

Item Level Versioning


- 1 [What is Item Level Versioning?](#)
- 2 [Important warnings - read before enabling](#)
- 3 [Enabling Item Level Versioning](#)
 - 3.1 [Steps for XML UI](#)
 - 3.2 [Steps for JSP UI](#)
- 4 [Initial Requirements](#)
- 5 [User Interface](#)
 - 5.1 [General behaviour: Linear Versioning](#)
 - 5.2 [Creating a new version of an item](#)
 - 5.3 [View the history and older versions of an item](#)
- 6 [Architecture](#)
 - 6.1 [Versioning model](#)
- 7 [Configuration](#)
 - 7.1 [Versioning Service Override](#)
 - 7.2 [Identifier Service Override](#)
 - 7.3 [Version History Visibility](#)
 - 7.4 [Allowing submitters to version their items \(JSPUI only\)](#)
- 8 [Identified Challenges & Known Issues in DSpace 4.0](#)
 - 8.1 [Only Administrators and Collection/Community Administrators can add new versions](#)
 - 8.2 [Conceptual compatibility with Embargo](#)
 - 8.3 [Conceptual compatibility with Item Level Statistics](#)
 - 8.4 [Exposing version history](#)
 - 8.4.1 [Hide Submitter details in version table](#)
- 9 [Credits](#)

What is Item Level Versioning?


Versioning is a new functionality to build the history of an item. Users will have the opportunity to create new version of an existing item any time they will make a change.

Important warnings - read before enabling

AIP Backup & Restore functionality only works with the Latest Version of Items

 If you are using the [AIP Backup and Restore](#) functionality to backup / restore / migrate DSpace Content, you must be aware that the "Item Level Versioning" feature is **not yet compatible** with AIP Backup & Restore. **Using them together may result in accidental data loss.** Currently the AIPs that DSpace generates only store the *latest version* of an Item. Therefore, past versions of Items will always be lost when you perform a restore / replace using AIP tools. See [DS-1382](#).

DSpace 6 changed the way Handles are created for versioned Items

 With version 6 the way DSpace creates Handles for versioned Items was changed. If you want to keep the old behavior of DSpace 4 and 5 you have to enable the `VersionedHandleIdentifierProviderWithCanonicalHandles` in the XML configuration files `[dspace]/config/spring/api/identifier-service.xml`. See [IdentifierServiceOverride](#) below for details and the comments in the configuration file.

Enabling Item Level Versioning

By default, Item Level Versioning is disabled in DSpace 3, 4, 5 and 6.

Starting from DSpace 4.0, Item Level Versioning is also supported in JSPUI.

Steps for XML UI

If you wish to enable this feature, you just have to uncomment the "Versioning" aspect in your `[dspace]/config/xmlui.xconf` file (and restart your servlet container):

```
<!-- =====
Item Level Versioning
===== -->
<!-- To enable Item Level Versioning features, uncomment this aspect. -->
<aspect name="Versioning Aspect" path="resource://aspects/Versioning/" />
```

Steps for JSP UI

If you wish to enable this feature, you just have to edit the `versioning.enabled` settings in your `[dspace]/config/modules/versioning.cfg` file. Alternatively, you may override it in your local.cfg config (see [Configuration Reference](#)).

```
#-----#
#----- VERSIONING CONFIGURATIONS -----#
#-----#
# These configs are used by the versioning system #
#-----#
#Parameter 'enabled' is used only by JSPUI
versioning.enabled=false
```

Initial Requirements

The Item Level Versioning implementation in DSpace 3.0 builds on following requirements identified by the stakeholders who supported this contribution: [Initial Requirements Analysis](#)

1. What should be *Versionable*
 - a. Versioning happens at the level of an Individual Item
 - b. Versioning 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
 - b. Previous versions of Items should continue to be visible, citable and accessible
 - c. The Bitstreams for previous versions are retained. If something was once retrievable, it should always be retrievable.
3. Identifiers
 - a. Each version of an Item is represented by a separate "*versioned*" identifier
 - 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. 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.
 - b. To submitters, collection manager, administrators will be given to option to create new version of an item.
 - c. Rights to access a specific Item should transmute as well to previous versions
 - d. 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 and AIP packaging and be maintained in updates and restorations.

User Interface

General behaviour: Linear Versioning

From the user interface, DSpace offers **linear** versioning. As opposed to hierarchical versioning, linear version has following properties:

- A new version can only be created started from the latest available version
- When new version has been created and still needs to pass certain steps of the workflow, it is temporarily impossible to create another new version until the workflow steps are finished and the new version has replaced the previous one.

Creating a new version of an item

Administrators and collection/community administrators can create new versions of an item from the Item View page.

1. Click "Create a new version" from the Context Menu in the navigation bar.
2. Provide the reason for creating a new version that will lateron be stored and displayed in the version summary.

Create new version of item: 123456789/127

Reason for creating new version:

Version
Cancel

3. Your new version is now creates as a new Item in your Workspace. It requires you to go through the submission and workflow steps like you would do for a normal, new submission to the collection. The rationale behind this is that if you are adding new files or metadata, you will also need to accept the license for them. In addition to this, the versioning functionality does not bypass any quality control embedded in the workflow steps.

Item submission

My Test Item

Doe, Jane

Date: 2012-11-21

Abstract:

This is the test abstract for my new versioned item. When I resume this, I go through the submission steps and the workflow before my new version becomes archived in the repository.

Files in this item

Files	Size	Format	View
There are no files associated with this item.			

The following license files are associated with this item:

- [Creative Commons](#)

Show full item record

Resume
Cancel

After the submission steps and the execution of subsequent workflow steps, the new version becomes available in the repository.

View the history and older versions of an item

An overview of the version history, including links to older versions of an item, is available at the bottom of an Item View page. The repository administrator can decide whether the version history should be available to all users or restricted to administrators. Since DSpace 6 the repository administrator can decide whether all users should be able to see the version submitter/editor or if this information is restricted and can be seen by administrators only. As this may expose data that may be considered personal (name and email address of the submitter), we encourage everyone to leave the default setting and reveal those information to administrators only.

Version History

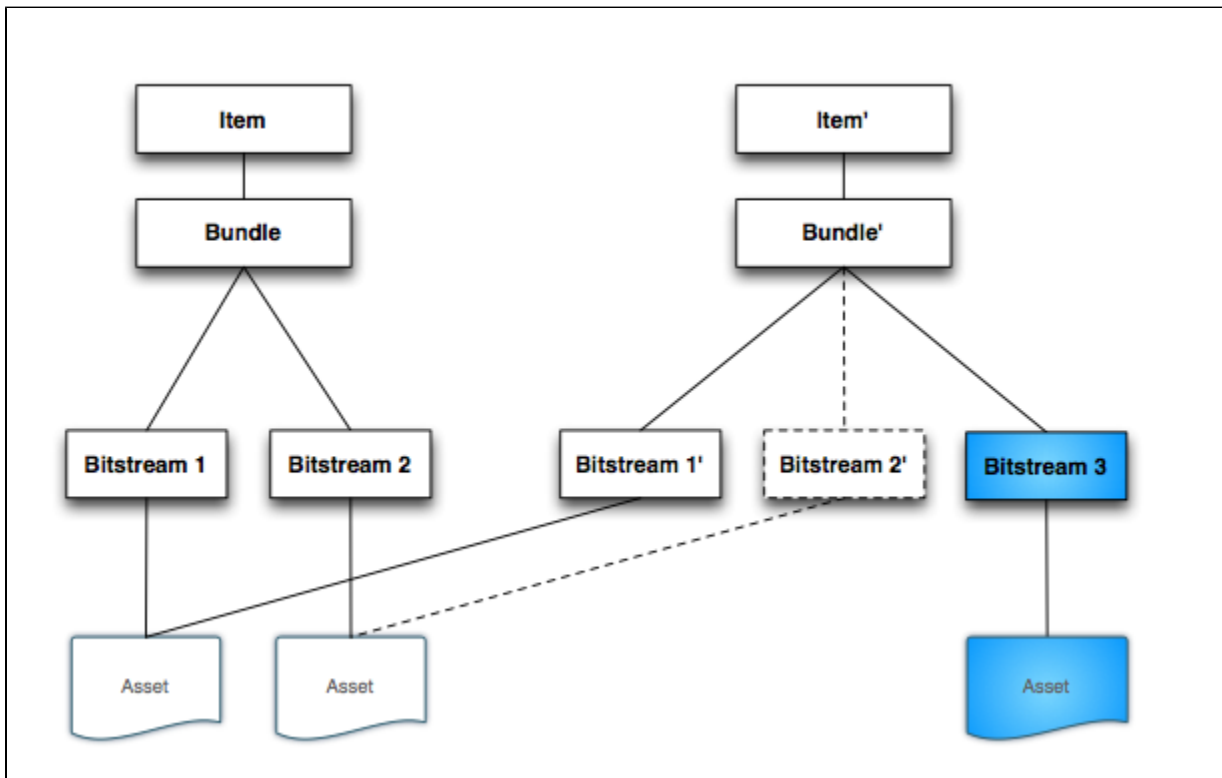
Version	Item	Editor	Date	Summary
2	10673/138*	Demo Administrator	2012-10-23T12:11:46Z	Second version of this item - nothing actually changed
1	10673/138.1	Demo Administrator	2012-10-23T12:10:04Z	

*Selected version

Architecture

Versioning model

For every new Version a separate DSpace Item will be created that replicates the metadata, bundle and bitstream records. The bitstream records will point to the same file on the disk.



The *Cleanup* method has been modified to retain the file if another Bitstream record point to it (the dotted lines in the diagram represent a bitstream deleted in the new version), in other words the file will be deleted only if the Bitstream record processed is the only one to point to the file (*count(INTERNAL_ID)=1*).

Configuration

Versioning Service Override

You can override the default behaviour of the Versioning Service using following XML configuration file, deployed under your dspace installation directory:

[\[dspace_installation_dir\]/config/spring/api/versioning-service.xml](#)

In this file, you can specify which metadata fields are automatically "reset" (i.e. cleared out) during the creation of a new item version. By default, all metadata values (and bitstreams) are copied over to the newly created version, with the exception of **dc.date.accessioned** and **dc.description.provenance**. You may specify additional metadata fields to reset by adding them to the "ignoredMetadataFields" property in the "versioning-service.xml" file:

```
<!-- Default Item Versioning Provider, defines behavior for replicating
     Item, Metadata, Bundles and Bitstreams. Autowired at this time. -->
<bean class="org.dspace.versioning.DefaultItemVersionProvider">
  <property name="ignoredMetadataFields">
    <set>
      <value>dc.date.accessioned</value>
      <value>dc.description.provenance</value>
    </set>
  </property>
</bean>
```

Identifier Service Override

Persistent Identifiers are used to address Items within DSpace. The handle system is deeply integrated within DSpace, but since version 4 DSpace can also mint DOIs. DSpace 4 and 5 supported one type of versioned handle: The initial version of an Item got a handle, e.g. 10673/100. This handle was called the canonical one. When a newer version was created, the canonical handle was moved to identify the newest version. The previously newest version got a new handle build out of the canonical handle extended by a dot and the version number. In the image below you see a version history of an item using this handle strategy.

Version History			
Version	Item	Date	Summary
3	123456789/15*	2016-05-11 16:54:19.188	test
2	123456789/15.2	2016-05-11 16:53:57.92	test
1	123456789/15.1	2016-05-11 16:53:44.0	
* Selected version			

The canonical handle will always point to the newest version of an Item. This makes sense if you hide the version history. Normal users won't be able to find older versions and will always see just the newest one. Please keep in mind, that older versions can be accessed by "guessing" the versioned Handle if you do not remove the read policies manually. The downside of this identifier strategy is that there is no permanent handle to cite the currently newest version, as it will get a new Handle when a newer version is created.

With DSpace 6, versioned DOIs (using DataCite as DOI registration agency) were added and the default versioned Handle strategy was changed. Starting with DSpace 6, `VersionedHandleIdentifierProvider` creates a handle for the first version of an item. Every newer version gets the same handle extended by a dot and the version number. To stay with the example above, the first version of an Item gets the Handle 10673/100, the second version 10673/100.2, the third version 10673/100.3 and so on. This strategy has the downside that there is no handle always pointing to the newest version. But each version gets an identifier that can be use to cite exactly that version. If page numbers change in newer editions, the old citations stay valid. This strategy makes sense, especially if you present the version history to all users. In the image below you see a version history using this strategy.

Versionshistorie			
Version	Ressource	Datum	Zusammenfassung
3	10673/181.3	2016-05-11 14:38:39.161	test
2	10673/181.2*	2016-05-11 14:37:30.074	Test
1	10673/181	2016-04-29 20:41:03.0	
* Ausgewählte Version			

In DSpace 4 and 5, only the strategy using canonical handles (one handle that always points to the newest version) was implemented. In DSpace 6 the strategy of creating a new handle for each version was implemented and became the default. The strategy using the canonical handle still exists in DSpace but you have to enable `VersionedHandleIdentifierWithCanonicalHandles` in the file `[dspace]/config/spring/api/identifier-service.xml`. With DSpace 6, versioned DOIs were introduced using the strategy that every new version gets a new DOI (extended by a dot and the version numbers for versions ≥ 2). To use versioned DOIs, you have to enable DOIs, use DataCite as the registration agency, and enable `VersionedDOIIdentifierProvider` in the named configuration file.

You can configure which persistent identifiers should be used by editing following XML configuration file, deployed under your dspace installation directory:

[\[dspace_installation_dir\]/config/spring/api/identifier-service.xml](#)

No changes to this file are required to enable Versioning. This file is currently only relevant if you want to keep the identifier strategy from DSpace 4 and 5 or if you want to enable [DOIs](#) or even versioned DOIs.

Version History Visibility

Version History

Version	Item	Editor	Date	Summary
2	10673/138*	Demo Administrator	2012-10-23T12:11:46Z	Second version of this item - nothing actually changed
1	10673/138.1	Demo Administrator	2012-10-23T12:10:04Z	

*Selected version

By default, **all** users will be able to see the version history. To ensure that only administrators can see the Version History, enable `versioning.item.history.view.admin` in the `[dspace]/config/modules/versioning.cfg` OR in your `local.cfg` file.

```
versioning.item.history.view.admin=false
```

Allowing submitters to version their items (JSPUI only)

With DSpace 6.0 it became possible to allow submitters to create new versions of their own items. This currently works in JSPUI only and is switched off by default to keep the same behavior as XMLUI. The new versions are going through the normal workflow process if the collection is configured this way. To allow submitters to create new versions of Item they originally submitted, you have to change the configuration property `versioning.submitterCanCreateNewVersion` and set it to `true`. It is part of the configuration file `[dspace]/config/modules/versioning.cfg` but can be overridden in your `local.cfg` too.

Identified Challenges & Known Issues in DSpace 4.0

Item Level Versioning has a substantial conceptual impact on many DSpace features. Therefore it has been accepted into DSpace 3.0 as an optional feature and it is still an option feature in DSpace 4.0. Following challenges have been identified in the current implementation. As an early adopter of the Item Level Versioning feature, your input is greatly appreciated on any of these.

Only Administrators and Collection/Community Administrators can add new versions

There is currently no configuration option to allow submitters to create new versions of an item. This functionality is restricted to Administrators and Collection/Community Administrators. In a context where original submission of DSpace items is done by non-administrator users, it might also make sense to allow them to create new versions. Especially given the fact that new versions have to pass through the workflow anyway.

Conceptual compatibility with Embargo

Lifting an embargo currently does not interact with Item Level Versioning. Additional implementation would be required to ensure that lifting an embargo actually creates a new version of the item.

Conceptual compatibility with Item Level Statistics

Both on the level of pageviews and downloads, different versions of an item are treated as different items. As a result, an end user will have the impression that the stats are being "reset" once a new version is installed, because the previous downloads and pageviews are allocated to the previous version.

One possible solution would be to present an end user with aggregated statistics across all viewers, and give administrators the possibility to view statistics per version.

Exposing version history

The version history is added on the bottom of a versioned item page. A repository administrator can either decide to show this to all users, or restrict it to admins only. If it is shown to admins only, an end user will have no way to find the way to an older version. Since DSpace 6 you can also configure if the submitter's name and email address should be part of the version history or if they should be hidden. To show the submitter might actually be useful if the editor account is always a generic institutional email address, but may conflict with local privacy laws if any personal details are included.

Version History				
Version	Item	Editor	Date	Summary
2	10673/138*	Demo Administrator	2012-10-23T12:11:46Z	Second version of this item - nothing actually changed
1	10673/138.1	Demo Administrator	2012-10-23T12:10:04Z	

*Selected version

Hide Submitter details in version table

In either the `[dspace]/config/modules/versioning.cfg` configuration file or your `local.cfg`, you can customize the configuration option `versioning.item.history.include.submitter`. By default this is set to `false`, which means that information about the submitter is only viewable by administrators. If you want to expose the submitters information to everyone (which be useful if all submitters uses generic institutional email addresses, but may conflict with local privacy laws if personal details are included) you can set this configuration property to `true`.

```
# The property item.history.include.submitter controls whether the name of
# the submitter of a version should be included in the version history of
# an item.
versioning.item.history.include.submitter=false
```

Credits

The initial contribution of Item Level Versioning to DSpace 3.0 was implemented by [@mire](#) with kind support from:

- [MBLWHOI Library](#)
- [Woods Hole Oceanographic Institution](#)
- [Marine Biology Laboratory, Center for Library and Informatics, History and Philosophy of Science program](#)
- [Arizona State University, Center for Biology and Society](#)
- [Dryad](#)

The JSPUI compatibility has been added in DSpace 4.0 by [CINECA](#)