and resource policies attached to the item.
2. Access, Search and Discovery
 - a. Only the most recent version of an item is available via the search interface (possibly configurable)
 - b. Previous Versions of Items should continue to be visible, citable and accessible
 - c. The Previous Versions Bitstreams are retained. If something was once retrievable, it is always retrievable.
3. Identifiers
 - a. Each version of an Item is represented by a separate "versioned" identifier (Handle or DOI)
 - b. A base "versionhistory" Identifier points to the most recent version of the Item.
 - c. A revision identifier also exists that is unique to the specific version.
 - d. When a new version of an Item is deposited, a new revision identifier will be created.
4. Presentation
 - a. On the item page, there is a link to view previous/subsequent versions.
 - b. Each version of the item should include additional provenance details about who created the version, when and why.
 - c. By examining the metadata or identifiers, it is possible to determine whether an item is the most recent version, the original version, or an intermediate version.
5. Access Control and Rights
 - a. Certain roles should be able to generate a new version of the item via submission, sword or LNI deposit.
 - b. Submitters: Creation of a new version is initiated by the original submitter. On the "submissions" page, submitter should see all of their archived submissions.
 - c. Collection Manager, Administrators: should have a button to submit a new version of a Item accessible in the Edit Item administrative interface.
 - d. Rights to access a specific Item should transmute as well to previous versions
 - e. Rights to access a specific Bitstream should also transmute to previous versions.
6. Data Integrity
 - a. The relationships between versions should not be brittle and breakable by manipulating Item metadata records.
 - b. The relationships between versions should be preserved and predictable in various Metadata Exports (OAI, Packagers, ItemExport)
 - c. The relationships between versions should be maintained in SWORD, LNI and AIP packaging and be maintained in updates and restorations.

Lower Priority Requirements

1. All metadata and content changes within a "Version" should be logged in an audit trail

Mockups

Item View

Versioning edits to an Item

With a full feature versioning capability on Items, we may possibly be able to support restoration of Item Versions through an interface such as this WIKI page example interface.

Version History			
Item	Version	Date	Summary
doi:10.5061/dryad.r460n.2*	2	2012-04-30 14:45:27.657	Testing versioning by adding a file.
doi:10.5061/dryad.r460n.1	1	2012-04-30 14:45:27.647	
* Selected Version			

Examples of Items with an additional Item Version Attached

Version 1

An example of an initial Version within Dryad

Data from: Testing embargo

This is not the most current version of this Item [\[View current version\]](#)

When using this data, please cite the original article:

Citation is not yet available for this article from Testing journal. It will become available shortly after the article is published.

Additionally, please cite the Dryad data package:

Feinstein E(2012) Data from: Testing embargo. Dryad Digital Repository.
[doi:10.5061/dryad.r460n.1](https://doi.org/10.5061/dryad.r460n.1)

[Cite](#) | [Share](#)

**Dryad
Package
Identifier** doi:10.5061/dryad.r460n.1 3 views

**Date
Deposited** 2012-04-30T18:40:49Z

[Show Full Metadata](#)

testing_data 0 downloads [View File Details](#)

At the request of the author, this item is embargoed until one year after the associated article is published.

To the extent possible under law, the authors have waived all copyright and related or neighboring rights to this data.



Version History

Item	Version	Date	Summary
doi:10.5061/dryad.r460n.2	2	2012-04-30 14:45:27.657	Testing versioning by adding a file.
doi:10.5061/dryad.r460n.1 *	1	2012-04-30 14:45:27.647	

* Selected Version

Version 2

An example of a second version within Dryad

Data from: Testing embargo

When using this data, please cite the original article:

Citation is not yet available for this article from Testing journal. It will become available shortly after the article is published.

Additionally, please cite the Dryad data package:

Feinstein E(2012) Data from: Testing embargo. Dryad Digital Repository.
[doi:10.5061/dryad.r460n.2](https://doi.org/10.5061/dryad.r460n.2)

[Cite](#) | [Share](#)

**Dryad
Package
Identifier** doi:10.5061/dryad.r460n.2 0 views

**Date
Deposited** 2012-04-30T19:37:26Z

[Show Full Metadata](#)

testing_data 0 downloads [View File Details](#)

At the request of the author, this item is embargoed until one year after the associated article is published.

To the extent possible under law, the authors have waived all copyright and related or neighboring rights to this data.



[OPEN](#) [DATA](#)

testing_data_2 0 downloads [View File Details](#)

additional file added during versioning

At the request of the author, this item is embargoed until the associated article is published.

To the extent possible under law, the authors have waived all copyright and related or neighboring rights to this data.



[OPEN](#) [DATA](#)

Version History

Item	Version	Date	Summary
doi:10.5061/dryad.r460n.2 *	2	2012-04-30 14:45:27.657	Testing versioning by adding a file.
doi:10.5061/dryad.r460n.1	1	2012-04-30 14:45:27.647	

* Selected Version

Edit Item and Version Management

[TODO]

Technical Requirements and Existing Implementation

Administration

1. Current Code Locations
 - a. <https://dryad.googlecode.com/svn/trunk/dryad/dspace/modules/>

Services that are effected by user scenarios

- Local User Submissions and Curator Reviews.
- SWORD and LNI Deposit interfaces
- Local ItemImport and Export Command-line services.
- External Harvesters (OAI and DSpace)
- Academic Search Engines and Catalogs.

Work Areas for Implementation of Version History

1. Database Modifications
2. Data Access Objects and Domain Model definition
3. Enhancing DSpace XMLUI Item Adapter to expose Version Details on Items
4. Item User Interface Changes to support Versioning
 - a. Details that should be presented in User Interface
 - b. Actions that should be possible on existing versions (compare, revert, delete, withdraw)
 - c. Actions that should be possible to generate a new version (submission, deposit, import)
 - d. Roles for which new versions should be possible (submitter, author, curator)
5. Local User Submissions and Curator Reviews.
6. Services that are effected by user scenarios
 - a. SWORD v1 and v2 Version creation on Update
 - b. LNI Version on Update
 - c. Local ItemImport and Export Command-line services.
 - d. Metadata Export and Import
 - e. External Harvesters (OAI and DSpace)
7. Further Requirements to Consider
 - a. Deleting an Item from the Repository
 - b. Moving and Item Between Collections
 - c. Mapping an Item to another collection

Dryad Solution for Versioning

DSpace Data Model and Versioning

Past Architectural Review Group work on the DSpace Data Model and versioning focused its research and recommendations on replicating the entire DSpace Item contents when a new DSpace Item was created in DSpace. This means that all item metadata and content would be replicated. This was done to ease the complexity of managing references across individual Items to the content and metadata that would be considered the same across those versions.

In the GSoC project to address versioning, we made an effort to optimize on the above situation and employed the following strategy for the production of a New version of a DSpace Item

DSpace Item Objects:

For every new Version a separate DSpace Item will be created that replicates the metadata and bundle records to allow multiple Items to share references to existing bitstreams that do not change across versions.

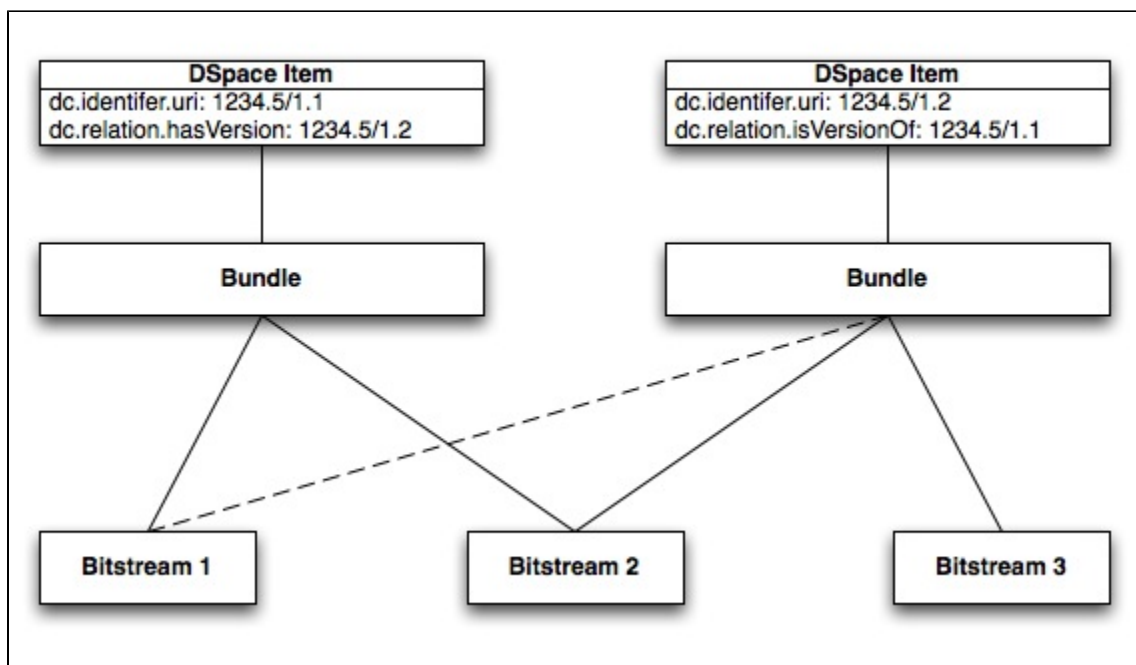
DSpace Bundle Objects:

For every new version of an Item, a Bundle will be created and the Bundle will link to all the preexisting bitstreams for the original Item. This means that Bitstreams may be associated with more than one Item by being linked by Bundle.

DSpace Bitstream Objects:

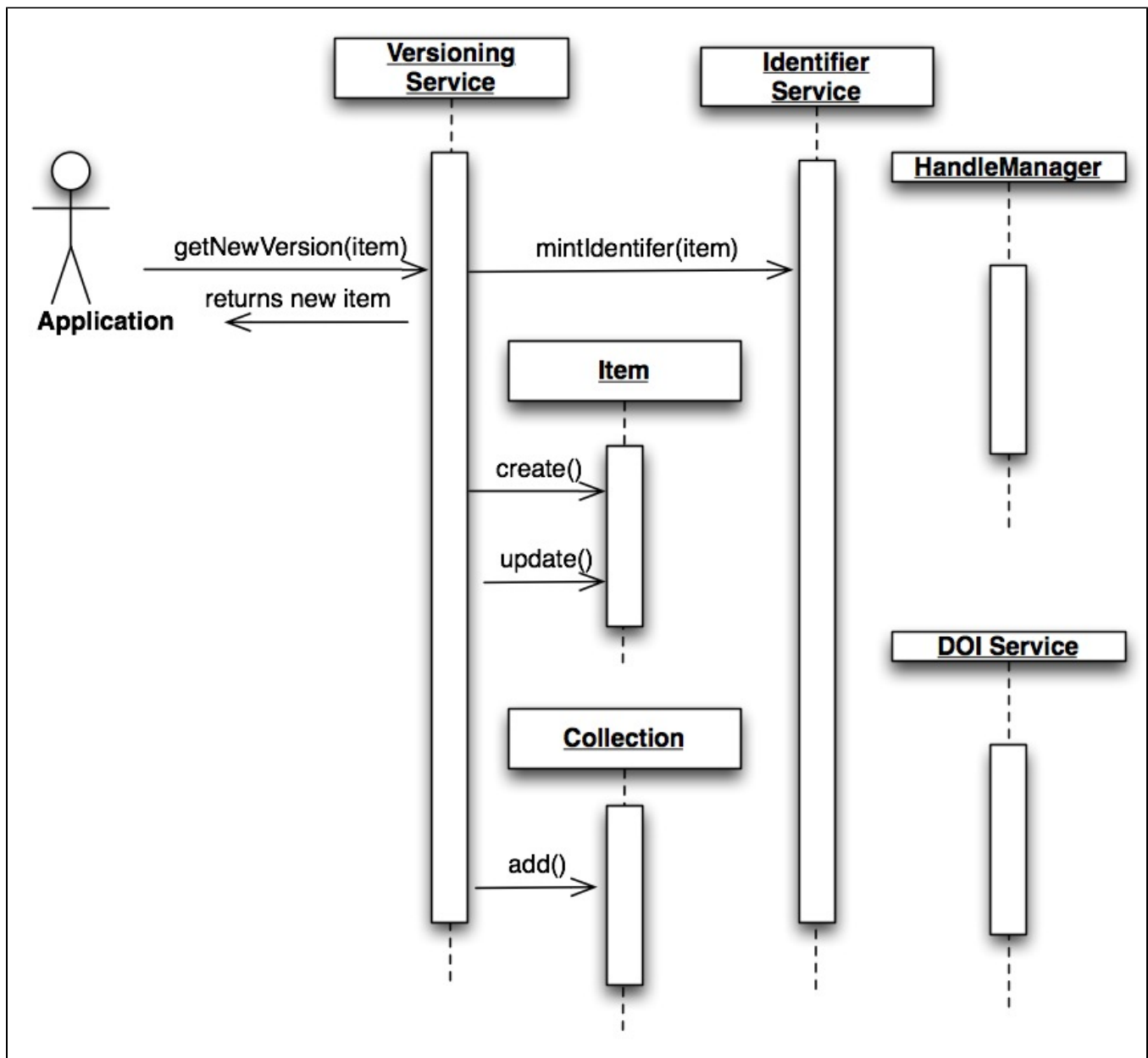
The versioning support in Dryad is based on replicating DSpace Item, MetadataValue and Bundle Records, creating an identical version of all Metadata and Bitstream relationships. The new version of the Item "conserves" Bitstream Contents by reusing references to the persisted Bitstreams across individual Item Revisions.

The Bitstream deletion logic in DSpace is enhanced to support detecting if the Bitstream is no longer a member of any Item Revisions before being flagged as deleted.



Services to support Versioning and Alternative Identifiers.

DSpace Item Versioning will be encapsulated as an Extensible Service that may be reimplemented by the local repository maintainers to produce alternate versioning behaviors and Identifier Schemes. Versioning Services layer on top of IdentifierServices dedicated to Encoding, Resolution, Minting and Registration of Identifiers for specific DSpace Items and Bitstreams. It is through this highly extensible layering of functionality where local developers can alter the versioning behavior and introduce their own local enhancements. The DSpace Service Manager, based on the Spring Framework, provides the key leverage for this flexibility.



Versioning Service

The Versioning Service will be responsible for the replication of one or more Items when a new version is requested. The new version will not yet be preserved in the Repository, it will be preserved when the databases transactional window is completed, thus when errors arise in the versioning process, the database will be properly kept in its original state and the application will alert that an exception has occurred that is in need of correction.

The Versioning Service will rely on a generic IdentifierService that is described below for minting and registering any identifiers that are required to track the revision history of the Items.

Dryad Example: [Version 1](#), [Version 2](#) of same document (see versions listed at bottom of page)

Code: [Google Code](#)


```

public interface VersioningService {

    Version createNewVersion(Context c, int itemId);

    Version createNewVersion(Context c, int itemId, String summary);

    void removeVersion(Context c, int versionID);

    void removeVersion(Context c, Item item);

    Version getVersion(Context c, int versionID);

    Version restoreVersion(Context c, int versionID);

    Version restoreVersion(Context c, int versionID, String summary);

    VersionHistory findVersionHistory(Context c, int itemId);

    Version updateVersion(Context c, int itemId, String summary);

    Version getVersion(Context c, Item item);

}

```

Identifier Service

The Identifier Service maintains an extensible set of IdentifierProvider services that are responsible for two important activities in Identifier management:

1. Resolution: IdentifierService act in a manner similar to the existing HandleManager in DSpace, allowing for resolution of DSpace Items from provided identifiers.
2. Minting: Minting is the act of reserving and returning an identifier that may be used with a specific DSpaceObject.
3. Registering: Registering is the act of recording the existence of a minted identifier with an external persistent resolver service, these services may reside on the local machine (HandleManager) or exist as external services (PURL or DEZID DOI registrations services)

Dryad Example: The same document can be accessed via different identifiers: DOI link (<http://dev.datadryad.org/resource/doi:10.5061/dryad.1385.2>), Handle link (<http://dev.datadryad.org/resource/info:hdl/10255/dryad.36265>), default DSpace Link (<http://dev.datadryad.org/resource/10255/dryad.36265>)

Code: [Google Code](#)

Application IdentifierService Interface

```

public interface IdentifierService {

    void reserve(Context context, Item item) throws AuthorizeException, SQLException, IdentifierException;

    String register(Context context, Item item) throws AuthorizeException, SQLException, IdentifierException;

    DSpaceObject resolve(Context context, String identifier) throws IdentifierNotFoundException, IdentifierNotResolvableException;

    void delete(Context context, Item item) throws AuthorizeException, SQLException, IdentifierException;

}

```

Backend IdentifierProvider Interface


```

public abstract class IdentifierProvider {

    protected IdentifierService parentService;

    protected ConfigurationService configurationService;

    @Autowired
    @Required
    public void setConfigurationService(ConfigurationService configurationService) {
        this.configurationService = configurationService;
    }

    public void setParentService(IdentifierService parentService) {
        this.parentService = parentService;
    }

    public abstract boolean supports(String identifier);

    public abstract String register(Context context, DSpaceObject item) throws IdentifierException;

    public abstract String mint(Context context, DSpaceObject dso) throws IdentifierException;

    public abstract DSpaceObject resolve(Context context, String identifier, String... attributes)
        throws IdentifierNotFoundException, IdentifierNotResolvableException;;

    public abstract void delete(Context context, DSpaceObject dso) throws IdentifierException;
}

```

Example of Spring Configuration for Default HandleProvider Support

<http://code.google.com/p/dryad/source/browse/trunk/dryad/dspace/modules/identifier-services/src/main/resources/spring/spring-dspace-core-services.xml>

```

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-2.5.xsd">

    <bean class="org.dspace.identifier.HandleIdentifierProvider" scope="singleton"/>
    <bean class="org.dspace.identifier.InternalIdentifierProvider" scope="singleton"/>
    <bean class="org.dspace.identifier.IdentifierServiceImpl"
        id="org.dspace.identifier.IdentifierService"
        autowire="byType" scope="singleton"/>

</beans>

```

Addons Can easily add additional Providers without directly needing to alter DSpace Code

See [Dryad Specific Identifier and Versioning Services](#)

```

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-2.5.xsd">

    <bean class="org.dspace.versioning.DryadPackageVersionProvider" autowire="byType"/>
    <bean class="org.dspace.identifier.DOIIdentifierProvider" scope="singleton"/>

</beans>

```

The IdentifierProviders are also responsible for altering any existing DSpace metadata fields that need to be altered on the new and previous items to record the relationship between Item Versions.

Support for resolving identifiers via DSpace User Interface

The Resource Path "/resource/..."

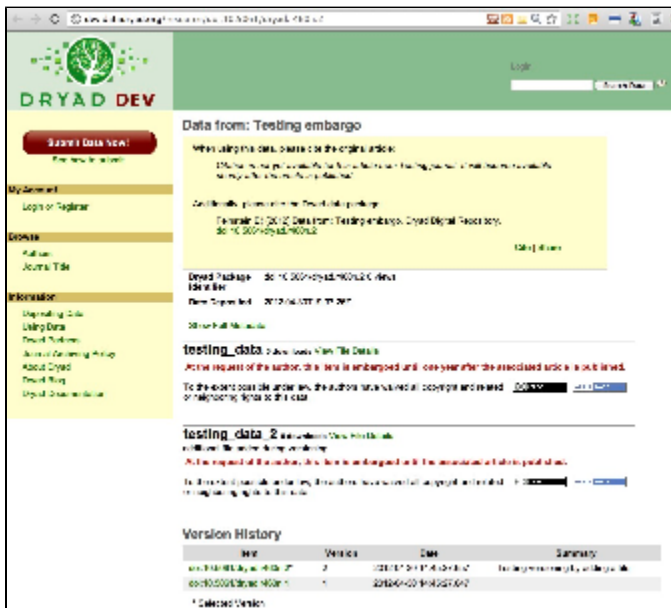
Dryad supports a unique identifier resolving service based on Spring WebMVC which allows Identifiers registered within the IdentifierServices to be dereferenced to varied representations of the actual DSpace Items. This resolution can span dspace to alternative webapplications to support various representation formats. The mechanism is extendable in the XMLUI and will be utilized to support dereferencing identifiers via content negotiation to expose alternative format representations of the Item, in Dryad the following are currently implemented.

Basic XMLUI HTML Representation

<http://dev.datadryad.org/resource/doi:10.5061/dryad.r460n.2>

<http://dev.datadryad.org/resource/info:hdl/10255/dryad.39114>

<http://dev.datadryad.org/resource/10255/dryad.39114>



RIS Representation

<http://dev.datadryad.org/resource/doi:10.5061/dryad.r460n.2/citation/ris>

```
TY - DATA
ID - doi:10.5061/dryad.r460n.2
T1 - Data from: Testing embargo
AU - Feinstein E(
Y1 - 2012/04/30/
JF - Testing journal
PB - Dryad Data Repository
UR - http://dx.doi.org/10.5061/dryad.r460n.2
DO - doi:10.5061/dryad.r460n.2
ER -
```

BibTex Representation

<http://dev.datadryad.org/resource/doi:10.5061/dryad.r460n.2/citation/bib>

```
@misc{dryad_r460n_2,  
  title = {Data from: Testing embargo},  
  author = {Feinstein, E()},  
  year = {2012},  
  journal = {Testing journal},  
  URL = {http://dx.doi.org/10.5061/dryad.r460n.2},  
  doi = {doi:10.5061/dryad.r460n.2},  
  publisher = {Dryad Digital Repository}  
}
```