SoleCRIS Datasets

– Navigating multiple import sources and ways to identify datasets

Niilo Lahti
Product owner
Solenovo

Introduction

In Finland, higher education institutions are not required to report research datasets, unlike publications, annually to the Ministry of Education to receive national funding. However, the awareness regarding the importance of datasets for higher education institutions and their researchers has increased over the past years. One of Solenovo's customers, University of Oulu, is in the forefront of Finnish higher education institutions in their want to report datasets to the Finnish national authorities and more specifically to CSC's Metax service.[1] To this end, University of Oulu has requested us, Solenovo,[2] to develop Research datasets feature to our current research information system, SoleCRIS.[3]

In this paper, I wish to share a couple of essential challenges in developing the Datasets feature and our planned solutions for them. More to the point, since datasets affiliated with University of Oulu will be imported from more than one source, there must be well-defined import scenarios so that correct data is prioritized in the right circumstance. Multiple import sources and various processes for creating new metadata entities for actual datasets could also cause situations where the same dataset metadata entity ends up in SoleCRIS. In most cases, doubles of the same dataset are unwanted. Thus, we need several ways to identify the correct metadata entity that points to a given actual dataset. Currently in late March, the final steps before the start of production use are remaining. During euroCRIS Leuven, the production use of the feature may have just begun.

Datasets feature

Before discussing the issues regarding multiple import sources and the identification of (unique) dataset metadata entities, I shall briefly introduce the Datasets feature to clarify the context around these issues. In SoleCRIS, Datasets feature consists of everything except the user adding new dataset metadata entities to the system (for now).[4] "Everything" means importing metadata entities from two sources, Metax API

[1] To learn more, https://wiki.eduuni.fi/spaces/Fairdataverkosto/pages/181341930/Organisaation+aineistojen+kuvailutiedot+Metaxiin

[2] Solenovo is a Finnish IT company offering various solutions to research and education sectors. Its main office resides in Joensuu. To learn more, https://www.solenovo.fi/#1.

[3] SoleCRIS has logged and public environment. For public environment, see e.g., https://oulu.cris.fi/fi.

[4] Researchers describe and publish datasets with Qvain service https://qvain.fairdata.fi/. These dataset metadata entities, that in this case belong to the University of Oulu, are imported to SoleCRIS from Metax API.

and DataCite API.[5] It means also that in SoleCRIS, the metadata can be enriched in edit view by, e.g., distinguishing local actors from other actors, refining keywords from imported subject headings and imported keywords and linking other material related to the dataset. The data enrichment and its quality are aided by process states (workflow)[6] and validations[7]. Furthermore, the metadata is displayed to different user roles in different ways.[8] The metadata entities can of course be searched and reported. Logged user can write comments in the metadata entities. Lastly, the datasets that have been imported from or are updated by a non-Metax import source are exported to Metax.[9]

Identifying unique dataset metadata entities

After the brief feature introduction, we can proceed to the challenge of multiple import sources and the identification of (unique) dataset metadata entities. The goal with the importing, in general, is to provide the metadata manager the latest version of the metadata entity containing best data quality. When importing, unique datasets can be identified by comparing the imported dataset's persistent identifier, other identifier(s), or the persistent identifier of previous versions with the same identifiers of datasets that already exist in SoleCRIS. Between existing datasets, title can be used to indicate potential doubles. However, title should not be used to discard any dataset outright.

Dataset metadata entities include a single "Persistent identifier" (persistent_identifier). It represents the primary unique identifier of a dataset. Every dataset must have one, but they can vary depending, e.g., on the source of the dataset. In SoleCRIS, "Persistent identifier" is displayed in a read-only field so that the metadata manager cannot change it. In the first feature development phase that precedes the beginning of production use, imported and existing datasets are compared only based on "Persistent identifier". Since we have two import sources, this comparison causes four import scenarios. The imported entity can be either from Metax or DataCite and existing entities can likewise be either from Metax or DataCite. I will go through these scenarios in detail below.

"Other identifiers" are needed because the same dataset version can have metadata entities with different "Persistent identifiers". Thus, double metadata entities can be

---

[5] Metax API V3 documentation https://wiki.eduuni.fi/spaces/Fairdataverkosto/pages/469015907/In+English+Data+Model+and+metadata+in+Metax+V3. CSC launched V3 of the API in February 2025. Introduction to DataCite REST API https://support.datacite.org/docs/api. Instead of DataCite, the initial plan was to import dataset metadata using Elsevier API / Mendeley Data Monitor. However, during fall 2024 the University of Oulu discovered that Elsevier was planning to discontinue the public API and incorporate it to Pure. See https://data.mendeley.com/api/docs/#section/Introduction.

[6] Datasets feature process states are "Imported", "Incomplete", "Approved" and "Doesn't belong to university".

[7] Validations consist of format and process state validations.

[8] In logged view, our customers utilize two user roles: Researcher and Librarian. Public view / research portal user can be regarded as a third user role.

[9] Since datasets are imported from Metax and exported to Metax, it needed some consideration during the feature specification process how to avoid endless import-export loop. This problem was readily solved by the fact that Metax has different catalogs which are used for importing and a separate University of Oulu specific catalog which is used for exporting. However, Metax would in this case still end up containing doubles of the same dataset entity both of which originate from Metax. This contemplation resulted in that only non-Metax entities are exported from SoleCRIS to Metax.

caught by comparing the “Persistent identifier” or any of the “Other identifiers” of an imported entity to the “Persistent identifier” or any of the “Other identifiers” of existing entities. These comparisons will cause several more import scenarios which are accounted for in a later feature development phase. In SoleCRIS, fields for “Other identifiers” are automatically generated based on the imported data. In other words, the Datasets form can contain any number of “Other identifier” data in separate fields depending on how many “Other identifiers” are included in the imported entity. In addition, user should be able to manage these fields and add new “Other identifiers”. In the long run, they will help to prevent doubles.

“Previous version identifiers” are needed when two or more metadata entities are pointing to different versions of the same actual dataset, and they thus have different “Persistent identifiers” and “Other identifiers”. When a new version of a dataset is established, a new corresponding metadata entity is also created which in turn has a different “Persistent identifier” compared to the previous version. University of Oulu is interested in managing, displaying and exporting only the metadata that refers to the latest version and so a comparison at least between imported entity’s “Previous version identifier” and “Persistent identifier” of existing entities is eventually needed. Similar to “Other identifier”, in SoleCRIS, fields for “Previous version identifiers” are automatically generated based on the imported data. Also similarly, user should be able to manage these fields and add new “Previous identifiers”. Again, these comparisons would cause additional import scenarios which are accounted for in a later phase.

Import scenarios
As mentioned above, matching imported and existing entity with “Persistent identifier” results in four import scenarios:

1. Metax + DataCite (other than Metax)
2. Metax + Metax
3. DataCite + Metax
4. DataCite + DataCite

In these scenarios, entities from Metax are prioritized over entities from other sources (e.g., DataCite) since the data is (in some cases) exported to Metax and SoleCRIS data model for datasets aligns with the Metax data model.[10]

1. Metax + DataCite
In this scenario, a dataset metadata entity has already been imported to SoleCRIS from Metax. This successful import of a new entity is logged to an integration report in SoleCRIS. Afterwards, the same dataset metadata entity (i.e., has the same “Persistent identifier”) is offered to be imported from DataCite (other source than Metax). Since Metax entity is prioritized, the DataCite entity is discarded. The intentionally failed import is logged to the integration report. User refines the data in Metax entity which will not be exported to Metax.[11]

[10] This enables e.g., codeset values from Metax to be mapped as codeset values in SoleCRIS. With certain data such mapping cannot be reliably done with other import sources. Especially if there are many such sources.
[11] See previous note above for considerations regarding Metax exporting.

2. Metax + Metax
Again, a metadata entity has already been imported from Metax, and the successful import has been logged to the integration report. Afterwards, the same dataset metadata entity (same PID) is imported again from Metax. This causes specific fields to be overwritten in the metadata form.[12] Thus, specific refined data may be lost, but it is assumed that the newly imported data is better. The update is logged to the integration report.

3. DataCite + Metax
In this scenario, a dataset metadata entity has already been imported from DataCite (other source than Metax). Successful new import has been logged. If this entity is not updated by Metax, it will be exported to Metax via the integration platform of the University of Oulu. However, if the same dataset metadata entity is imported from Metax, specific fields in the form are overwritten, the update is logged and Metax is marked as the source that updated the form. Thus, the entity will not be exported anymore, and it will not be updated by sources other than Metax (see scenario 1).

4. DataCite + DataCite
In the fourth and final scenario, a dataset metadata entity has already been imported from DataCite and this has been logged. The metadata entity will be exported. Afterwards, the same dataset metadata entity is imported again from DataCite. Specific fields are overwritten,[13] and the update is logged. The entity will be exported again. It is assumed in this scenario that the newly imported entity contains better read-only data than the existing entity.

Summary and next steps
In conclusion, the four basic import scenarios that are implemented before the beginning of the production use, in which imported and existing entities are compared with “Persistent identifier”, successfully prioritize Metax entities over entities from other sources (e.g., DataCite). While double datasets are prevented to a degree, the lack of utilizing “Other identifier” and “Previous version identifier” may cause doubles in some cases. The metadata manager can monitor these four scenarios from the integration report. In some of these scenarios, the data is updated without explicit consent from the metadata manager. While there are good reasons to trust that the updated data is mostly better than the previous data, the manager should in principle (in some cases) make the final call.

The next development phase is informed by the initial production use by the University of Oulu. However, we likely want to develop Datasets feature in which

- Double datasets are better identified with “Other identifiers”, “Previous version identifiers” and perhaps also by “Title”
- The integration report is expanded to account for the more nuanced double dataset identification

[12] These fields likely include ”MinEdu [=Ministry of Eductation] field of science classification”, “Dataset language”, “License”, “Access type”, “Restriction grounds” and “End date of the embargo”.
[13] From editable fields, “Other identifier” and “Previous version identifier” fields that are automatically generated by imported data will be overwritten.

- Imported data and data managed by the manager are more clearly distinguished
- The manager can better compare data from different sources and draw the final verdict what data is updated and by what source