**Research Data-Alliance Data Compilation of Versioning Use Cases**

# Foreword

# Web Sources

## W3C Data on the Web Best Practices

<https://www.w3.org/TR/dwbp/#dataVersioning>

Datasets published on the Web may change over time. Some datasets are updated on a scheduled basis, and other datasets are changed as improvements in collecting the data make updates worthwhile. In order to deal with these changes, new versions of a dataset may be created. Unfortunately, there is no consensus about when changes to a dataset should cause it to be considered a different dataset altogether rather than a new version. In the following, we present some scenarios where most publishers would agree that the revision should be considered a new version of the existing dataset.

## 

## W3C Dataset Exchange Use Cases and Requirements

<https://w3c.github.io/dxwg/ucr/#ID4>

Most datasets that are maintained long-term and evolve over time have distributions of multiple versions. However, the current DCAT model does not cover versioning with sufficient details. Being able to publish dataset version information in a standard way will help both producers publishing their data on data catalogues or archiving data and dataset consumers who want to discover new versions of a given dataset, etc. We can also see some similarities with software versioning and dataset versioning, for instance, some data projects release daily dataset distributions, major/minor releases etc. Probably, we can use some of the lessons learned from software versioning. There are several existing dataset description models that extend DCAT to provide versioning information, for example, HCLS Community Profile.

### Links:

* <https://www.w3.org/TR/hcls-dataset/#datasetdescriptionlevels>
* <https://www.w3.org/TR/dwbp/#dataVersioning>
* <https://www.w3.org/TR/dwbp-ucr/#R-DataVersion>
* <http://db.csail.mit.edu/pubs/datahubcidr.pdf>
* <https://lists.w3.org/Archives/Public/public-dxwg-wg/2017Jun/thread.html#msg6>
* <https://github.com/w3c/dxwg/issues?q=label%3Aversion>

### Related use cases:

* [5.32 Relationships between Datasets [ID32]](https://w3c.github.io/dxwg/ucr/#ID32)

### Related requirements:

* [6.5 Define version](https://w3c.github.io/dxwg/ucr/#RID5)
* [6.6 Version identifiers](https://w3c.github.io/dxwg/ucr/#RID6)
* [6.7 Version release dates](https://w3c.github.io/dxwg/ucr/#RID7)
* [6.8 Version changes](https://w3c.github.io/dxwg/ucr/#RID8)
* [6.9 Version discovery](https://w3c.github.io/dxwg/ucr/#RID9)

## Wikipedia Page on Software Versioning

<https://en.wikipedia.org/wiki/Software_versioning>

**Software versioning** is the process of assigning either unique *version names* or unique *version numbers* to unique states of [computer software](https://en.wikipedia.org/wiki/Computer_software). Within a given version number category (major, minor), these numbers are generally assigned in increasing order and correspond to new developments in the software. At a fine-grained level, [revision control](https://en.wikipedia.org/wiki/Revision_control) is often used for keeping track of incrementally different versions of electronic information, whether or not this information is computer software.

Modern computer software is often tracked using two different software versioning schemes—an [internal version number](https://en.wikipedia.org/wiki/Software_versioning#Internal_version_numbers) that may be incremented many times in a single day, such as a [revision control](https://en.wikipedia.org/wiki/Revision_control) number, and a *released version* that typically changes far less often, such as *semantic versioning*[[1]](https://en.wikipedia.org/wiki/Software_versioning#cite_note-semver-1) or a [project code name](https://en.wikipedia.org/wiki/Code_name#Project_code_name).

## 

## Semantic Versioning

<http://semver.org/>

In the world of software management there exists a dread place called “dependency hell.” The bigger your system grows and the more packages you integrate into your software, the more likely you are to find yourself, one day, in this pit of despair.

In systems with many dependencies, releasing new package versions can quickly become a nightmare. If the dependency specifications are too tight, you are in danger of version lock (the inability to upgrade a package without having to release new versions of every dependent package). If dependencies are specified too loosely, you will inevitably be bitten by version promiscuity (assuming compatibility with more future versions than is reasonable). Dependency hell is where you are when version lock and/or version promiscuity prevent you from easily and safely moving your project forward.

As a solution to this problem, I propose a simple set of rules and requirements that dictate how version numbers are assigned and incremented. These rules are based on but not necessarily limited to pre-existing widespread common practices in use in both closed and open-source software. For this system to work, you first need to declare a public API. This may consist of documentation or be enforced by the code itself. Regardless, it is important that this API be clear and precise. Once you identify your public API, you communicate changes to it with specific increments to your version number. Consider a version format of X.Y.Z (Major.Minor.Patch). Bug fixes not affecting the API increment the patch version, backwards compatible API additions/changes increment the minor version, and backwards incompatible API changes increment the major version.

I call this system “Semantic Versioning.” Under this scheme, version numbers and the way they change convey meaning about the underlying code and what has been modified from one version to the next.

## Recommended Practice for Statisticians

Bryan, J. (2018). Excuse Me, Do You Have a Moment to Talk About Version Control? *The American Statistician*, 72(1), 20–27. https://doi.org/10.1080/00031305.2017.1399928

Data analysis, statistical research, and teaching statistics have at least one thing in common: these activities all produce many files! There are data files, source code, figures, tables, prepared reports, and much more. Most of these files evolve over the course of a project and often need to be shared with others, for reading or edits, as a project unfolds. Without explicit and structured management, project organization can easily descend into chaos, taking time away from the primary work and reducing the quality of the final product. This unhappy result can be avoided by repurposing tools and workflows from the software development world, namely, distributed version control. This article describes the use of the version control system Git and the hosting site GitHub for statistical and data scientific workflows. Special attention is given to projects that use the statistical language R and, optionally, R Markdown documents. Supplementary materials include an annotated set of links to step-by-step tutorials, real world examples, and other useful learning resources. Supplementary materials for this article are available online.

## Git Workflows

Almost all programming projects work with some kind of version control. When I started to work with Git, I used the tool also directly for my private projects. But especially at the beginning I found it hard to structure my commits and branches in a practical way. For this reason I would like to show you some common strategies today, the so-called Git Workflows.

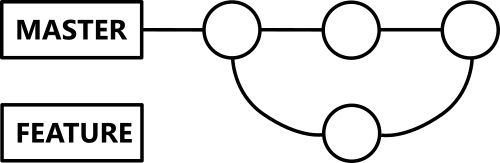
### Simple Workflow

The simple workflow consists of a single master branch. There is only this one branch to which changes are pushed. This workflow is only suitable for very small projects, e.g. private ones, where only you work on yourself. As the team grows, this workflow becomes very messy and you’ re going to have to deal with a lot of merge conflicts.



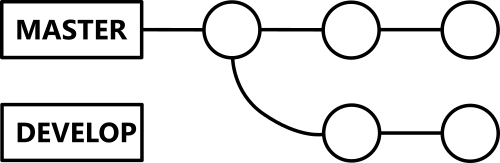
### Feature Branches

This second level adds feature branches to the simple workflow. These branches are used to develop new functionalities separately from the rest of the project. After a feature is completed, the branch is merged. Unlike the master branch, the feature branches are therefore short-lived and only exist until their merge. Depending on their complexity, feature branches can often be further subdivided. Just make sure you don’t exaggerate, which could again affect the overall structure.



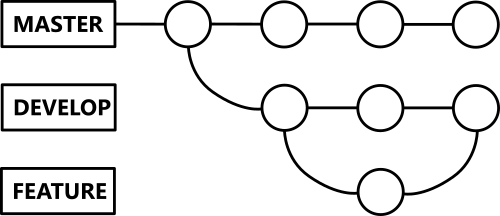
### Developer Branch

With the Developer Branch, a second, long-lived branch is created next to the Master Branch. This is the only place where development takes place, so that the master branch always remains in a release-ready state. Here, however, similar problems arise as with the simple workflow, which is why it should only be used for very small teams.



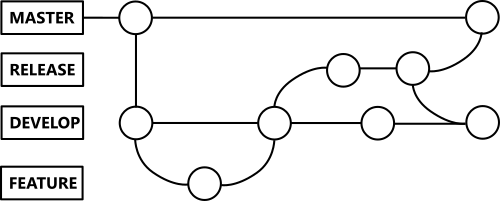
### Developer and Feature Branches

The previous two strategies can be combined very well. Again, the master branch must always be ready for release, feature branches are only ever merged with the developer branch. After successful testing of the functionalities on the developer branch, this branch is merged to master, which then can be released.



### Release Branches

This extension of the developer and feature branch workflow is often used for large projects that are planning frequent releases. For a new release, a new release branch is created from the developer branch. This only is used for final bug fixes, no new features are developed here. As soon as the release can be shipped, the branch will be merged into both the master and the developer branch. The fixes in the release branches allow other teams to work on new features without disturbing the work on the release.  
  
The model is often complemented by another branch: the hotfix branch which allows direct bug fixing from the master branch.



### But which concept is right for me?

Basically, the more complex your project, the more complex the workflow should be. But also for one-man projects it often makes sense not to use the simple workflow and to use a branching strategy already here. For my own projects, for example, I currently use the Developer Branch concept. But whatever you decide to do: Make sure you have a consistent naming strategy for branches (and commits, of course) and you’re very good to go.

*From Marcel Jurtz “A Software Developer’s Blog”, with permission.* [*https://blog.mjurtz.com/2018/09/git-workflows/*](https://blog.mjurtz.com/2018/09/git-workflows/)

# RDA Sources

## RDA Data Citation Recommendation

<https://www.rd-alliance.org/group/data-citation-wg/outcomes/data-citation-recommendation.html>

Digitally driven research is dependent on quickly evolving technology. As a result, many existing tools and collections of data were not developed with a focus on long term sustainability. Researchers strive for fast results and promotion of those results, but without a consistent and long term record of the validation of their data, evaluation and verification of research experiments and business processes is not possible.

There is a strong need for data identification and citation mechanisms that identify arbitrary subsets of large data sets with precision in a machine-actionable way. These mechanisms need to be user-friendly, transparent, machine-actionable, scalable and applicable to various static and dynamic data types. As changes to data can affect anything ranging from an individual value to entire subsets of data and happen at any time interval, ranging from milliseconds to annual batch updates, we would like to have a single mechanism that applies to all kinds of data, data representations, and amounts of changes. Imposing any kind of semantic structure on such versions turns out to be difficult and not applicable generally and across different data sources. For example, the same update to data that does not impact its interpretation for a specific use case might lead to a different interpretation on another use case (such as exact reproducibility), making the difference between a major and minor version number update confusing. It also violates the principle of not embedding semantics in an identifier. Any semantic interpretation of the impact of updates should thus be performed separately as provenance metadata (update documentation) and not be included in any identifier creation.

The RDA Recommendations on data citation thus recommend not applying version numbers to entire data sets (and, specifically, not to pre-defined subsets of such data), but to (1) version and timestamp individual updates to data items on an element/record level (i.e. marking each addition of a record with a timestamp when it became available in the data set, marking deleted records as deleted with the according timestamp, and marking updates to values as deleted and re-inserted with the new value at a specific timestamp); and to (2) assign identifiers to timestamped queries which allow to retrieve the specific subsets at any given point in time. Instead of discrete version numbers, a version of a dataset thus is indicated by the status of the data set at a given point in time. This allows any state of a data set to be retrieved, and allows the current version of any data set to be used at any point in time. The principle is applicable to all types of data, ranging from numeric data to software code or document editing systems, with versioning systems allowing to retrieve the state of any code document as it existed at a specific point in time). Optimizations specifically for high-frequency updates to data may include not maintaining/keeping the update states of the dataset that were never read/accessed, i.e. states that were never observed.

## RDA Data Foundations and Terminology IG

<https://smw-rda.esc.rzg.mpg.de/index.php?title=Versioning>

Definition: Generate a (changed) copy of a data object that is uniquely labeled with a version number. The intent is to enable access to prior versions.

Explanation: Note that a version is different from a backup copy, which is typically a copy made at a specific point in time, or a replica, which is a copy of a data object that can be periodically updated.

Related term – version, replication

Example:

Scope: RDA Term Collection Core

# Use cases

## 1. da|ra Registration agency for social and economic data

da|ra (Registration agency for social and economic data) provides recommendations on versioning:

<https://www.da-ra.de/fileadmin/media/da-ra.de/PDFs/TechnicalReport_2014-18.pdf>

In general the following aspects should be considered regarding versioning (p.16)

* An object with an assigned DOI name should not be changed.
* Each change must be saved as a new version and a new DOI name must be assigned.
* The publication agent is responsible for versioning.

The GESIS Leibniz Institute for the Social Sciences ([www.gesis.org](http://www.gesis.org)) is compliant with this recommendation and is using a three-digit-versioning.

Major.Minor.Revision

Major number starts with „1“, Minor and Revision number start with „0“ separate with „.“

First version of a data file is „1.0.0“.

1. Increase of the first digit if new data is added (e. g. waves, samples etc.)
2. Change of the second digit if corrections are made, which influence the analysis (e. g. change of values of respondents)
3. If the documentation is changed or amended (typing error or more detailed text added etc.) only the third digit will be increased

This versioning is based on the recommendations of the Data Documentation Initiative (DDI). DDI-Lifecycle 3.2

<http://www.ddialliance.org/Specification/DDI-Lifecycle/3.2/drafts/IVMR_DRAFT.pdf> ; page 2/3 “Versioning”

In the GESIS Data catalogue (DBK) the versioning and corresponding errata are documented. Description (in German only) please see here:

<http://www.gesis.org/fileadmin/upload/forschung/publikationen/gesis_reihen/gesis_methodenberichte/2012/TechnicalReport_2012-01.pdf>

page 13/14

## 2. DIACHRON project

<https://cordis.europa.eu/project/rcn/108537_en.html>

DIACHRON is an integration project that addresses certain issues arising from the evolution of the data such as:

* Detect the changes of that happen to datasets (tracking the evolution)
* Archive multiple versions of data and cite them accordingly to make the reference of previous data feasible (archiving and citation)
* Retrieve and query previous versions (time traveling queries)
* Validate and repair various data deficiencies (curation problem)
* Identify the cause of the evolution of the datasets in respect with the real world evolution of the entities the datasets describe (provenance problem)
* Provide various quality metrics so as to enable quality assessment of the harvested datasets and determination of the datasets versions that need to be preserved (appraisal)

The DIACHRON solution aims not only to store previous versions for preservation in case of future need of them, but to create a live repository of the data that captures and highlights data evolution by keeping all data (current and previous) accessible, combined with a toolset that handles the full life cycle of the Data Web.

## 3. United States Geological Survey (USGS, Draft Policy) (Leslie Hsu)

<https://www2.usgs.gov/datamanagement/share/dataversioning.php>

### Guidance on Documenting Revisions to Data Releases:

This guidance describes a formal revision process for datasets and associated metadata that have been released as an information product and require change.

Not covered in this guidance are USGS approved databases or Web data services for data that are expected to change continuously or on a schedule, with additions and updates made over time. Examples of these systems or services include NWIS, USA-NPN, and BISON. These data products have processes in place for data quality evaluation prior to data being loaded.

The guidance lays out three levels of revision:

* A Level 1 revision does not change the data itself (but to metadata and landing page, and something like misspelling of a data header etc.)
* A Level 2 revision creates a new version of the data release that will normally be used instead of the previous version, except for purposes of repeating analyses that used the uncorrected data. The changes for a Level 2 revision, however, affect only a small number of data values.
* For a Level 3 revision, the data are significantly and substantially changed

Notes that only Level 2 & 3 revision leads to a new version. Version number follows the format “major.minor”, each corresponding to the level 3 and level 2 revision.

Revision of a data release is warranted, for example, when an error is detected and needs to be corrected (deleted, changed) for future use of the data. When correcting data errors, changes are made to the data only where needed, but no other alterations are made to either the structure or content. Another example case for revision is the release of data in stages in order to meet project timelines, so that the amount of data provided in an information product increases through subsequent versions.

If, in the revision, data are corrected or added, the data release must again be reviewed for quality and accuracy, and the modifications must be documented as described below. For substantial or major revisions (defined below, in the Version Numbering section), the review process should be documented in the Information Product Data System (IPDS).

The revision process is described below for the following four cases:

1. Correcting an error

2. Appending new data

3. Extending the data structure

4. Archiving deprecated data

Following these cases, guidance is provided for assigning revision numbers.

### Correcting an Error:

If an error is found that is not in the data itself, such as a misspelling in a data header or a site location name, replace or update the erroneous file and update the metadata record and any additional documentation to reflect the update.

If an error is found in the data, the author should correct the data release. If the error is large enough to affect outcomes of future data use, create a new data release Mrecord in the IPDS. A new version of the corrected data should note that the revised version (as opposed to previous versions) is current. The landing page should describe the error and point users to the new version of the data. The original or previous version of the data should also be preserved in case it is needed to understand previous uses, and in accordance with records management disposition schedules and litigation holds requirements. The revision process will result in a new IPDS record, an updated metadata record, updates to the online documentation including a revision history, and a new incremental version number (e.g., version 1.1, refer to “Version numbering” below.). All review requirements apply to the new version. Once revised data are released, the citation should reflect the new incremental version of the data as shown in the example citations below.

If the error could affect existing USGS scientific conclusions, consult your local Bureau Approving Official in the Office of Science Quality and Integrity (<https://internal.usgs.gov/fsp/toolbox/approvingofficials.html>) for guidance.

**Examples of the citation change on data release landing page:**

Original citation:

Klunk, O.T., 2012, Bathymetry of the Bermuda Triangle: U.S. Geological Survey data release, https://doi.org/10.5066/XXXXXXXX.

Revised citations:

Klunk, O.T., 2012, Bathymetry of the Bermuda Triangle (ver. 1.1, July 2012): U.S. Geological Survey data release, https://doi.org/10.5066/XXXXXXXX.

Klunk, O.T., 2013, Bathymetry of the Bermuda Triangle (ver. 2, May 2013): U.S. Geological Survey data release, https://doi.org/10.5066/XXXXXXXX.

Note that the title and DOI do not change but that the citation changes by adding version information. Additionally, it is possible that the year of the publication will also change.

**On the landing page of the data release, include text reflecting the revision.**

First release: 2012

Revised: July 2012 (ver. 1.1)

Revised: May 2013 (ver. 2.0)

**Revision history:**

Additionally, there should be a revision history text file available that explains exactly what changed in each revision. (See the revision history file in the example provided below for appending new data.)

### Appending New Data:

Addition of data to released datasets, such as updating a project’s data release with data from a new time period, place, or new field activity, requires most of the same steps as an original data release. In addition to the inclusion of new data, errors in previously released data may also be corrected. The following are required when new data is added: a new IPDS record, updated citation, updated metadata record, updated revision history, and text on the landing page reflecting the new version.

**NOTE:** The new IPDS data release record is used to ensure requirements of [SM 502.7](https://staging-www.usgs.gov/usgs-manual/500/502-7.html) and [SM 502.8](https://www2.usgs.gov/usgs-manual/500/502-8.html) have been met. No new digital object identifier should be created. That is, use the existing DOI for the revised data release.

For an example, see Pendleton, E.A., Ackerman, S.D., Baldwin, W.E., Danforth, W.W., Foster, D.S., Thieler, E.R., and Brothers, L.L., 2016, High-resolution geophysical data collected along the Delmarva Peninsula, 2014, USGS Field Activity 2014-002-FA (ver. 4.0, October 2016): U.S. Geological Survey data release, [https://doi.org/10.5066/F7MW2F60](http://dx.doi.org/10.5066/F7MW2F60)

### Extending the Data Structure

There are cases in which the data structure is modified to allow the inclusion of new data types through the addition of new tables or fields. The extended structure is then considered a new version. These revisions are appropriate for data releases that are stand-alone research products, rather than the data foundations of scientific reports. In this case, the requirements include: a new IPDS record, updated citation, updated metadata record, updated revision history, and text on the landing page reflecting the new version. Changes should reflect a new version of the data release (e.g., version 2.0, refer to “Version numbering” below.).

**NOTE:** The new IPDS data release record is used to ensure requirements of [SM 502.7](https://staging-www.usgs.gov/usgs-manual/500/502-7.html) and [SM 502.8](https://www2.usgs.gov/usgs-manual/500/502-8.html) have been met. No new digital object identifier (DOI) should be created. That is, use the existing digital object identifier for the extended data structure.

### Archiving Deprecated Data:

When data with identified errors are corrected and replaced - for example, as a new incremental version - the version with errors should not be publicly offered, but may be available on request, to future users. Because the errored data may have been used to support conclusions in a publication or a policy decision, there may be future consequences; therefore, it is essential to preserve the original data, for example in a dark archive (an offline location for preservation), with errors intact. The filename and accompanying documentation should make clear that the data are deprecated. This provides a snapshot in time of the data in terms of provenance, while ensuring that they are not recommended for future use. If size constraints make archiving a full copy impractical, some other process should be provided for making the original data available.

### Version numbering:

Version numbers consist of two parts, a major and a minor component, separated by a period. In the example “version 1.2,” the number to the left of the period, “1,” is the major component and represents the number of separate major revisions. The number to the right of the period, “2,” is the minor component and represents the number of separate substantial revisions.

The original release is considered version 1.0.Either the major or the minor component of the version number will be incremented when a revision is released. When a major revision is released, the major component increases by one number and the minor component is reset to zero (0). Substantial revisions (see definition below), regardless of how many, do not trigger a change in the major component of the version number. For example, if the data release was revised on seven separate occasions for substantial revisions, the version number will be 1.7.

**Minor Revisions:**

Minor revisions that are so insignificant that they do not affect the use or interpretation of the data include, but are not limited to: correcting misspelled words in data or metadata; and improvements in presentation of ancillary information on data landing pages. There is no version numbering system for these types of minor revisions and therefore no need to develop a version history document.

Using a ScienceBase (sciencebase.gov) data release page as an example, a minor revision could correct a misspelled word in the title or in the abstract. In another example, the author may wish to revise one of the contacts listed on the landing page. In other words, in a minor revision, the data did not change.

Action: No new version number required.

**Substantial Revisions:**

Substantial revisions are corrections to the data or metadata that are large enough to affect outcomes of future data use. Such errors typically involve missing or incorrect data values, but could also be missing or unclear annotations in table headings or in metadata records. Substantial revisions might also improve the usability or interpretation of the product content such as a modification in a polygon shapefile, slightly shifting a line so that a western boundary is consistent with another polygon shapefile that was just released.

An example of a substantial revision would be correcting a geospatial file in which a small number of negative longitudes were entered as positive numbers. The revision would change the incorrect longitude values to negative numbers. In a substantial revision, some of the data have been changed.

Action: Create a new minor component number (for example, version 1.0 is changed to version 1.1).

**Major Revisions:**

Major revisions include changes in the data structure and updates that add or modify substantial amounts of data. Also included are large corrections to data, for example, correcting a data file in which many data values were consistently incorrect as a result of improper processing.

An example of a major revision would be a new release of a bathymetry grid when an error was detected in the processing step that applied tide corrections. The data themselves have undergone a significant change.

Action: Create a new major component number (for example, version 1.0 is changed to version 2.0).

**4. BCO-DMO** (Danny Kinkaide)

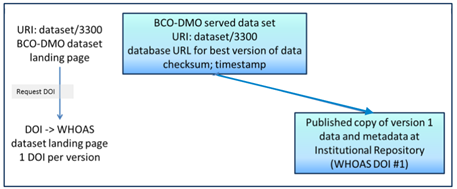
The Biological and Chemical Oceanography Data Management Office ([BCO-DMO](https://www.bco-dmo.org/)) works with investigators to serve data online from research projects funded by the Biological and Chemical Oceanography Sections, the Division of Polar Programs Arctic Sciences and Antarctic Organisms & Ecosystems Program at the U.S. National Science Foundation.

The BCO-DMO system is a data server plus a DSpace archive for data publication where data packages (timestamped, checksummed copy of the data, plus ISO metadata record and supplemental docs) are deposited.

To summarise, BCO-DMO curated data are:

* Served: http://bco-dmo.org (URLs, URIs)
* Published: at an Institutional Repository (WHOAS) http://dx.doi.org/10.1575/1912/4847
* Archived: at NCEI, a US National Data Center http://data.nodc.noaa.gov/cgi-bin/iso?id=gov.noaa.nodc:0078575

Figure 1: BCO-DMO data publication system components

  
Source: Chandler, C. *et al* (2016)

In 2016, BCO-DMO received funding from RD-A to implement the recommendations of the [RD-A Data Citation Working Group](https://www.rd-alliance.org/groups/data-citation-wg.html). The 14 [WG recommendations](https://rd-alliance.org/system/files/documents/RDA-DC-Recommendations_151020.pdf) can be summarised as:

* ensure that data are stored in a versioned and timestamped manner
* identify data sets by storing and assigning persistent identifiers (PIDs) to timestamped queries that can be re-executed against the timestamped data store

The BCO-DMO evaluation of the recommendations found that R1-11 of the WG were a good fit with BCO-DMO architecture, R12 was regarded as doable, and R13-14 were consistent with Linked Data approach to data publication and sharing.

A primary driver for the BCO-DMO to implement the WG recommendations was to support citation of published data. As a result of the RD-A funded project, the following procedure is now invoked when a BCO-DMO data set is updated:

* A copy of the previous version is preserved
* Request a DOI for the new version of data
* Publish data, and create new landing page for new version of data, with new DOI assigned
* BCO-DMO database has links to all versions of the data (archived and published) Both archive and published dataset landing pages have links back to best version of full dataset at BCO-DMO
* BCO-DMO data set landing page displays links to all archived and published versions

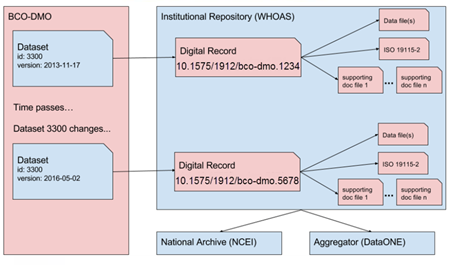
The BCO-DMO identified five use cases for managing dataset DOIs for data versions.

* Use case 1: when new dataset is published with status = final
  + assign a DOI

Noting that changes to the dataset that might result in different conclusions require a new version (timestamped, checksum) and a new DOI

* Use case 2: dataset is modified (columns added or removed)
  + mint/assign a new DOI in this case
  + create a new landing page for the new DOI, and link dc.related old and new one
* Use case 3: routine dataset extension over time (ie active time-series)
  + when adding new time range to dataset, inherit existing DOI
  + data replacement is not permitted, only extension in time
  + metadata temporal range is updated
* Use case 4: update to metadata only (eg typos corrected)
  + handle is appended with .1 in the local repository; DOI does not change
* Use case 5: minor replacement (fixes, adjustments, format) within a dataset (# sig digits)
  + data object modified as needed; small changes
  + internal version control (new version date), update metadata to clearly reflect changes
  + DOI remains the same
  + version 1.0 gets a DOI
  + new version declared if different science result
  + new columns etc., different conclusions – new landing page

Figure 2: BCO-DMO data citation system components

  
Source: Chandler, C. *et al* (2016)

References  
Chandler, C., Shepherd, A., Bassendine, D. (2016) *Adoption of Data Citation Outcomes by BCO-DMO.* Presentation to RDA 8th Plenary Data Versioning BoF meeting, Denver Colorado.

**5. NASA: EOSDIS and SEDAC** (Bob Downs)

The Earth Observing System Data and Information System ([EOSDIS](https://earthdata.nasa.gov/about)) is a core capability in NASA’s Earth Science Data Systems (ESDS) Program. It provides end-to-end capabilities for managing NASA’s Earth science data from various sources – satellites, aircraft, field measurements, and various other programs.

EOSDIS does not use a common standard to number versions

* Diverse data producers (instrument or science teams) contribute data to the NASA Distributed Active Archive Centers (DAACs)
  + Designate versions in their own ways
  + Use various terms: Version 1, 2; Collection 1, 2; Release 1, 2; Edition 1, 2
* Number of versions vary, depending on volumes from instrument
* Absence of standard numbering scheme for versions is not problematic
  + each data set title is identified by a particular version
  + software version that generated data is identified, when applicable
  + provenance is tracked independent of a standard numbering scheme
* Version information is recorded in the data metadata
* New DOIs are assigned as new versions are generated
  + Landing pages reference older versions if they exist
  + Landing pages for superseded versions will persist and refer to newer versions

The Socioeconomic Data and Applications Center ([SEDAC)](http://sedac.ciesin.columbia.edu/) is a data centre within EOSDIS. It has established the following practice for versioning data products

* Establish titles with consistent designations for version or year
  + Uniquely identify each edition of collection, dataset, or product
  + Unique aspects can include year or range of dates for observations
* Default version number is 1; implied if not stated explicitly
  + New version number is assigned if a dataset is changed
  + Subsequent versions are Version 1.01, Version 2, or Revision 01
* New collection version reflects new stage of development
  + Collection versions are assigned as integers
  + New collection supersedes previous collection

PIDS at SEDAC

SEDAC uses the following guidelines for assigning global persistent identifiers:

* assigned to the landing page for each dataset disseminated by SEDAC
* included in the recommended citation for each dataset
* recorded and maintained to identify current location and optimize discovery

Their procedure for assigning global persistent identifiers:

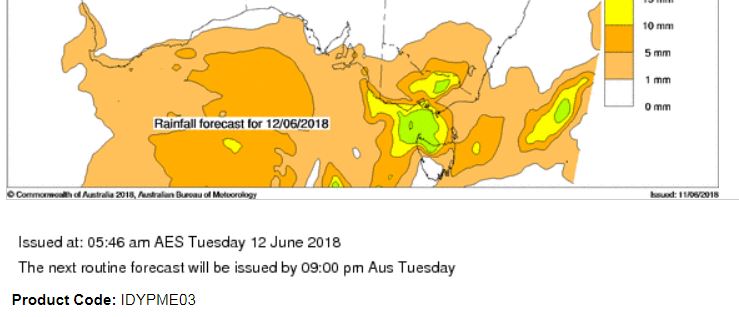
* DOIs assigned to datasets and documentation; software and services may be next
* assigned using EZID and the DataCite Metadata Schema
* DOIs also recorded in the FGDC CSDGM (Federal Geographic Data Committee Content Standard for Digital Geospatial Metadata)
* Related Identifier field is used to link to other data, documentation, publications
* DOI record is modified when location of landing page changes

**6. Australian Bureau of Meteorology** (Martin Schweitzer)

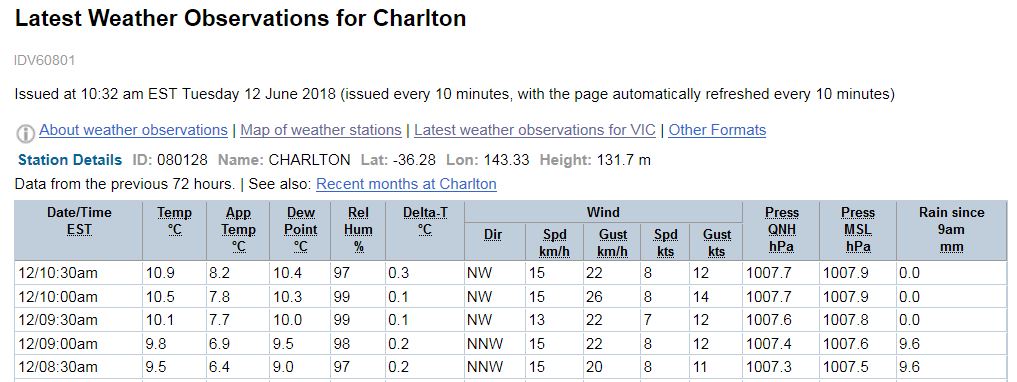
The Australian Bureau of Meteorology (BoM) is Australia's national weather, climate and water agency. BoM collects observational data and produces a lot of downstream data products. There are requirements to go back to the point of time a data product was created, a need to have “version” to be able go back in time, to link product with the version of data (raw data - the source of truth).

The BoM uses a relational database to manage observational data. The database for climate data is called ADAM (Australian Data Archive for Meteorology). ADAM stores observations by daily, hourly and minute. Data comes in two types: point of data and grid of data. Data comes from various sources such as stations - about 600 stations that records data per minute (temperature, rain, humanity, wind, etc. about 12 variables), paper records from farmers - registered farmers send rain gauge to BoM, BoM then add data into DB. Currently, the ADAM DB holds over 120 years of records..

The BoM applies QC consistently for any inconsistency by station and time series etc. All changes are recorded and time-stamped in an audit table. Data products made available from the BoM web portal are identified by Product Code, State Date (date to be forecasted, or time where an observation was made) and Product Issue Date. The combination of the three “IDs” defines a “version”, it can be traced back in time where the data product was reproduced and can be reproduced if required.



Forecast data product



Observational data product

**7. Australian Integrated Marine Observing System (IMOS)**  (Natalia Atkins)

Since 2006, [IMOS](http://imos.org.au/about/) has been routinely operating a wide range of observing equipment throughout Australia’s coastal and open oceans, making all of its data accessible to the marine and climate science community, other stakeholders and users, and international collaborators. There are five major research themes that unify IMOS science plans and related observations:

* Long-term ocean change;
* Climate variability and weather extremes;
* Boundary currents;
* Continental shelf and coastal processes; and
* Ecosystem responses.

Most of IMOS data are dynamic: new data is continuously added, existing data can be both modified or updated. Data are file based, e.g., in NetCDFs, stored in databases, and there are also data types such as AUV images and acoustic recordings. Datasets vary in size from a few 1000 rows in a database to 20TB of satellite data.

Data is quality checked before being sent to IMOS but may be corrected or reprocessed several times. When new versions are published, the previous version is archived (except for satellite data). Therefore it is possible that a disparity may occur between data previously accessed and cited, and the data that is currently available.

For data stored on Amazon S3 (object storage), they use the versioning feature from the S3 object storage to keep all previous versions (except for satellite data). The version is identified by date and time, and if “versioning” is enabled, assigned a randomly generated Version ID (versions are “linked” by virtue of having the same file name). This version information is not publicly viewable, and users have to contact IMOS for access to. For data in netCDFs file format, (change) history is captured with a file.

IMOS advises their data consumer to cite their data as follows:

IMOS [year-of-data-download], [Title], [data-access-URL], accessed [date-of-access].

**8. Australia Astronomy Observatory (AAO) Data Centre** (Simon O’Toole)

The All-Sky Virtual Observatory (ASVO) is a federated system of astronomical data nodes. There are five nodes of the ASVO: the AAO stores optical astronomy data at the AAO; ANU/Mt Stromlo stores images from the SkyMapper survey at NCI; MWA-ASVO stores data from the Murchison Widefield Array (MWA) and CSIRO/CASDA stores data from the ASKAP telescope. The last two are radio telescope pathfinders for the SKA, and their data are stored at the Pawsey Supercomputing Centre. There is also the Theoretical Astrophysics Observatory, which generates and stores numerical simulations at Swinburne University.

Data types: images, spectra, image-spectral data cubes, raw visibility data (radio interferometry)

Data size: ~2 petabytes of optical, >12 petabytes radio, 100s terabytes theory

Data formats/models: FITS, HDF5, PostgreSQL, Hadoop/Spark

Data access: 1) web UI, 2) third-party VO apps, 3) APIs

Data ingest: mostly dynamic, but only released publicly at discrete time points, e.g. yearly observation

The ASVO stores optical imaging data at the National Computational Infrastructure (NCI), and other types of optical data at the AAO. Every version is stored, but only published versions are publicly accessible. DOIs will be attached to data on an ongoing basis soon.

Radio data from ASKAP (Australian Square Kilometre Array Pathfinder) and the MWA are stored at Pawsey, each version is kept. Data process is an ongoing activity, old versions may be rewritten. Improvement of data quality is driving data versioning. New version may come from different calibrations, with improved calibration pipeline.

General approach to versioning AAO:

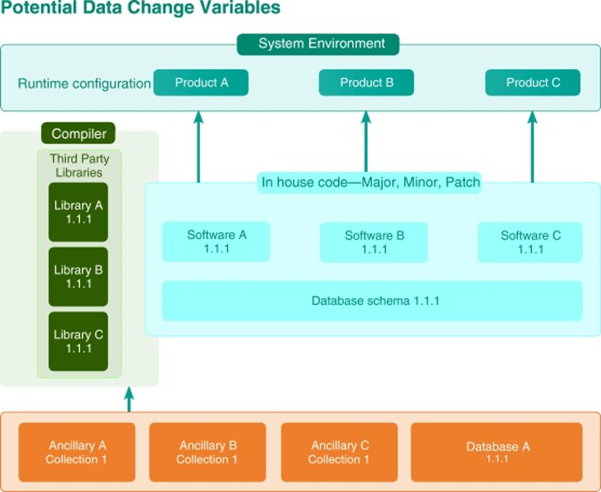
* Data Release: new DOIs for each public data release (manual validation & release process). E.g. \_v01 to first version, \_v02 to 2nd version
* Old versions of data/branches remain available, but the current version is the default

**9. Digital Earth Australia** (Simon Oliver)

[Digital Earth Australia](http://www.ga.gov.au/dea/products) (DEA) is a platform that uses spatial data and images recorded by satellites orbiting the planet to detect physical changes across Australia in unprecedented detail. Using high performance computing power provided by the National Computational Infrastructure and commercial cloud computing platforms, DEA organises and prepares satellite data into stacks of consistent, time-stamped observations that can be quickly manipulated and analysed to provide information about a range of environmental factors such as water availability, crop health and ground cover.

Satellite earth observation data are highly structured and stored as large to very large binary data files, each of which may contain gigabytes or even terabytes of data. The data are diverse and dynamic with new data being continuously added and/or existing data being updated (e.g., as calibration algorithms improve over time, errors are found in existing data, etc.)

Figure 1: Inputs to processing of Earth observation raw archives to produce products.



Source: Lewis, A. *et al* (2017)

Figure 1 shows how multiple ancillary inputs are used which change from time to time as quality is improved. Software code libraries, which may be internal to the processing agency or from third party providers, have specific versions. The hardware and operating system environments are also significant and are frequently upgraded. Finally, products often are chained, with one product being an input to the next, for example as higher levels of correction are produced.

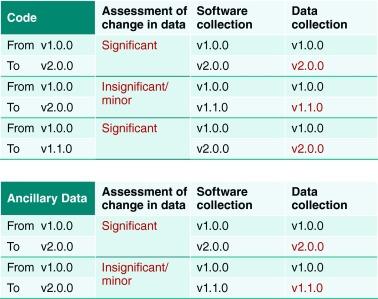
DEA uses the concept of Managed Collections to address these challenges. The approach differs significantly from other current models of processing which often use common systems of differing versions across multiple platforms to produce “like” products.

The concept of a managed collection includes:

* Software versioning, and governed production software upgrades;
* Ancillary input collection versioning and update control;
* Assessment of the scope and significance of a proposed change for the collection. Scope refers to the proportion of the collection affected, whereas significance is the established through comparison with benchmarks and acceptable deviations from these in regard to radiometric and geometric changes, for example;
* Business processes, which determine a course of action based on the scope and significance of the change, for example to:
  + Add to a backlog until significance and scope reach thresholds;
  + Upgrade the collection;
  + Update components of the collection; and
  + Patch components of the collection.

Underpinning the approach is a three level hierarchy major.minor.patch convention in line with the semantic versioning scheme commonly in use in the IT domain. Data collections and their subcomponents are attributed via this convention to enable patch and repair of various components of the collection. The scheme allows for variation within a data collection but enables management practices to enact a virtual self healing methodology. Decisions on the major/minor/patch attribution to a dataset are made via a Change Control Board assessment of the scope and significance of a proposed change to the collection (whether it be to software, systems, data or metadata).

Figure 2: Collection numbering approaches adapted from software versioning. The numbering of versions depends upon the scope and significance of each change.



Source: Lewis, A. *et al* (2017)

References

Lewis, A., Oliver, S., Lymburner, L. et al (2017) *The Australian Geoscience Data Cube : Foundations and lessons learned*. Remote Sensing of Environment, vol 202, pp.276-292. <https://doi.org/10.1016/j.rse.2017.03.015>

**10. Geoscience Australia: Enterprise metadata catalogue** (Martin Capobianco & Andy Marshall)

Geoscience Australia (GA) is Australia's pre-eminent public sector geoscience organisation. GA is the government’s technical advisor on all aspects of geoscience and custodian of the geographical and geological data and knowledge of the nation.

GA manages hundreds of thousands of datasets, collected over many years. Many of the datasets are publicly accessible via the [eCAT catalogue](https://ecat.ga.gov.au/geonetwork/srv/eng/catalog.search#/search?resultType=details&from=1&to=20&sortBy=changeDateDesc). While there is no formal, documented approach to versioning within GA, there are standard approaches that are applied.

Example 1:a database snapshot is captured and published.   
*SHRIMP U-Pb Geochronology Interim Data Release July 2007*

<http://pid.geoscience.gov.au/dataset/ga/65358>

The data published here are a snapshot of the database at the “Ending-Date”, although entry into the database is continuous. The snapshot was created as an interim form of data release while web-based accessibility to the continually updated database was explored.

The Lineage statement in the catalogue record explains: *These data are derived directly from Geoscience Australia's corporate Oracle OZCHRON database for U-Pb ages derived using the SHRIMP method. An ASCII extraction of the database is generated as ASCII comma separated values (CSV)*

The metadata record offers a File Download link to a zip file that includes:

* Copyright txt
* Data dictionary pdf
* Geochron data extract as at 27 July 2007 xls
* Metadata txt

This form of snapshot data release is not commonly practiced within GA.

Example 2**:**  a new metadata record is created for each version. Public access to older versions is retained.

*Index of airborne geophysical surveys*

<http://pid.geoscience.gov.au/dataset/ga/79134> (14th ed)

<http://pid.geoscience.gov.au/dataset/ga/74888> (13th ed)

<http://pid.geoscience.gov.au/dataset/ga/72767> (12th ed)

A new edition or version of the Index is published annually. A new metadata record is created for each new edition. The version is indicated by the edition number.

The Lineage statement in the catalogue record for the 14th edition explains: *This Record is published as the thirteenth in a series of GA Records which contain regularly updated information as the specifications of surveys already completed are incorporated and as new surveys are added to the National Airborne Geophysical Database (ARGUS Oracle database). This version of the Index includes details of surveys completed since the previous edition in January 2013 as well metadata for new surveys.*

The catalogue record for the 14th edition provides 3 links to related products.   
One link leads to this dataset:

[Digital files for the Index of airborne geophysical surveys, Fourteenth edition, 2014](https://ecat.ga.gov.au/geonetwork/srv/eng/catalog.search?node=srv#/metadata/f3ad4f15-96bd-0cf3-e044-00144fdd4fa6).

The Lineage statement for the Digital files dataset explains:

*This dataset supercedes the previous version of the product released in January 2013 (Geocat #74888) and earlier versions released in October 2011 (Geocat #73075), May 2004 (#61337), May 2003 (#47656), June 2002 (#40757), June 2001 (#36834) & October 2000 (#35181).*

No similar explanation is provided in the Lineage statement for the Index itself.

A DOI has been assigned to the 14th edition, but is not visible in the default metadata record view in the catalogue.

Separate catalogue records exist for the 12th and 13th editions of the Index with information provided in Description and Lineage fields updated accordingly. No link to a later version of the Index is provided in the catalogue records for 12th and 13th editions. DOI have not been retrospectively assigned to 12th and 13th editions.

An eCAT search for “Index of airborne geophysical surveys” returns results for several versions of the Index. There appears to be no weighting to the search results that elevates the most recent version.

Example 3:a single metadata record, and DOI, for a dataset subject to update.

*Electricity Transmission Lines*

<http://pid.geoscience.gov.au/dataset/ga/83105>

The National Electricity Transmission Lines dataset presents the spatial location, in line format, of all known high voltage electricity transmission lines that make up the electricity transmission network within Australia.

A DOI has been assigned to the catalogue metadata record for the dataset, not the dataset being described. No version number appears in the catalogue record.

The Lineage statement in the catalogue record explains:

*The electricity transmission lines were digitized in 2011 from the library of imagery held within Geoscience Australia. Imagery used ranged from 0.15m resolution aerial photos to 2.5m resolution satellite images. The database was revised in January 2014 to reflect the most current version of the Australian Energy Market Operator (AEMO) Transmission Network Diagrams dated 14 February 2013.*

More detail is provided in the Metadata Statement pdf which can be downloaded from the catalogue record.

The Metadata Statement pdf is titled:

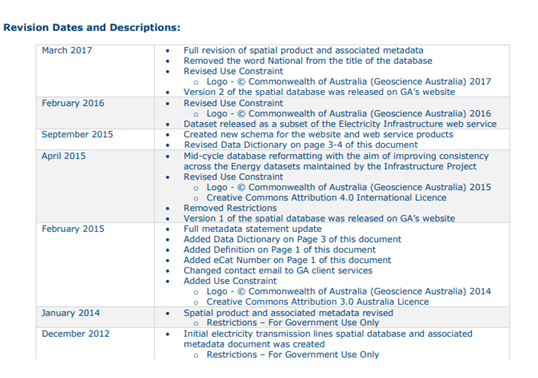
Electricity Transmission Lines Database Metadata Statement

Version 2 Last updated in 2017 eCatID: 83105

The lineage statement in the Metadata Statement pdf states:

*The electricity transmission lines were digitized in 2011 from the library of imagery held within Geoscience Australia. Imagery used ranged from 0.15m to 2.5m resolution. The electricity transmission lines dataset was revised (Version 2) in March 2017 using Esri World Imagery. Version 1 of the database was first released publicly on Geoscience Australia’s website in April 2015 and the updated revision re-released as Version 2 in March 2017 The electricity transmission lines webservice – Version 1 was released as a subset of the Electricity Infrastructure web service in February 2016.*

A full revision history is also provided in the Metadata Statement pdf



The catalogue record links to the current version of the data. A user must read the Metadata Statement pdf to determine the version number of the dataset and when the dataset was last updated.

**11. Geoscience Australia: Earthquake Seismic Data** (Margie Smith)

Geoscience Australia (GA) receives real-time data from over 60 seismic stations in Australia and more than 130 international seismic stations. The seismic information is automatically analysed by Geoscience Australia's seismic monitoring and analysis systems that form part of the 24 hours a day, seven days a week operations centre.

According to the *National Archives of Australia Records Disposal Authority for Geoscience Australia* (2005), GA has a legal requirement to retain:

* Records documenting advice/technical advice provided to government agencies on potentially damaging earthquakes and tsunamigenic events.
* Records documenting advice/technical advice provided by the agency on earthquakes and engineering seismology to standards bodies, insurance industry, public.

In order to satisfy this requirement, at the time such advice is provided, GA must capture a snapshot of the entire seismic dataset, package it with related inputs and store it in the Corporate Data Store for retention in line with NAA requirements. This snapshot is regarded as a version of the database.

The models below shows the process developed by GA to meet NAA requirements. A proof-of-concept implementation has successfully been undertaken to test the process.

Figure 1: The problem was how to capture all the associated information related to a packaged product

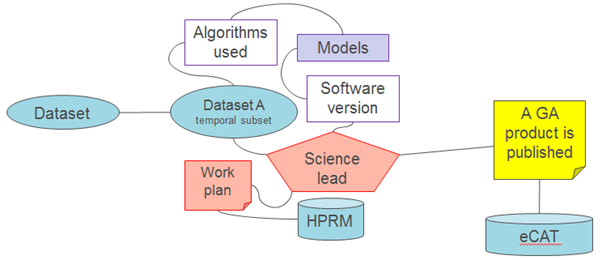
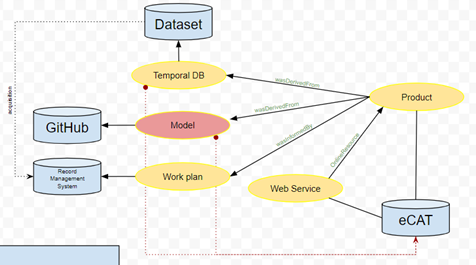


Figure 2: A model was developed and successfully tested with Tropical Cyclone Risk Model Stochastic Event Catalogue



**12. Climate Model Data (CMIP6)** (Kate Snow)The purpose of CMIP (Coupled Model Intercomparison Project) is to provide state-of-the art multi-model advancements coordinated at an international level to improve our understanding of past, present and future climate change. The CMIP data forms an important component of the high end climate research that is assessed as part of the Intergovernmental Panel on Climate Change (IPCC) and to inform policy makers in making evidence-based decisions in relation to current and future climate change.

For CMIP6 data, version labels are standardized to be vYYYYMMDD and the version date is the publication date.

An already published version of a publication unit must not be changed. This means no addition, deletion or replacement of files which are part of a publication unit. Any change must lead to a new version.

A new version can only be created on discovery of an [errata](https://es-doc.github.io/esdoc-errata-client/) and if justification is given for the requirement of a new version. Publication of a newer version of a dataset needs to have a valid motivation, which is referred to as an issue.

The [ESdoc-errata project](https://meetingorganizer.copernicus.org/EGU2017/EGU2017-7647.pdf) created an issue tracker platform for CMIP6 to keep track of the issues affecting specific versions of datasets/files. It enables users to resolve the history tree of each dataset/file.

It is [recommended](https://www.geosci-model-dev-discuss.net/gmd-2018-52/gmd-2018-52.pdf) that the unit of versioning be an atomic dataset: a complete timeseries of one variable from one experiment and one model. The implication is that other variables need not be republished, if the error is found in a single variable. If an entire experiment is retracted and republished, all variables will get a consistent version number.

Old versions do not need be published and PIDs are important in the version management. For example, the data might be no longer available in a certain version as it gets revised and published under a new version, but the information on its previous version remains (a PID on such a file should point to a tombstone page). Ideally, the PID target page for the old and unpublished data version should include errata information and provide a link to the latest (revised) data version.

There a requirement that at least one instance of each submitted dataset be stored at a Tier 1 node (in addition to its primary residence) within a reasonably short time period following submission and that subsequent versions of submitted datasets are also replicated by at least one Tier 1 node.

The description below of the CMIP abstract versioning workflow and version domains is drawn from this discussion [document](https://docs.google.com/document/d/1tOaFQEXFyjqAOOlvcdiaX3XrXuxIv_nlE5_FCme1id4/edit#heading=h.lqx3exufg7s8) that has not yet been officially ratified, but is reproduced here as an example of one approach.

# DRAFT Abstract versioning workflow:

*Initial version publication:*

**pre-condition:** unversioned, complete CMIP6 publication units in agreed CMIP6 directory structure

**pre-condition:** version-string in agreed format

**pre-condition**: map file for publication units (including version information, generated by

esgscan\_directory command or locally developed software - but same

format of map files, format definition at ...site… )

* (major storage sites: add version-info in directory tree of the new CMIP6 publication units)
* merge with existing “ESGF accessible” (thredds accessible) storage pool
* publish in ESGF

**post-condition**: local CMIP6 data pool including versioning information reflected in directory structure

**post-condition:** ESGF published thredds catalogs reflecting version

**post-condition:** ESGF solr index with latest etc. version information

**post-condition:** PID metadata reflecting published units as “latest” version

*New Version:*

**pre-condition:** complete new CMIP6 publication units in agreed CMIP6 directory structure

**pre-condition:** version string in agreed format, indicating “newer” version

**pre-condition:** separate “annotation” describing the background of the version change:

e.g. reason, reference to additional info, changes made etc.

* check version (existence, version format, greater than previous) and check checksums in case of “same version publication actions” (broken publication activities requiring “same version” publications)
* add version-info in directory tree(s) of CMIP6 publication units
* merge with existing “ESGF accessible” storage pool
* publish in ESGF

**post-condition:** local CMIP6 data pool including versioning information reflected in

directory structure

**post-condition:** ESGF published thredds catalogs reflecting version

**post-condition:** ESGF solr index with latest etc. version information

**post-condition:** PID metadata including link to previously published version

**post-condition:** versioning related annotation (optionally “basic stub”) published alongside new version

*Addition of an old version of publication unit:*

**pre-condition:** an old version of a publication unit is to be published. (E.g. re-publishing after a fault or upgrade).

**pre-condition:** a more recent version of the publication is already published to ESGF.

**pre-condition:** complete new CMIP6 publication units in agreed CMIP6 directory structure

**pre-condition:** version string in agreed format, indicating “newer” version

**pre-condition:** separate “annotation” describing the background of the version change:

e.g. this might include information about the newer version.

* check version (existence, version format, greater than previous) and check checksums in case of “same version publication actions” (broken publication activities requiring “same version” publications)
* add version-info in directory tree(s) of CMIP6 publication units
* merge with existing “ESGF accessible” storage pool
* publish in ESGF

**post-condition:** local CMIP6 data pool including versioning information reflected in

directory structure

**post-condition:** ESGF published thredds catalogs reflecting version

**post-condition:** ESGF solr index with latest etc. version information

**post-condition:** PID metadata including link to previously published version AND newer  
 versions

**post-condition:** versioning related annotation (optionally “basic stub”) published alongside new version

*Version retraction:*

**pre-condition:** identification of CMIP6 publication units to be retracted

(including version info)

**pre-condition:** separate “annotation” describing the background of the version retraction:

e.g. reason, reference to additional info, changes made etc.

* un-publish in ESGF, including publication of “annotation”
* adapt the “latest” link in the storage pool (if relevant)

**post-condition:** unchanged local CMIP6 data pool

**post-condition:** updated ESGF published thredds catalogs removing version inf

**post-condition:** updated ESGF solr index removing involved versions

**post-condition:** PID metadata updated reflecting new version chain

(re-publication of retracted data = new version + annotation !)

**post-condition:** versioning related annotation (optionally “basic stub”) published alongside new version

*Version removal:*

**pre-condition:** complete set of CMIP6 publication unites to be unpublished

**pre-condition:** separate “annotation” describing the background of the removal

e.g. reason, reference to additional info, changes made etc.

* remove involved CMIP6 publication units in storage pool (e.g. adapting “latest” link”)
* un-publish in ESGF (including publication of annotation)

**post-condition:** changed publication units from ESGF storage pool

* adapt versioning information CMIP6 data pool with involved publication units removed

**post-condition:** new ESGF published thredds catalogs reflecting removal

**post-condition:** new ESGF solr index reflecting removal

**post-condition:** PID metadata updated with indication that version was removed

(object permanently unavailable)

**post-condition:** versioning related annotation (optionally “basic stub”) published alongside new version

# Versioning domains:

Thus different “domains” or levels of versioning can be separated:

1. Versioning of datasets (and individual files) at the storage level (reflecting versioning info on the file system level e.g. by consistently maintaining soft/hard links)
2. Versioning of ESGF published datasets at the ESGF infrastructure level (ESGF metadata in thredds and solr consistently searchable etc.)
3. Versioning of datasets (and individual files) at the PID infrastructure level: PID metadata associated to PID based tracking ids (and collection ids) contains versioning information (links to predecessor/successor PIDs)

Sources (and with thanks to Kate Snow, NCI)

[CMIP6 versioning requirements collection](https://docs.google.com/document/d/1tOaFQEXFyjqAOOlvcdiaX3XrXuxIv_nlE5_FCme1id4/edit#heading=h.lqx3exufg7s8)

[CMIP6 Data Citation and Long-Term Archival](https://www.earthsystemcog.org/site_media/projects/wip/CMIP6_Data_Citation_LTA.pdf)  
[Requirements for a global data infrastructure in support of CMIP6](https://www.geosci-model-dev-discuss.net/gmd-2018-52/gmd-2018-52.pdf) (Section 7)

**13. CSIRO Data Access Portal** (Dom Hogan)

The Commonwealth Scientific and Industrial Research Organisation (CSIRO) is Australia's national science agency and one of the largest and most diverse research agencies in the world. The [CSIRO Data Access Portal](https://data.csiro.au/dap/home?execution=e1s1) (DAP) provides access to research data, software and other digital assets published by CSIRO across a range of disciplines.

In the DAP, a new version is created whenever a dataset or its metadata record are changed. A new landing page is created, and a new DOI may be assigned. The criteria to mint a new DOI involves a substantial change to t he metadata fields that make up the attribution statement and/or adding or deleting files. Metadata updates that do not substantially affect the attribution statement, and where no change is made to the data, will not receive a new DOI. Instead, the last DOI for that exact version of the data will redirect to the most recent metadata record. In all cases, the version number is automatically included in the citation statement.

Any alteration to a DAP collection is recorded accurately through the use of version control. Changes to metadata and/or files in the DAP create a new version. The previous file(s), Archival Information Packages (AIPs) and Dissemination Information Packages (DIPs) are retained. The current version is returned in query results.

Fig 1: a logged in user can view the full version history of a collection

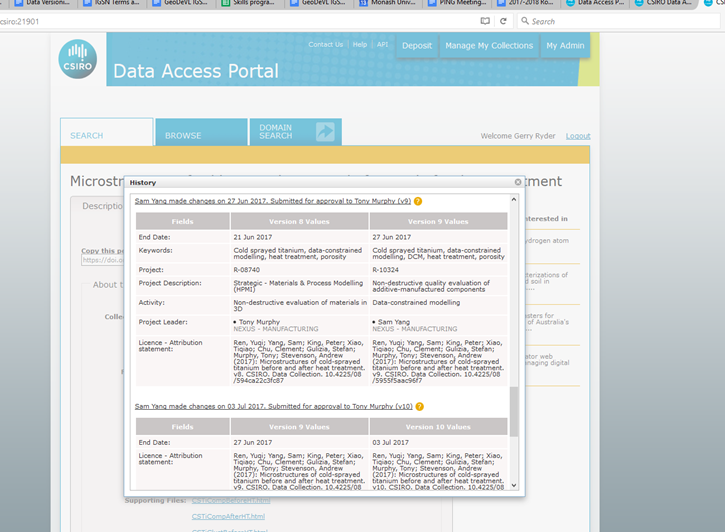
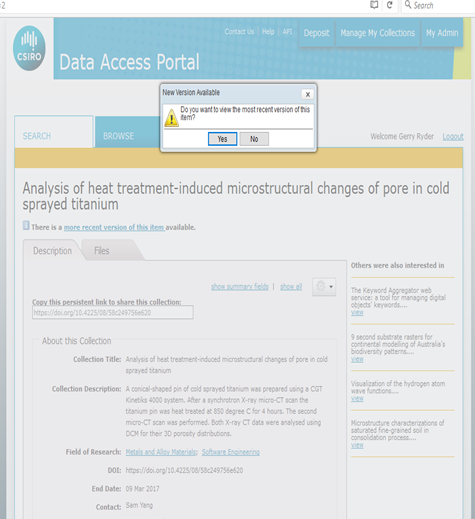
****

Fig 2: a user that resolves the DOI for a ‘previous’ version is alerted to the most current version



**Software** published via the DAP is assigned a DOI. Depositors are advised that best practice is to use the DAP to publish major releases and to make their code repository accessible and linked if they wish to make minor releases available.

To create a new release for software already in the DAP, users can update the record with the new files and the DAP will automatically create a new DAP version and assign a new DOI, with access to the previous version retained.

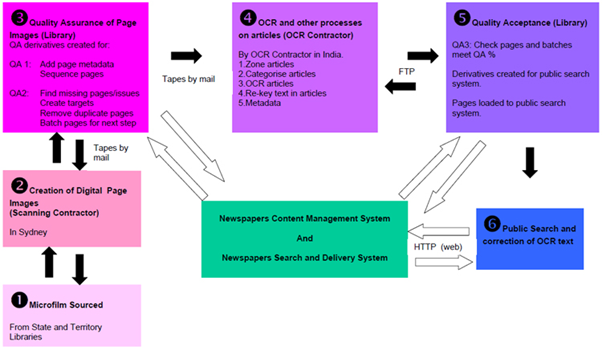
## 14. NLA TROVE Government Gazettes Collection on CloudStor (Catherine Brady & Julia Hickie)

A gazette is an official publication for the purpose of notifying the public of government business. All Australian governments (Commonwealth, State and Territory) publish official gazettes.

Notices published in government gazettes cover all aspects of government concern and regulation, and most are published because of a requirement of law. Acts, regulations and other subordinate legislation are notified in all gazettes, with some states publishing regulations in full as part of the notification.

Background

The National Library of Australia provides access to a digitised collection of gazettes via the [TROVE](https://trove.nla.gov.au/gazette) service. Broadly, the digitisation process involves a 6 step process represented in the diagram below.



Source: http://help.nla.gov.au/trove/for-digitisation-partners/digitisation-workflow-process-overview

The dataset published via TROVE is comprised of scanned page images which are displayed alongside the OCR text for the page. Each article in the dataset has metadata added to indicate Title, Issue number, Pagination and more.

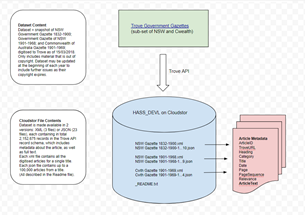
Once published on TROVE, members of the public can correct errors in the OCR text and may add comments and tags such as keywords that can be viewed by opening a dialogue box in the user interface. These changes are saved in the database and available to others. There are already millions of lines of text corrected representing millions of changes to the original published dataset.

Gazettes on Cloudstor

A current project, run under the auspices of the HASS DeVL project, is to upload a snapshot of the OCR text from a subset of the TROVE Government Gazettes database to Cloudstor. The intent is to make the OCR text more accessible and interoperable for research purposes to enable for example, text mining and analysis. As of March 2018, the Cloudstor dataset provides access to:

* NSW Government Gazette 1832-1900;
* Government Gazette of NSW 1901-1968;
* Commonwealth of Australia Gazette 1901-1969

The workflow for achieving this is illustrated below.



Credit: Catherine Brady

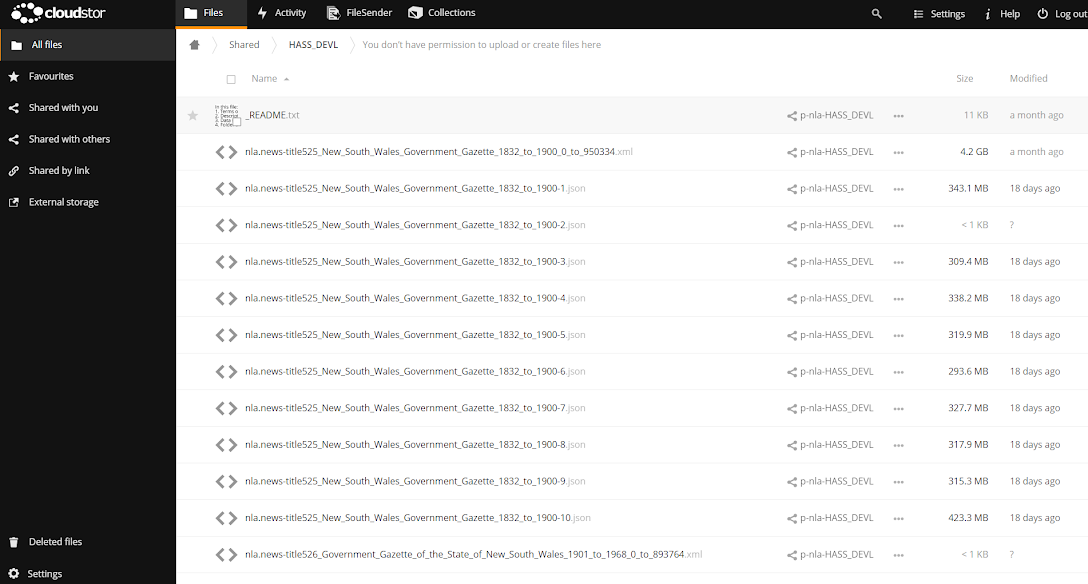
On Cloudstor, the dataset is made available in 2 file formats

1. XML (3 files) containing all the digitised articles for a single (Gazette) title
2. JSON (23 files) containing up to 100,000 articles from a (Gazette) title

All are described in the readme file.

Each format provides access to more than 2 million records in the TROVE API record schema which includes metadata about the article, as well as the full text.

The screenshot below from Cloudstor shows the file structure of the dataset.



Versioning challenges in the Cloudstor collections

The Cloudstor collections are a snapshot of the evolving dataset on TROVE. The dataset evolves in two ways:

1. Frequent updates to the OCR text in TROVE are made as members of the public annotate and correct the OCR text.
2. The size and time coverage of the dataset will increase each year as later issues of the Gazettes are digitised as their copyright expires.

There is likely to be demand from the research community to periodically update the snapshot on Cloudstor so the content available there is aligned to content in the TROVE dataset. This presents a number challenges for managing versions in the Cloudstor collections.

1. Currently, there is no formal version management procedure in the TROVE dataset that could be mirrored in the Cloudstor dataset.
2. The dataset in Cloudstor is structured differently to the dataset in TROVE with Cloudstor providing access via 26 discrete collections as described earlier.
3. User annotations, and information about them (who, when, what etc) are captured in TROVE, but the Cloudstor collections contain the most corrected version of the OCR text as it stood at the time the snapshot was taken ie. OCR text plus user corrections. Users of the Cloudstor dataset wishing to roll-back to an earlier version of OCR text, or view information about user annotations, would currently need to do so via the TROVE interface.
4. A further issue associated with 3. is that in some cases, the user annotations may themselves be the object of study. Hence, the project team is currently exploring how the provenance associated with OCR text could be made available via the Cloudstor dataset.

The TROVE team are currently discussing approaches to versioning the Cloudstor collections. They are also keen to assign DOI to the collections. Current thinking is to:

* Create an annual snapshot or version of the data that will be stored on CloudStor using the existing folder or collections structure
* Assign a DOI to each annual snapshot or version
* Create a landing page on Drupal for each snapshot or version that the DOI will resolve to.

**15. Molecular Bioscience** (Jeff Christiansen)

**Nucelotide sequence data** (i.e. genomic (DNA) or transcriptomic (mRNA/cDNA))

It is a community recognised requirement that prior to peer-reviewed manuscript publication, biological nucleotide sequence data generated to underpin any study findings are deposited into the global nucleotide sequence archives. These archives are managed primarily by the NCBI (US) or EBI (Europe), and sequences are submitted through a number of submission portals, and then exchanged between the two resources. At the point of submission, sequence data are assigned an accession number (i.e. A persistent identifier) and a version number (i.e. #1) (e.g. see example K03160.1)

Once housed within NCBI or EBI, curation actions are undertaken on the sequences submitted (for example, removal of artifacts such as short linking sequences from the ends). After such activities, the curated sequences (which are still identified with the original accession ID) are assigned a new version to identify the change, and appropriate notes are added in the associated metadata to describe the change.

Additionally, much work is undertaken to align multiple observed raw sequences deposited into the archives to generate evidence-based (yet artificial) ideal ‘reference sequences’. Reference sequences are assigned a new and unique accession ID, and will contain metadata indicating the raw sequences that have been used to contribute to the generation of the ideal reference sequence.

All nucleotide sequence data distributed by NCBI-GenBank is in flat file format. As shown in the figure below. The area within the red rectangle is head file, the fourth line contains the current GenBank file format release number. The release number consists of three numbers separated by a decimal point. The number to the left of the decimal is the major release number. The digit to the right of the decimal indicates the version of the major release; it is zero for the first version. Note the format has been stable since 1992. [Copied from [NCBI-GenBank Flat File Distribution Release Notes](ftp://ftp.ncbi.nih.gov/genbank/gbrel.txt)]

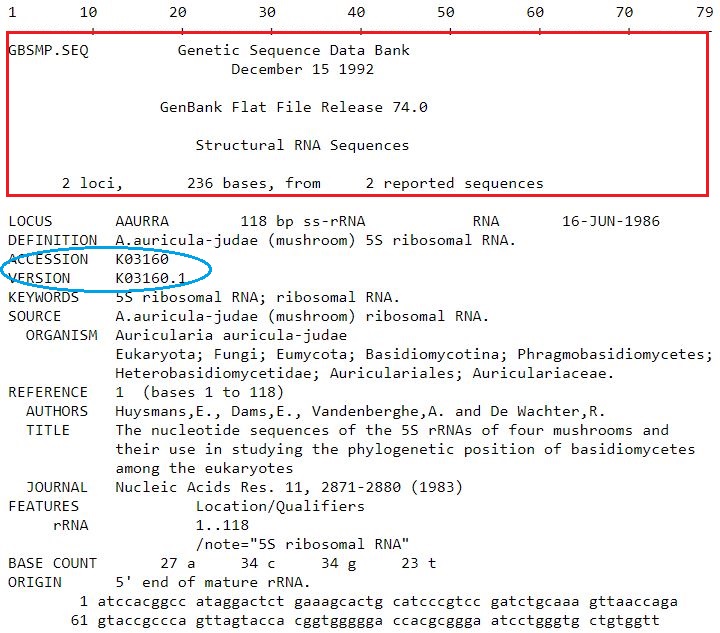
[The release](https://www.ebi.ac.uk/cgi-bin/sva/sva.pl?query=AC005060&search=Go%21&snapshot=&page=-1&session=%2Ftmp%2FSESSION13604-1535503300-1&format=EMBL&help=1) refers to the quarterly EMBL (Europe equivalent of NBCI) release in which a flat file appeared, or was expected to appear.

Below the red rectangle is the data entry, including a global persistent identifier (ACCESSION) and Version.

ACCESSION - The primary accession number is a unique, unchanging  
identifier assigned to each GenBank sequence record. (Please use this  
identifier when citing information from GenBank.)

VERSION - A compound identifier consisting of the primary  
accession number and a numeric version number associated with the  
current version of the sequence data in the record. This is optionally  
followed by an integer identifier (a "GI") assigned to the sequence  
by NCBI.

This example data file is displayed at the NCBI web portal as [this](https://www.ncbi.nlm.nih.gov/nuccore/K03160.1?report=genbank&to=118).

****

(A sample sequence data file from N[CBI-GenBank Distribution Release Notes](ftp://ftp.ncbi.nih.gov/genbank/gbrel.txt))

Like other PIDs, Accession number consists of  [a prefix](https://www.ncbi.nlm.nih.gov/Sequin/acc.html) and numerals. The GenBank [data submission guide states](https://www.ncbi.nlm.nih.gov/genbank/submit/): GenBank will provide accession numbers for submitted sequences, usually within two working days. This accession number serves as an identifier for your submitted your data, and allows the community to retrieve the sequence upon reading the journal article. The accession number should be included in your manuscript, preferably in a footnote on the first page of the article, or as required by individual journal procedures.

Versions of synthetic data objects (i.e. synthetic sequences) are created through computational methods (eg assembling short sequences into longer sequences), or adding annotations (e.g. where does a gene start or end) as well as curation activities done by the database (e.g. Genbank at NCBI) either via computational pipelines or manual / semi-manual processes. Anyone who made a change to a previous version can submit a new version to NCBI DB. [This guideline](https://www.ncbi.nlm.nih.gov/genbank/update/) explains how to submit a revised or updated a sequence, revision includes editing source information, updating publication information, updating nucleotide sequence, adding features, and updating features. There is a format and encoding for each type of updates.

VERSION is made of the accession number followed by a dot and a version number (and is therefore sometimes referred to as the “accession.versionNo”). Example of a sequence in its 5th version <https://www.ncbi.nlm.nih.gov/nuccore/NM_182700.5>. NM\_182700.5 is the pid for the specific 5th version. An accession number without a version suffix always refers to the latest version of the sequence data.

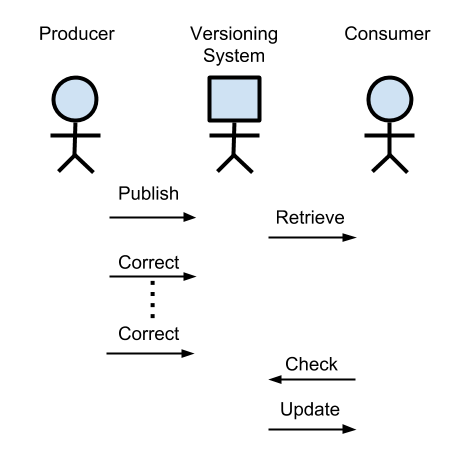
To see the revision history of a sequence, append *report=girevhist* to the record's URL. For example, [accession U46667](https://www.ncbi.nlm.nih.gov/nuccore/U46667)'s revision history's URL is [http://www.ncbi.nlm.nih.gov/nuccore/U46667 **?report=girevhist**](https://www.ncbi.nlm.nih.gov/nuccore/U46667?report=girevhist), or Query an Accession Number will enable access of all version of the ID.

EMBL-EBI offers a Version Checker, which highlights what changes have been made between two selected versions. Here is [an example.](https://www.ebi.ac.uk/cgi-bin/sva/sva.pl?query=AC005060&search=Go%21&snapshot=&page=-1&session=%2Ftmp%2FSESSION13604-1535503300-1&format=EMBL) This Version Checker compares two flat files and uses different coloured lines to represent whether a line has remained unchanged (white), whether it has been deleted (orange), or whether it has been inserted (green).

<https://www.ebi.ac.uk/cgi-bin/sva/sva.pl>

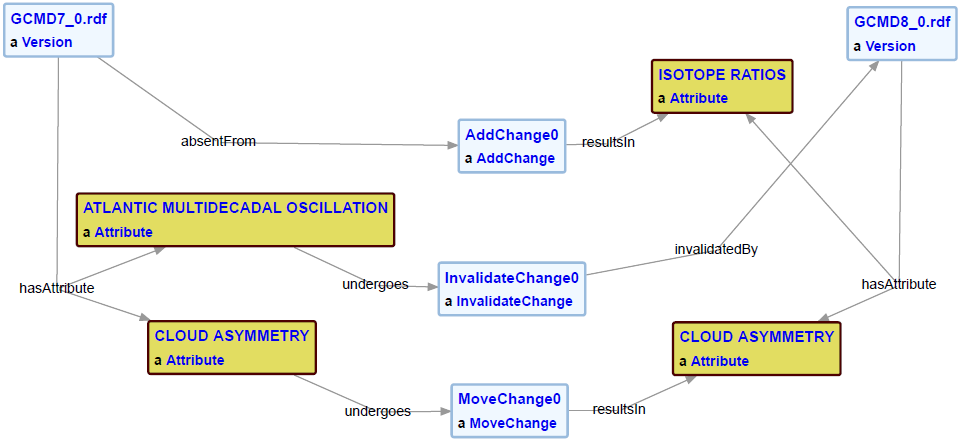
## 16. Versioning Ontology (VersOn) (Benno Lee)

A major tradition in versioning practice is the use of dot-decimal identifiers. The identifiers categorize data objects as a major, minor, or smaller difference from the previous iteration. One major issue in using data identifiers to indicate or measure the amount of change between versions is illustrated in the following diagram.

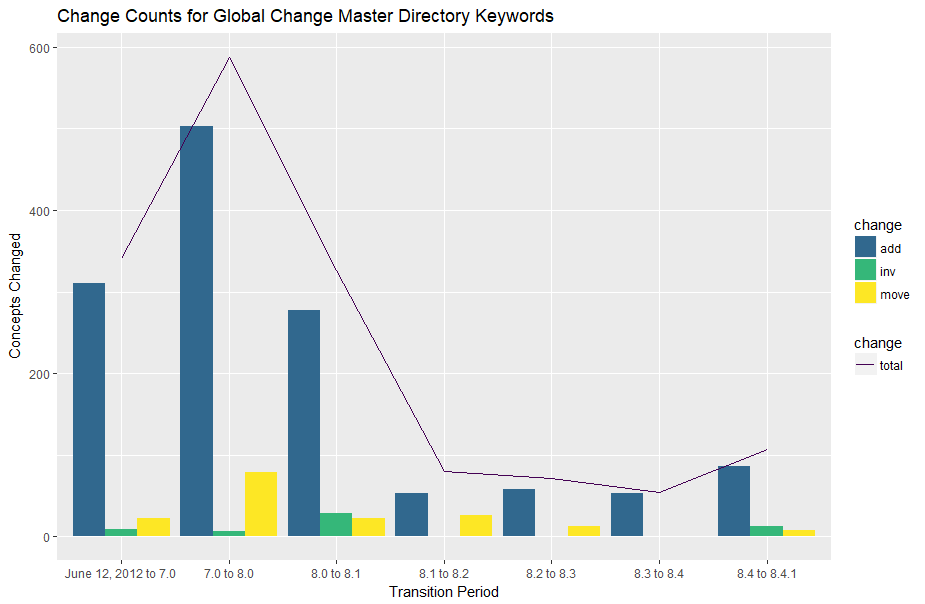


Between the Data Producer and Data Consumer, only the Producer supplies information into the versioning system. Leaving authority solely with the Producer means that the impact introduced by changes is not assessed using the Consumer’s context.

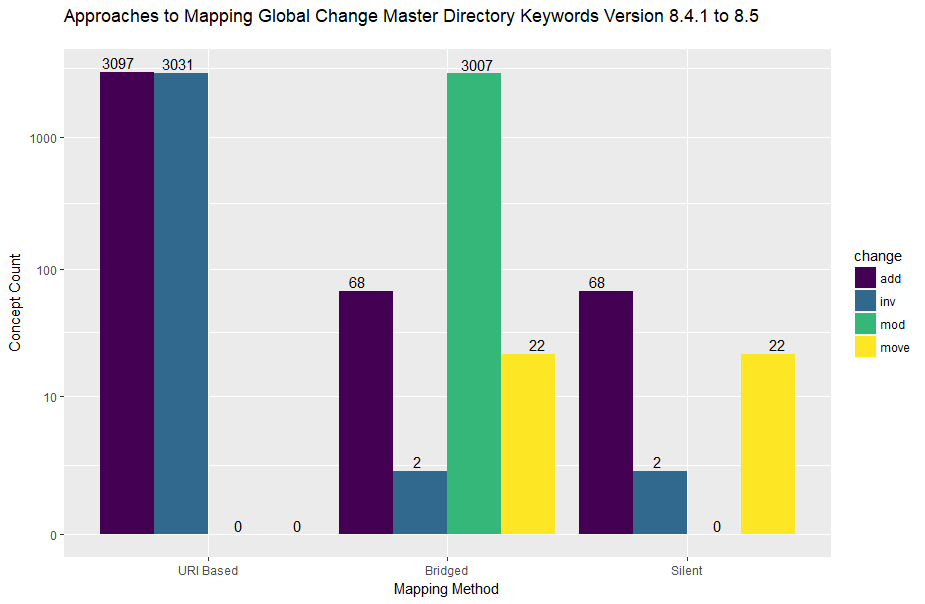
The Versioning Ontology (VersOn) is a linked-data ontology (currently located at <http://orion.tw.rpi.edu/~blee/VersionOntology.owl>) which captures individual changes between data objects as linked data. It organizes changes into three classes: Additions, Invalidation, and Modifications. The resulting versioning graph forms a ladder-like structure where the rungs can be counted as a method to assess change to a greater precision than the broad categories of dot-decimal identifiers.



The ontology was used to capture the changes within NASA’s Global Change Master Directory (GCMD) Keyword taxonomy.



VersOn breaks the total change into component parts so that trends based on operations can be seen. Here, GCMD Keywords experience steady growth with many additions in blue as compared to more stable data sets which may primarily feature modifications. The VersOn method also illustrates an interesting dynamic between Producer and Consumers with the publication of GCMD Keywords Version 8.5.



In Version 8.5, the protocol of the keyword namespace was changed from http to https, but the change is non-trivial to web URIs. As a result, using URI best practices results in the assessment that the entire taxonomy has been invalidated and a new set of keywords are added. The GCMD Keywords group directed some consumers to adjust their software to manually map between the new keywords. From the informed consumer’s perspective, a Bridged assessment can be made where around 3000 keywords have been modified in green. Finally, the Silent method ignores the changed namespace. We know the GCMD Keywords group used the Silent method because it assigned an minor version dot-decimal identifier to the release, indicating an incremental release. The URI-Based and Bridged methods both have changes on the order of the entire data set, constituting a full release and necessitating a major version identifiers.

VersOn enables detailed change assessment to be conducted regardless of the major or minor identifier assigned to the version. The assessment can be performed after the data set’s publication, that is, after the version identifier has been assigned. Additionally, VersOn enables an assessment to be conducted by the consumer in the context of the consumer’s application.

## Adopters of the RDA Recommendations on Dynamic Data Citation:

The following data centers have abandoned semantic versioning as a versioning strategy and are adopting a time-stamping based approach to document changes to a data with historization allowing to go back to any early state of the data at any arbitrary point in time. Detailed documentation of these adoptions including webinar recordings, slide sets as well as additional material are available at the Webinar Series page of the RDA Working Group at:

* **Implementing of the RDA Data Citation Recommendations by the Climate Change Centre Austria (CCCA) for a repository of NetCDF files**
  + Presenter: **Chris Schubert**, Head of the CCCA Data Center, Vienna, Austria
  + **Recording is available at**:<https://www.rd-alliance.org/implementing%C2%A0-rda-data-citation-recommendations-climate-change-centre-austria-ccca-repository-netcdf>
  + **Slides are available at**:<https://www.rd-alliance.org/rda-wgdc-webinar-slides-chris-schubert-climate-change-centre-austria-ccca>

* **Implementing the RDA Data Citation Recommendations for Long-Tail Research Data / CSV files**
  + Presenter: **Stefan Pröll**
  + **Recording is available at:**<https://www.rd-alliance.org/implementing-rda-data-citation-recommendations-long-tail-research-data-csv-files>
  + **Slides are available at**:<https://www.rd-alliance.org/system/files/documents/20170518-RDA-StefanProell.pdf>
  + **Supporting papers are available at**:
    - Stefan Pröll, Kristof Meixner and Andreas Rauber. Precise Data Identification Services for Long Tail Research Data. 13th International Conference on Digital Preservation (iPRES). 2016.<https://www.rd-alliance.org/ipres2016-paper-implementing-wgdc-recommendations-long-tail-research-data-csv-files>
    - *Stefan Pröll. Enabling Reproducibility for Small and Large Scale Research Data Sets. D-Lib Magazine, January/February 2017, Volume 23, Number 1/2*
    - *<http://www.dlib.org/dlib/january17/proell/01proell.html>*
    - *Stefan Proell and Andreas Rauber. A Scalable Framework for Dynamic Data Citation of Arbitrary Structured Data," in 3rd International Conference on Data Management Technologies and Applications (DATA2014), 2014*
* **Implementing the RDA Data Citation Recommendations in the Distributed Infrastructure of the Virtual and Atomic Molecular Data Center (VAMDC)**
  + Presenter: **Carlo Maria Zwölf**, VAMDC, Observatoire de Paris, France
  + **Recording is available at:**<https://www.rd-alliance.org/dynamic-data-citation-within-distributed-infrastructure-virtual-and-atomic-molecular-data-center>
  + **Slides are available at**:<https://www.rd-alliance.org/webinar-slides-carlo-maria-zw%C3%B6lf-implementing-rda-data-citation-recommendations-distributed>
  + Supporting paper is available at: *C.M. Zwölf, N.Moreau, M-.L. Dubernet, New Model for dataset citation and extraction reproducibility in VAMDC, Journal of Molecular Spectroscopy, doi:10.1016/j.jms.2016.04.009, (arXiv version at* [*http://arxiv.org/abs/1606.00405*](http://arxiv.org/abs/1606.00405)*)*
  + <https://www.rd-alliance.org/journal-molecular-spectroscopy-paper-implementing-wgdc-recommendations-vamdc-infrastructure>
* **Implementation of Dynamic Data Citation at the Vermont Monitoring Cooperative**
  + Presenter: **James Duncan**, VMC, University of Vermont, Burlington, VT
  + **Recording is available at**:<https://www.rd-alliance.org/implementation-dynamic-data-citation-vermont-monitoring-cooperative>
  + **Slides are available at**:<https://www.rd-alliance.org/webinar-slides-james-duncan-implementation-dynamic-data-citation-vermont-monitoring-cooperative>

* **Adoption of the RDA Data Citation of Evolving Data Recommendation to Electronic Health Records**
  + Presenter: **Leslie McIntosh**, PHD, MPH, Director Center for Biomedical Informatics, Washington University in St.Louis
  + **Recording is available at**:<https://www.rd-alliance.org/adoption-rda-data-citation-evolving-data-recommendation-electronic-health-records>
  + **Slides are available at**:<https://www.rd-alliance.org/webinar-slides-leslie-mcintosh-adoption-rda-data-citation-evolving-data-recommendation-electronic>
  + **Supporting paper is available at**:<https://www.rd-alliance.org/amia-joint-summits-2017-paper-implementation-wgdc-recommendation-biomedical-data-wustl> (AMIA Joint Summits 2017)

# 

# Appendix 1: Incomplete use cases

## (A1). TERN (Terrestrial Ecosystems Research Network) Eco-informatics Facility

The TERN Eco-informatics Facility works with governments, researchers, educators and students to make ecological “plot” data (including quadrats, transects, pitfall traplines, cage trap arrays, and other systematic collection methods) discoverable and freely accessible.

The Facility is underpinned by a data submission service known as [SHaRED](https://shared.ecoinformatics.org.au/login) and the [AEKOS](http://www.portal.aekos.org.au/) portal that enables discovery of, and access to, ecological datasets. AEKOS provides access to primary data, provided to the Facility by government agencies and research organisations, as well as derived data, submitted by researchers via the SHaRED service.

The Facility offers a DOI service for data published by researchers via SHaRED.  
In cases where data and corresponding metadata is updated, the repository supports version control via the DOI. New versions are linked to older versions via metadata using the old DOI. In the case of annual and periodic data, new data is appended to previous published version (e.g., 2000‐2015, add 2016 to give 2000‐2016) and published as a full dataset with a new DOI to maintain the integrity of the data collected. This means all versions of a particular dataset are accessible.

A search in AEKOS on “Ausplots Forest Monitoring Network - Forest Fuel Survey” returns records for both v1 and v2 of the dataset.

Fig 1: The landing page for version 1 of the dataset does not reference v.2

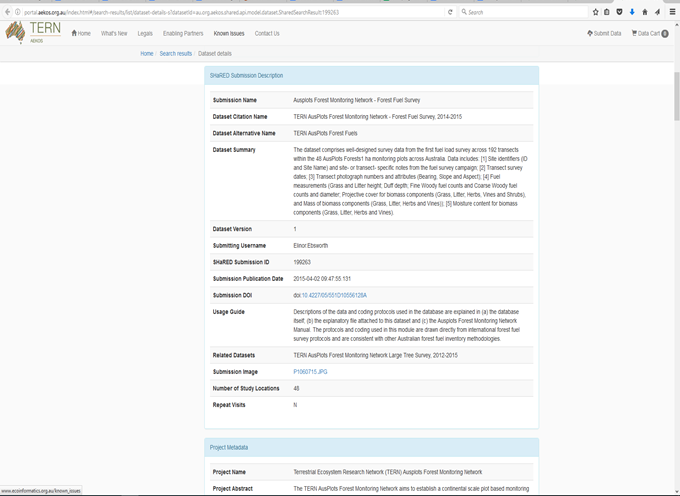
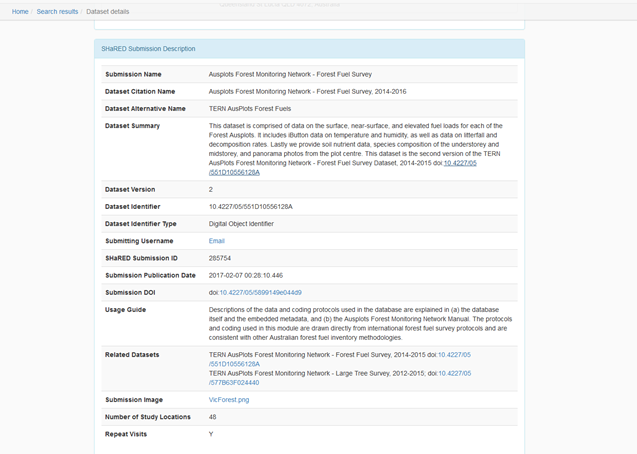


Fig 2. The landing page for version 2 references version 1 and and provides a link to it via the DOI



## 

**(A2). Australian Data Archive (ADA)**

The [Australian Data Archive](https://ada.edu.au/) provides a national service for the collection and preservation of digital research data. ADA holds over 6000 datasets from more than 1500 projects and studies from 1838 through until the present day. ADA data cover longitudinal studies, social attitudes surveys, health data, elections & political studies and public opinion polls. ADA adopts Dataverse platform for managing data.

ADA current versioning approach has been in place since around 2009, versioning numbering system at decimal level was implemented in 2010. Versioning is done at the collection/project level, about 10% of ADA data have been versioned.

* Full new wave of collection = major version change
* Minor changes between full waves = minor version change

Data usually changed annually. Longitudinal data have a new version every year. Actual versioning is not done at ADA - ADA receives updated version from agencies, e.g., Australia Bureau Statistics. Older versions are stored but made available only by request only (i.e. not be available and searchable online).

**(A3) Research School of Earth Sciences (RSES) ANU**

The Passive Seismic group of ANU RSES raised the issue of versioning of passive seismic data at an NCI/RSES meeting. However, we have not yet had any response back from them based on follow up emails Lesley has done.

**NCI Climate Data**