# DSpace-CRIS 7

## version: 2021.03.00

*30th December 2021*

# Technical Documentation

# Technical documentation



## DSpace-CRIS Version 2021.03.00 December, 30th

**Versioning & support model**

Starting from DSpace-CRIS 7 the project has adopted its own versioning model to assure accurate tagging and tracking of the changes across the different releases. Version numbers will use the following schema year.major.minor. The major number will reset to 01 each year, minor number starts from 00 so for example 2021.01.00, 2021.01.01, 2021.02.00, 2022.01.00, etc.

- *Minor versions* are expected to be easier to upgrade to, REST API is guarantee to be backward compatible;
- *Major* versions are used to highlight important functional or breaking changes. The changelog will highlight the new features, the configuration and REST API changes and the version of the plain DSpace that is used as basis;
- *Year versions* are assumed to be major. Minor version related to the previous year can still occur for security fix.

The latest version is actively maintained by 4Science as volunteer work, support for past versions will be based on availability of resources and funding from the community. Security fixes will be bring to the current version and the previous major if the current version is a new one (i.e. minor = 00). For example if version 2021.02.00 is the current one, security fixes will be released for both 2021 major 01 and 02, i.e. version 2021.01.01 and 2021.02.01 will be released. Once version 2021.02.01 will be out, security fixes will comes as 2021.02.02.

DSpace-CRIS and DSpace releases are independent, the used plain DSpace version as basis for a DSpace-CRIS version will be noted in the changelog but, when appropriate it will be possible to get an official DSpace-CRIS release based on a DSpace unreleased version as it was the case for the first DSpace-CRIS 7 release. In such situation the DSpace-CRIS development team have reviewed the know issues in DSpace and provided custom remediation, workaround or alternative functionalities to support the relevant use cases. Some limitation can also be acceptable and just noted in the release notes as they afflict edge scenario or a limited user base.

# Version 2021.03.00 - Release notes

## DSpace-CRIS 7 2021.03.00 December, 30th (REST | Angular)

This version, released on the 30th Dec 2021, provides a new way of rendering an Item detail page, possibility of adding custom html fragments in home page, share functionality. Search and browsing performances have been improved, and further checks have been added to improve security and prevent the saving of Items in an invalid state when edited. Details of this release are reported below.

### Key Enhancements

- New Item Detail page based on the concept of matrix layout
- Enforced metadata profile in no-administrative edit screens (prevent to save change to item with invalid or missing mandatory metadata)
- Edit of html fragments in the home page
- Share on social networks via addthis
- Several minor bugs fixed

### Architecture

- Aligned with DSpace 7 community last improvement, with a preview of several features of 7.2 tag
- performance improvements in browsing and search listing

### Data presentation

- new Item Detail page, with matrix layout
- Hide the bitstream list in the item page when no bitstreams are available
- Provided a standard and semi automatic way to define i18n keys for item layout pages
- Filters and facets can be built using display value for metadata set using lists or controlled vocabularies
- Extended valuepair rendering to metadata defined via a controlled vocabulary
- Layout configuration performed via DSpace-CRIS 7 process or CLI command

### Data security

- Items cannot be saved in an invalid state
- Item security evaluation when exporting in csv or xls.

### Other

- Invitation to join groups

*Based on DSpace commits dbede2b (backend) and 7abdceb (front-end)*

## Version 2021.02.02 - Release notes

The key changes in this version, released on the 14th Dec 2021, relate to alignment with DSpace 7.1.1

**This is an updated version of the dspace-cris-2021.02.01, aligned with DSpace 7.1.1 release, which includes a security update for CVE-2021-44228 (log4j v2 critical vulnerability).** It is fully compatible with the DSpace-CRIS 7 Frontend `dspace-cris-2021.02.01` release.

**We highly recommend ALL users of DSpace-CRIS 2021.01.x or 2021.02.x upgrade to cris-2021.02.02 to resolve CVE-2021-44228.**

To fully protect your DSpace-CRIS 2021.x site from CVE-2021-44228, three steps are required:

1. Upgrade your DSpace-CRIS backend to 2021.02.02 OR manually install #8065, rebuild and redeploy your DSpace-CRIS backend. Make sure to restart your Tomcat after the update.
2. Upgrade to Apache Solr v8.11.1 (or above), **OR** ensure that `-Dlog4j2.formatMsgNoLookups=true` is specified in your `SOLR_OPTS` environment variable. For more information, see https://solr.apache.org/security.html#apache-solr-affected-by-apache-log4j-cve-2021-44228
3. If you use the Handle.Net Registry Support in DSpace-CRIS 2021.x, make sure to restart your Handle Server (after performing step 1), so that it uses the new (secure) version of log4j as well.

For the technical documentation please refers to the dspace-cris 2021.02.01 release notes

## Version 2021.02.01 - Release notes

The key changes in this version, released on the 11th Nov 2021, relate to alignment with DSpace 7.1 tag. Some additional nice features have been also introduced, see below for details

### Key Enhancements

- Alignement with DSpace 7.1 tag
- Performance improvements

### Architecture

- Aligned with DSpace 7.1 tag, several bugs have been resolved

### Data migration

- Added redirection from the legacy CRIS 5 details page to the new CRIS 7 details page

### Data presentation

- MyDSpace is now updated when a task is claimed
- Allowed to use of graphical facets in home page and explore sections
- Added PlumX widget for publication and person metrics
- Hide relations box if the content is empty
- Provided a valuepair rendering type for the item detail

### Data quality & accuracy

- Added the possibility to replace metadata value when item reference is resolved

### Data security

- Metrics boxes does not appear if the current user has no rights to see them
- Allowed to turnoff completely the end user agreement
- Prevented patch and put of item metadata if user is not allowed

### Other

- Fixed the partially breaking of the reset password feature when usage terms are not yet accepted
- Admin is now able to reset other user's password

## Version 2021.02.00 - Release notes

This version, released on the 27th Oct 2021, is aligned with **DSpace 7.0 tag**. It improves the stability of the platform and provide a better support for users migrating from DSpace-CRIS 5. Some additional nice features have been also introduced, see below for details

### Key Enhancements

- Migration from DSpace-CRIS 5 improved (migrate ORCID tokens, metadata values language and boolean values)
- Subscription feature: functionality that allows you to receive updates on specific dspace objects
- Introduced lucky search
- Allow anonymous user to export items and search results

### Architecture

- Aligned with DSpace 7.0 tag, several bugs have been resolved

### Data presentation

- Added tag rendering
- Entity name translation in MyDSpace dropdown
- Sorted user's processes by id
- Improved statistics map rendering with country name localized
- Fixed manage of newline flag for box conftool configuration

### Interoperability

- Introduced an OpenID authentication provider

### Data security

- Defined item_admin, submitter, submitter_group and group security levels for edit/correction modes
- Extended item export to evaluate security context while extracting metadata

## Version 2021.01.01 - Release notes

This is a minor release, no breaking changes have been introduced. This version is aligned with DSpace 7.0 tag, which carries several fixes of bugs present in previous SNAPSHOT version.

#### Key Enhancements

- Metadata security: it is possible to define at different levels (general, entity, metadata) if a metadata value, configured to be displayed in layout configuration, is accessible by all users, including anonymous, or should be visible to only users of given groups or only to administrators and DSpace-CRIS 7 Item's owner.
- Scripts to migrate DSpace-CRIS items from version 5 and 6 installations to new DSpace-CRIS 7 installations are provided. This feature is still in preview.
- Added support for Orcid Funded-By during orcid synchronization (https://info.orcid.org/funded-by-relationship-type/ , https://twitter.com/HelenGNewnham/status/1402183540637376512?s=19)

#### Data collection

- Added possibility of setting submission form's panels opened / closed when submission form is rendered
- Inline metadata groups are sortable

#### Architecture

- Aligned with DSpace 7.0 tag, several bugs have been resolved

#### Data presentation

- Added possibility of having icons in box headers
- Orcid identifier links to profile on ORCID registry
- Updated profile picture export
- OAI: Fixed list records in cerif format
- Statistic accessible via contextual menu
- Improved navigation out of statistics page

#### Profile management

- When a researcher profile is created, data are enriched with target collection's template item (if present)

#### Data quality & accuracy

- Updated deduplication checks: item type and target community and collection are evaluated while detecting duplicates.

#### Interoperability

- Added support for Orcid Funded-By during synchronization
- Improved handling of special characters during ORCID synchronization
- More crosswalk export formats available for Patent entity
- Virtual field to print a date in different formats during crosswalk dissemination
- Virtual field to explore and print part of the controlled vocabulary hierarchy during crosswalk dissemination

#### Data security

- If user belongs to a special group during process scheduling, this group is taken into account while running the process.
- Nested metadata groups are displayed only if security defined for their parent metadata allows their display
- Added security rule “CUSTOM DATA & ADMINISTRATOR” to make data available to Administrators or depending on some custom user defined rules.

*Based on DSpace commits 3d9df39 (backend) and 9fc7b57 (front-end) related to version 7.0 tag*

## Version 2021.01.00 - Release notes

### DSpace-CRIS 7 2021.01.00 June, 2nd (REST | Angular)

This version fully support the use cases for a modern repository and RIMS / CRIS system. It is target to new users as migration from previous version still require custom data extraction and transformation.

With DSpace-CRIS 7, 4Science is delighted to announce a number of key enhancements which improve the flexibility, integration, data quality and accuracy of DSpace-CRIS:

- Full ORCID v3 integration (push/pull information)
- Integration with dozen external data sources, including commercial ones, to retrieve bibliographic and bibliometric data
- Support for decentralised management, self-service researcher profile management and approval workflows
- Aligned to the latest OpenAIRE Guidelines for Literature Repositories, Data Archives and CRIS Managers
- Data quality tools ensure that your information is always complete and accurate

These enhancements with DSpace-CRIS 7 build on the new DSpace 7 architecture, featuring a new Angular UI and a fully-featured REST API.

#### *Architecture*

- Aligned with DSpace 7
- full REST API with a documented contract
- modern Angular SPA UI

#### *Authentication*

- DSpace-CRIS can be integrated with several Identity Provider ranging from Shibboleth, LDAP, OpenID Connect, ORCID to a local username/password (encrypted) database

#### *Data collection*

- Submission process for all the entities in the OpenAIRE CRIS information space (Publications, Patents, Products, People, OrgUnits, Projects, Fundings, Equipments, Journals)
- Import from external sources available for most entity types: Fundings from OpenAIRE, Patents from the European Patent Office, OrgUnits from Sherpa/RoMEO (publisher), People from ORCID, Publications from ORCID, PubMed, CrossRef, Scopus, Web of Science, OpenAIRE, arXiv, NASA/ADS, CiNii, Scielo, VuFind, PubMed Europe), Journals from Sherpa/RoMEO

- Automatic enrichment of manual submission looking up to the external providers by identifiers
- Bulk operations (creation, update, delete) via xls on all the entities with easy cross-linking (any identifier can be used to link any kind of entities, i.e. publications to a person via an ORCID, staffno, etc) and future reference (link to an entity that is not yet in the system via an identifier that will be resolved later). Bulk operations are validated against your data model, submission configuration and security (mandatory and available fields, relations, etc.)
- Publication Metadata extraction from Scholarly PDF via machine vision (based on the Grobid project)
- Receive automatic alert from compatible providers (OpenAIRE, ORCID) about missing publications or wrong/incomplete data on existing records
- Automatically import new publications for your researcher from Scopus and Web of Science
- Grab bibliometrics data for your publications and authors from Scopus and Web of Science
- Manage complex structured data as nested metadata and ternary relations

*Data presentation*

- define sections and entry points to explore your repository composing configurable widget such as sorted list of objects (Most viewed, Most cited, Recent additions, etc.), infographics for key indicators (number of publications, researchers, etc.), search facets, browse indexes, advanced search form and branding messages
- easily organise your data without code change in tabs and boxes
- include references to linked entities in any entity page (i.e. the list of publications of a researcher, the list of funding received by a project, etc.)
- present search results and linked entities in a graphical way with pie, line, bar charts
- export your researchers information in professional looking PDF/RDF CV
- export details about your other entities (Funding, Projects, Organisations, etc.) in PDF/RDF fact sheets
- export publications data in citation formats (APA, Chicago, MLA, etc.) via CSL
- show the bibliometrics collected for your publications and authors
- include alternative metrics information for your publication from AltMetric and/or Dimension
- ORCID and authenticated ORCID are properly displayed in researcher profiles and linked records (publications, projects, etc.)
- granular visibility at metadata level based on contextual rules (financial data of a funding visible only to the investigators involved in the project, personal contact data only to HR people, etc.)
- rich and extensible usage reports are available for all the entities including direct data (visualization and download) and aggregated data about the linked objects (visualization of researcher's publications, etc.). Data can be visualized in tabular and graphical form with maps and exportable charts (pie, line, bar). Reports can be produced for a specific time frame or since the system setup

*Profile management*

- Researchers can manage directly selected information in their profiles and linked records
- List of linked objects in the profile can be amended, hiding unwanted objects (old research) and forcing a preferred visualization order (selected publications, projects, etc.)
- ORCID Synchronisation: the researcher can connect/disconect her local profile with ORCID to received suggestion about missing publication and push update to the ORCID registry
- ORCID preferences: it is possible to configure which details are synchronised (biographic information, affiliation, qualification, education, publications, funding) setting a manual or automatic (over the night) push

*Data quality & accuracy*

- Identify potential duplicate during the submission and approval workflow
- get flags for unrelated entities or uncertain matches (i.e. not identified authors in a publication, investigator in a project, etc.). Option to automatically create new records for specific entity types or manual curate the authorities
- configurable lookup authorities both internals than externals, such as the personal staff, the ORCID registries, the recorded fundings, the OpenAIRE project database and more
- default to international approved data model (CERIF / OpenAIRE) and controlled vocabularies (COAR)
- retains identifier for external entities for future use and automatic match (i.e. ORCID of external authors)
- enforced validation in bulk operation to guarantee that the record structure always match your definition (i.e. the proper metadata are used according to the entity definition)
- Receive automatic alert from compatible providers (OpenAIRE, ORCID) about missing publications or wrong/incomplete data on existing records
- Automatically ingest publications for your researchers from Scopus and Web of Science
- Configurable workflows by collection and entity types to involve librarians, research officer, legal, ethical and financial department in data input and verification
- Correction workflow to be used by less privileged user to request correction on existing record that need to be moderated
- Easily to monitor and organise tasks queue for approvals, changes in correction requests are highlighted

*Data security*

- enforced granular security at the metadata level across the whole platform: REST API, export and import tool, visualization
- support for partial editing so that researcher can edit some (configurable) information in their profile and their related records without touching master data coming from external systems or under the Institution responsibility
- easily access to an audit log of all the operations performed on a record
- REST API are protected using JWT, SSL, CSRF Token

*Interoperability*

- Connectors to retrieve records (Publication, Person, Funding, OrgUnit, Journal) from 17 external data sources
- Connectors to retrieve bibliometrics data for your publications and authors from Scopus and Web of Science
- Full integration (push/pull) with ORCID via v3 API and support for WebHook (Premium API)
- Aligned to the latest OpenAIRE Guidelines for Literature Repositories (v4, v3), Data Archives (v4 unreleased) and CRIS Managers (v1.1)
- Full REST API
- export options in XML, CERIF XML, XLS for all the entities

## Getting Started

Please follow the basic DSpace installation process, once done:

DSpace-CRIS provides convenient scripts and configuration to quickly start to play with the system.

Entity definitions

```
./dspace dsrun org.dspace.app.util.InitializeEntityTypesOnly -d
```

Relationships used internally by some features (this step would be performed automatically in future)

```
./dspace dsrun org.dspace.app.util.InitializeEntities -f ../config
/entities/correction-relationship-types.xml
./dspace dsrun org.dspace.app.util.InitializeEntities -f ../config
/entities/hide-sort-relationship-types.xml
```

Creation of a communities & collections structure

```
./dspace dsrun org.dspace.administer.StructBuilder -e <admin-email>
  -o /fullpath-to-dspace/log/output-sample-structure.xml
  -f /fullpath-to-dspace/config/sample-structure.xml
```

Proceed configuring your layout and security via the configuration tool

## Upgrade from a previous DSpace-CRIS version

The DSpace-CRIS version 7.0 or better 2021.01.00 has been built on top of DSpace 7.0 instead than on previous DSpace-CRIS version. The basic assumption is to have a DSpace 7.0 working database, this reduce the number of scenarios that need to be supported (migrate a DSpace-CRIS 5 installation that was the result of a DSpace 4 upgrade, a native DSpace-CRIS 6 installation, a DSpace-CRIS installation that was gone trough several upgrade 1.6, 1.8, 3.0, etc).

To make possibile to work on such assumption the upgrade to DSpace-CRIS do the following

- any extra information from previous DSpace-CRIS installation is moved to backup tables. The schema_version table is backup and cleaned;
- the normal DSpace upgrade process run so that we have a working DSpace 7.0 clean database;
- scripts will be provided (in 2021.02.00) to migrated any eventual extra DSpace-CRIS data to the new architecture.

If you want to start the migration process now, the following command is required

```
./dspace database migrate ignore-missing
```

this will update your database and migrate basic DSpace information to the new version.

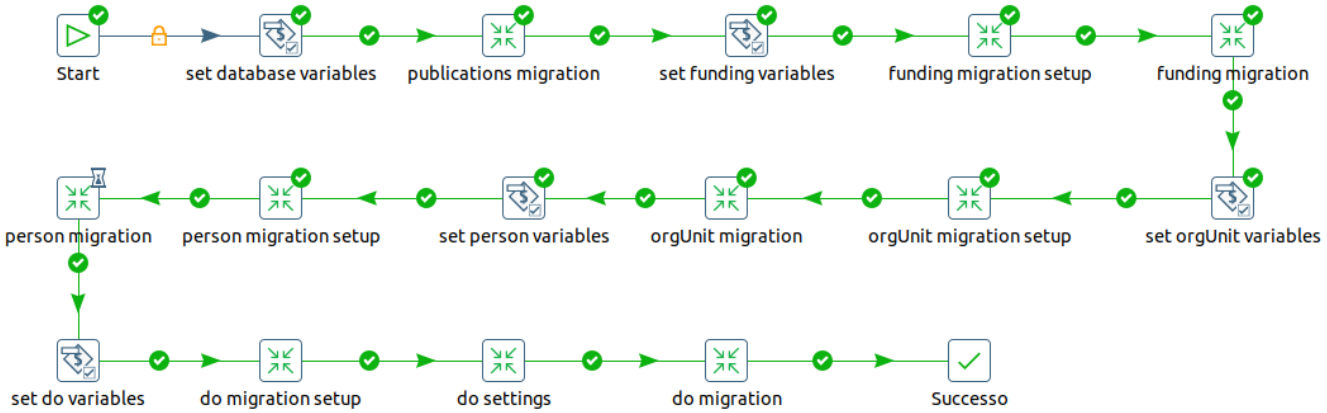
For users coming from a DSpace-CRIS 5.x version there is an additinal procedure available to migrate the additional information that were available in DSpace-CRIS, see Data migration from DSpace-CRIS 5

At this stage there is no such procedure for users coming from DSpace-CRIS 6. Adaption of the procedure available for version 5 are welcome and 4Science is willing to support any institution that would need that alternatively exporting the CRIS data from the previous version in excel, performing some offline manipulation and import them back via the bulk import procedure can be a suitable alternative for no too complex installation.

### Data migration from DSpace-CRIS 5

A Pentaho Data Integration (aka Kettle) ETL Job procedure is provided to migrate data from DSpace-CRIS version 5 to 7. The Job has been developed and verified with PDI 9.0 Community Edition that can be downloaded from here:
https://sourceforge.net/projects/pentaho/files/Pentaho%209.0/client-tools/

To use the tool it is needed to run the `dspace/etc/migration/dspace_cris_migration.kjb`. This can be done opening it in the PDI UI:




or from the command line `kitchen.sh -file:dspace_cris_migration.kjb -param:XXX=VALUE` where the available params (XXX) are described in the table below

| Parameter | Default value | Description |
|---|---|---|
| db_host_name | localhost | the database host name |
| db_name | dspace | the database name |
| db_port_number | 5432 | the database port number |
| db_username | dspace | the user name to use |
| db_password | | the password of the user to use |
| eperson_email | | the email of the eperson to be used |
| file_root | | the root of the bitstream path |

In the same folder of the tool it is included the excel file `migration_configuration.xls` representing the job configuration. The file has several tabs, the ones used for this job are

- **collections**: contains the collection to deposit the new item for each entity type
  - *entity_type*: the entity type
  - *collection_uuid*: the collection uuid where deposit the related entity type
- **metadata**: contains the mapping between metadata used in version 5 and metadata in version 7, based on entity type
  - *entity_type*: the entity type
  - *short_name*: the name of the metadata used in the old DSpace-CRIS
  - *schema:* the schema of the metadata field used in DSpace-CRIS 7
  - *element:* the element of the metadata field used in DSpace-CRIS 7
  - *qualifier:* the qualifier of the metadata field used in DSpace-CRIS 7
  - *language:* the language of the metadata value to be set
- **nested_metadata**: contains the mapping between the nested metadata used in version 5 and metadata in version 7, based on entity type
  - *entity_type*: the entity type
  - *short_name*: the name of the metadata used in the old DSpace-CRIS
  - *schema:* the schema of the metadata field used in DSpace-CRIS 7
  - *element:* the element of the metadata field used in DSpace-CRIS 7
  - *qualifier:* the qualifier of the metadata field used in DSpace-CRIS 7
  - *language:* the language of the metadata value to be set

- **do_types**: contains the mapping between the prefixes of the do entities of DSpace-CRIS 5 and the entity type of DSpace-CRIS 7
  - *entity_type*: the entity type
  - *entity_prefix*: the prefix of the crisid of the entities in the do table
- **metadata_visibility**: contains the mapping between the metadata visibility value of DSpace-CRIS 5 and the values used in DSpace-CRIS 7
  - l*egacy_value:* the metadata value visibility used in DSpace-CRIS 5
  - *new_value:* the metadata value visibility used in DSpace-CRIS 7
- **orcid_scopes**: contains the mapping between the orcid scope properties of DSpace-CRIS 5 and the metadata values used in DSpace-CRIS 7
  - *shortname:* the name of the metadata used in the old DSpace-CRIS related to a specific ORCID scope
  - *schema:* the schema of the metadata field used in DSpace-CRIS 7
  - *element:* the element of the metadata field used in DSpace-CRIS 7
  - *qualifier:* the qualifier of the metadata field used in DSpace-CRIS 7
  - *metadata_value:* the value of the metadata values to set
- **orcid_token**: contains the metadata field used in DSpace-CRIS 7 to store the ORCID access token
  - *schema:* the schema of the metadata field used in DSpace-CRIS 7
  - *element:* the element of the metadata field used in DSpace-CRIS 7
  - *qualifier:* the qualifier of the metadata field used in DSpace-CRIS 7

Before running the job **it is necessary to migrate the DSpace-CRIS 5 database** as described here, so that the table structure is updated and the old tables containing the entities are renamed by adding the prefix **old**_

The tool performs the following actions:

- **updates all the authorities** of the item's metadata values that correspond to a crisid (like ou00002) by adding the prefix **will be referenced::LEGACY-ID::** (like will be referenced::LEGACY-ID::ou00002)
- add the **dspace.entity.type** metadata field to the items without this metadata with **Publication** as value
- for each record in the **old_cris_<entity>** tables it creates a record in the imp_record table and, for each metadata present in the **old_cris_<entity_prefix>_prop old_cris_<entity_prefix>_no_prop** and tables not associated with a bitstream, inserts a record in the imp_metadatavalue table, associating it with the corresponding record in the imp_record table. The metadata present in the properties of the entities representing bitstreams are instead used to populate the imp_bitstream table, specifying the absolute path of the files. The metadata data that instead points to another entity is inserted in the imp_metadatavalue with an authority equal to **will be referenced::LEGACY-ID::<cris-id>**, where <crisid> is the crisid of the entity specified by the pointer
  For the old_cris_do table, in which more types of entities are inserted, a search is made many times, one for each row of the configuration xsl sheet do_type, filtering for the records that have a crisid with a specific prefix. To the metadata present in the properties of the entities are added:
  - the **cris.legayId** metadata field with the crisId as value; this metadata is used to resolve the references specified with will be referenced::LEGACY-ID
  - if the entities are researcher pages the **cris.owner** with the uuid of the eperson specified in the column epersonId of the old_cris_rpage table

#### Post import migration

To migrate the relationships present in **old_cris_relpref**, the records of **old_cris_orcid_history** and the subscriptions present in **old_cris_subscription** and **old_cris_statsubscription** you can use a specific Pentaho job defined by the file `dspace/etc/migration/dspace_cris_migration_post_import.kjb`. That job must be launched after having actually created the migrated items with the procedure described above. The post import migration performs the following actions:

- migrate the relations present in the old_cris_relpref
- migrate the old_cris_orcid_history records
- migrate the old subcriptions
- hide the private entities

The job parameters are:

| Parameter | Default value | Description |
|---|---|---|
| db_host_name | localhost | the database host name |
| db_name | dspace | the database name |
| db_port_number | 5432 | the database port number |
| db_username | dspace | the user name to use |
| db_password | | the password of the user to use |

The same excel file described above (`migration_configuration.xls`) is used to configure the transformation that migrate the relationships through the following sheets:

- **relations**: contains the mapping between the relation type present in the old_cris_relpref table and the discovery id related to the relation in DSpace-CRIS 7
  - *relation_type*: the relation type present in the old_cris_relpref

- *discovery_id*: the discovery id. The leftward_type and the rightward_type of the relationship_type to be used are calculated starting from this discovery id and the status column of the old_cris_relpref table

```
if(status == "selected"){
        var leftward_type = 'is' + discovery_id +
'SelectedFor';
        var rightward_type = 'hasSelected' + discovery_id;
} else {
        var leftward_type = 'is' + discovery_id + 'HiddenFor';
        var rightward_type = 'notDisplaying' + discovery_id;
}
```

#### Hiding of private entities

To hide the entities that was configured to be private the following query is performed by the post import migration:

```
DELETE FROM resourcepolicy
WHERE epersongroup_id = (SELECT uuid FROM epersongroup WHERE name =
'Anonymous')
AND action_id = 0
AND dspace_object IN (
        SELECT item.uuid
        FROM Item item, metadatavalue metadata_value,
metadatafieldregistry metadata_field, metadataschemaregistry
metadata_schema
        WHERE metadata_value.dspace_object_id = item.uuid
        AND metadata_value.metadata_field_id = metadata_field.
metadata_field_id
        AND metadata_field.element = 'legacyId'
        AND metadata_field.qualifier IS null
        AND metadata_field.metadata_schema_id = metadata_schema.
metadata_schema_id
        AND metadata_schema.short_id = 'cris'
        AND metadata_value.text_value IN (
                SELECT crisid FROM old_cris_orgunit WHERE status = false
                UNION
                SELECT crisid FROM old_cris_rpage WHERE status = false
                UNION
                SELECT crisid FROM old_cris_do WHERE status = false
                UNION
                SELECT crisid FROM old_cris_project WHERE status = false
        )
);
```

#### Update item references script

A script is provided to force the lookup of all the authority values `will be referenced` still unresolved. This is convenient when data are manipulated in the database directly or massive operations are performed that would lead to some failure in the synchronous lookup performed by the `referenceresolver` consumer The script can be run as follow

```
./dspace update-item-references
```

#### Summary

In summary, to migrate from version 5 to version 7 you need to:

1. migrate the DSpace-CRIS 5 database as described here
2. run the job `dspace/etc/migration/dspace_cris_migration.kjb` using Pentaho to create the records into imp_record, imp_metadatavalue and imp_bitstream
3. import the entities using the DBMS Import framework ( using the command dsrun org.dspace.app.batch.ItemImportMainOA -E <eperson-email> )
4. run the job `dspace/etc/migration/dspace_cris_migration_post_import.kjb` using Pentaho
5. in some situation, when the import is performed very fast or in multiple thread, some of the will be referenced authorities would be not resolved correctly. To fix that a dspace script named has been created `/dspace/bin/dspace update-item-referencesit` is safe to run the script in all the scenario as if all the reference were already solved the script will do nothing

#### Statistics migration

There is a further Pentaho job to migrate the statistics of a dspace-cris-5 in order to import them on a dspace-cris-7. The job is implemented by the `dspace/etc/migration/dspace_cris_statistics_csv_migration.kjb` script and, in addition to the parameters relating to the connection with the database to be migrated, there is also a parameter called **csv_path** which must be valued with a local path associated with an exported csv file generated by the core statistics of solr.

To generate the csv with the statistics data you can use the following command:

```
curl -X GET -G 'http://${solr-statistics-core}/select' -d 'q=*:
*&fl=continent,submitter,isBot,statistics_type,previousWorkflowStep,
city,latitude,type,owningItem,countryCode,id,owningComm,longitude,
workflowItemId,ip,workflowStep,dns,userAgent,actor,referrer,bundleName,
time,epersonid,owningColl,uid,search.uniqueid&rows=250617&wt=csv' -o
${output_file}
```

where `solr-statistics-core` represents the path of the solr webapplication linked to the statistics core and `output_file` must be the path of the file to be generated.

The script generates a csv file in the same folder as the input file, replacing the old item_id with the new uuid used in version 7. It is then possible to import the csv generated with the following command:

```
curl http://${solr-statistics-core}/update/csv --data-binary
@${input_file} -H 'Content-type:application/csv; charset=utf-8' -o
${result-name}.html
```

where `solr-statistics-core` represents the path of the solr webapplication linked to the statistics core to be populated, `input_file` represents the path of the csv file to be imported and `result-name` must be the name of the file with the response incoming from solr.

#### Redirect access from the legacy CRIS 5 details page to the new CRIS 7 details page

In order to search items using their old crisId a new configuration Spring Bean needs to be used in the discover.xml file under the lucky-search configuration. In the configuration below a new Spring Bean with the id `searchFilterLegacyId` has been added and then referenced unter the `searchFilters` property.

```
    <bean id="lucky" class="org.dspace.discovery.configuration.
DiscoveryConfiguration">
        <property name="indexAlways" value="true"/>
        <!-- filter queries for lucky configuration-->
        <property name="defaultFilterQueries">
```

```
            <list>
                <value>search.resourcetype:Item</value>
            </list>
        </property>
        <property name="searchSortConfiguration">
            <bean class="org.dspace.discovery.configuration.
DiscoverySortConfiguration">
                <!--<property name="defaultSort" ref="sortDateIssued"
/>-->
                <!--DefaultSortOrder can either be desc or asc (desc is
default)-->
                <property name="sortFields">
                    <list>
                        <ref bean="sortTitle"/>
                    </list>
                </property>
            </bean>
        </property>
        <!-- empty list of facets-->
        <property name="sidebarFacets">
            <list>
            </list>
        </property>
        <property name="searchFilters">
            <list>
                ...
                <ref bean="searchFilterLegacyId" />
                ...
            </list>
        </property>
    </bean>

    ...

    <bean id="searchFilterLegacyId" class="org.dspace.discovery.
configuration.DiscoverySearchFilter">
        <property name="indexFieldName" value="legacy-id"/>
        <property name="metadataFields">
            <list>
                <value>cris.legacyId</value>
            </list>
        </property>
        <property name="isOpenByDefault" value="true"/>
        <property name="pageSize" value="10"/>
    </bean>

    ...

    </bean>
```

This Spring Bean allows to configure the lucky-search fon an item using the legacy-id metadata (old crisId field). Once done with the configuring of the lucky-search, a new rewrite rule nees to be configured on the apache/nginx side. These rules allow to forward from the old paths defined as:

```
https://.../cris/xxxx/yyyyy
```

or

```
https://.../cris/xxxx/yyyyy/zzzzz.html
```

to the new one defined as:

```
https://.../lucky-search?index=legacy-id&value=yyyyy
```

where yyyyy is the legacy-id value.

To configure the forwarding rules, please check the Apache documentation at the following links:

https://httpd.apache.org/docs/current/mod/mod_rewrite.html

https://httpd.apache.org/docs/2.4/rewrite/remapping.html

## General site navigation

### Explore sections

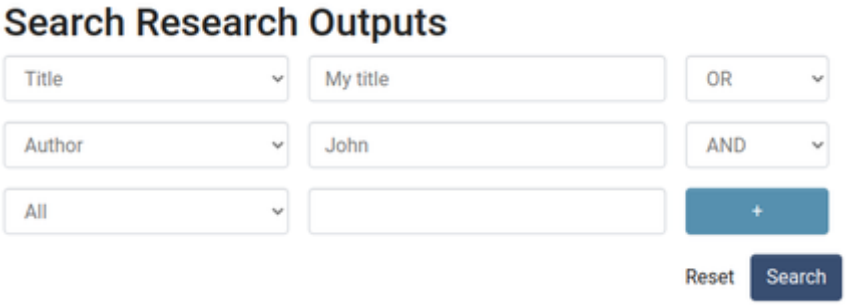
The explore sections allow to highlight specific area of the database defining an entry page for the entity's types present in DSpace CRIS (publications, organizations, researchers, etc.) and can represent a starting point for more in-depth research. It is possible to configure which entity's type should have a dedicated exploration page and it is also possible to establish which components should compose that page.

#### Section components

The explore page consists on a set of configurable components from those available. It is possible to choose between 4 different components to be placed on the page:

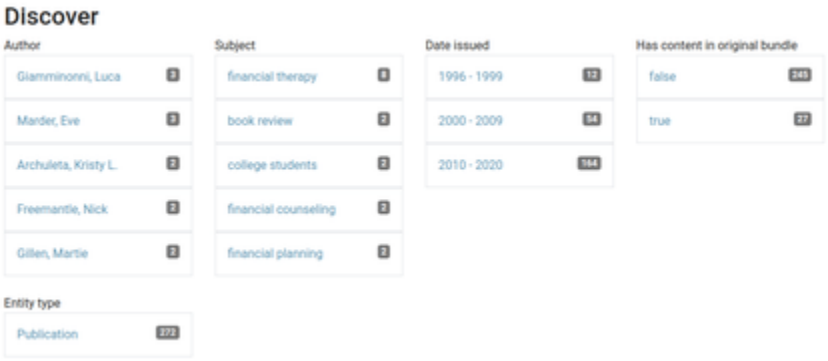
- **Browse component**: component consisting of a list of links that allow an entities browsing according to a specific configurable strategy (such as browse publications by author or browse projects by title).


- **Search component:** component with which it is possible to search for entities of the given type by filtering for additional fields. The filters applicable to the search can be linked with AND OR and NOT and are configurable.




- **Top component:** component that shows the first n items in the system ordered according to a certain criterion (such as sorting publications according to the last access date). The sort field and the order (ascending or descending) are configurable.


- **Facet component:** component composed of a series of configurable views that allow to highlight the occurrences of a certain field within the entities present in the system. For example, considering the publications, it is possible to configure facets to show the cardinalities of the authors highlighting those with a higher count.




In addition to being able to configure the components themselves, it is also possible to configure which ones to show, how to arrange them on the page and how many facet box per row to display.

- **Infographic (counters) component:** component composed by counters decorated with a custom icon, each one displaying the number of results returned by a query.


- **Text/Html component:** component containing a simple text, or its I18n key, or a link to an image, to be displayed as static content within explore section.

#### Configuration

For each section it is possible to define which browses proposes and which dynamic components include. The system looks up for a Singleton Service **CrisLayoutSectionServiceImpl** defined in **[dspace-installDir]/config/spring/api/cris-sections.xml** which manages a key-value map between the entity type that the section configuration refers to and a list of **CrisLayoutSectionComponent.** Each CrisLayoutSectionComponent then has a specialization for each possible type: CrisLayoutTopComponent, CrisLayoutBrowseComponent, CrisLayoutSearchComponent and CrisLayoutFacetComponent.

The map values therefore define which entities to associate an explore page with, while the list of components defines which sections must make up the page itself.

An example of configuration:

```
<bean class="org.dspace.layout.service.impl.
CrisLayoutSectionServiceImpl" >
    <property name="componentMap">
      <map>
        <entry key="publications">
```

```
            <list>
              <list>
                <bean class="org.dspace.layout.CrisLayoutBrowseComponent">
                  <property name="browseNames">
                    <list>
                      <value>rodept</value>
                      <value>author</value>
                      <value>title</value>
                      <value>type</value>
                      <value>dateissued</value>
                      <value>subject</value>
                    </list>
                  </property>
                  <property name="style" value="col-md-4"/>
                </bean>
                <bean class="org.dspace.layout.CrisLayoutSearchComponent">
                  <property name="discoveryConfigurationName" value="
publication" />
                  <property name="style" value="col-md-8"/>
                </bean>
              </list>
              <list>
                <bean class="org.dspace.layout.CrisLayoutTopComponent">
                  <property name="discoveryConfigurationName" value="
publication" />
                  <property name="sortField" value="dc.date.accessioned"
/>
                  <property name="order" value="desc" />
                  <property name="style" value="col-md-6"/>
                </bean>
                <bean class="org.dspace.layout.CrisLayoutTopComponent">
                  <property name="discoveryConfigurationName" value="
publication" />
                  <property name="sortField" value="dc.title" />
                  <property name="order" value="asc" />
                  <property name="style" value="col-md-6"/>
                </bean>
              </list>
              <list>
                <bean class="org.dspace.layout.CrisLayoutFacetComponent">
                  <property name="discoveryConfigurationName" value="
publication" />
                  <property name="style" value="col-md-12"/>
                </bean>
             </list>
            </list>
          </entry>
          <entry key="researcherprofiles">
            <list>
              <list>
```

```
              <bean class="org.dspace.layout.CrisLayoutBrowseComponent">
                <property name="browseNames">
                  <list>
                    <value>rpname</value>
                    <value>rpdept</value>
                  </list>
                </property>
                <property name="style" value="col-md-4"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutSearchComponent">
                <property name="discoveryConfigurationName" value="
person" />
                <property name="style" value="col-md-8"/>
              </bean>
            </list>
          </list>
        </entry>
        <entry key="orgunits">
          <list>
            <list>
              <bean class="org.dspace.layout.CrisLayoutBrowseComponent">
                <property name="browseNames">
                  <list>
                    <value>ouname</value>
                  </list>
                </property>
                <property name="style" value="col-md-4"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutSearchComponent">
                  <property name="discoveryConfigurationName" value="
organization" />
                  <property name="style" value="col-md-8"/>
              </bean>
            </list>
            <list>
                <bean class="org.dspace.layout.CrisLayoutFacetComponent"
>
                  <property name="discoveryConfigurationName" value="
organization" />
                  <property name="style" value="col-md-12"/>
                </bean>
            </list>
          </list>
        </entry>
        <entry key="fundings">
          <list>
            <list>
              <bean class="org.dspace.layout.CrisLayoutBrowseComponent">
                <property name="browseNames">
                  <list>
```

```
                    <value>pjtitle</value>
                  </list>
                </property>
                <property name="style" value="col-md-4"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutSearchComponent">
                  <property name="discoveryConfigurationName" value="
project" />
                <property name="style" value="col-md-8"/>
              </bean>
            </list>
          </list>
        </entry>
        <entry key="site">
          <list>
            <list>
              <bean class="org.dspace.layout.CrisLayoutTextRowComponent"
>
                <property name="order" value="0"/>
                <property name="contentType" value="image"/>
                <property name="content" value="<image url>"/>
                <property name="style" value="col-md-12 py-5 w-50
center"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutTextRowComponent"
>
                <property name="order" value="1"/>
                <property name="contentType" value="text-key"/>
                <property name="content" value="<I18n key>"/>
                <property name="style" value="col-md-12 h2 d-flex
justify-content-center py-3"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutTextRowComponent"
>
                <property name="order" value="1"/>
                <property name="contentType" value="text-raw"/>
                <property name="content" value="<raw text to be
displayed>"/>
                <property name="style" value="col-md-12 h2 d-flex
justify-content-center py-3"/>
              </bean>
            </list>
            <list>
              <bean class="org.dspace.layout.CrisLayoutSearchComponent">
                <property name="discoveryConfigurationName" value="
site" />
                <property name="style" value="col-md-12"/>
                <property name="searchType" value="basic"/>
                <property name="displayTitle" value="false"/>
              </bean>
```

```
            </list>
          <list>
            <bean class="org.dspace.layout.CrisLayoutCountersComponent">
              <property name="style" value="col-md-12 py-4"/>
                <property name="counterSettingsList">
                  <list>
                    <bean class="org.dspace.layout.
CrisLayoutCountersComponent.CounterSettings">
                      <property name="discoveryConfigurationName"
value="researchoutputs"/>
                      <property name="label" value="publications"/>
                      <property name="icon" value="fas fa-book fa-3x"/>
                      <property name="link" value="/explore
/researchoutputs"/>
                    </bean>
                    <bean class="org.dspace.layout.
CrisLayoutCountersComponent.CounterSettings">
                      <property name="discoveryConfigurationName"
value="project_funding"/>
                      <property name="label" value="project_funding"/>
                      <property name="icon" value="fas fa-project-
diagram fa-3x"/>
                      <property name="link" value="/explore
/fundings_and_projects"/>
                    </bean>
                    <bean class="org.dspace.layout.
CrisLayoutCountersComponent.CounterSettings">
                      <property name="discoveryConfigurationName"
value="person"/>
                      <property name="label" value="rprofiles"/>
                      <property name="icon" value="fas fa-graduation-
cap fa-3x"/>
                      <property name="link" value="/explore
/researcherprofiles"/>
                    </bean>
                  </list>
                </property>
              </bean>
            </list>
            <list>
              <bean class="org.dspace.layout.CrisLayoutTopComponent">
                <property name="discoveryConfigurationName" value="
researchoutputs" />
                <property name="sortField" value="metric.view" />
                <property name="order" value="desc" />
                <property name="style" value="col-md-4"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutTopComponent">
                <property name="discoveryConfigurationName" value="
researchoutputs" />
```

```
                <property name="sortField" value="metric.download" />
                <property name="order" value="desc" />
                <property name="style" value="col-md-4"/>
              </bean>
              <bean class="org.dspace.layout.CrisLayoutTopComponent">
                <property name="discoveryConfigurationName" value="
researchoutputs" />
                <property name="sortField" value="metric.scopus.
citation" />
                <property name="order" value="desc" />
                <property name="style" value="col-md-4"/>
              </bean>
            </list>
          </list>
        </entry>
      </map>
    </property>
  </bean>
```

Each component has the **style** property with which it is possible to set the css classes that the component will have a displayed format. Furthermore, each component according to the type has other properties:

- **CrisLayoutBrowseComponent**: has the `browseNames` property that defines the set of browse-related names applicable on the entity.
- **CrisLayoutSearchComponent**: has the `discoveryConfigurationName`property that represent the name of the discovery configuration related to the search ; that configuration defines the applicable filters. Optional parameters can be defined: `searchType` with value 'basic' or 'advanced' (default) defining whether displayed box should consist in a single input box or many containing multiple statements that can be combined with AND, OR, NOT keywords. Initial number of box to be displayed, can be configured with `initialStatements` property, default value is 3
- **CrisLayoutTopComponent:** has the `discoveryConfigurationName`property and also the `sortField` and `order` properties to define the order criteria.
- **CrisLayoutFacetComponent:** has the `discoveryConfigurationName`property and the optional `facetsPerRow` defining how many facet box UI will display, default value is 4.
- **CrisLayoutTextRowComponent**: has the order property, defining the `order` on which the content should appear among other text row elements in the same list of cris layout elements, `content` property, defining the static content to be displayed, while `contentType` defines the type of content and can be set to 'image', meaning that content is an image url, 'text-raw' meaning that content is a static text to be displayed and 'text-key' meaning that actual text to be displayed should be rendered using UI's I18n logic.
- **CrisLayoutCountersComponent**: has `counterSettingsList` property, each element representing a counter to be displayed by this infographic and having following properties: `discoveryConfigurationName` discovery configuration to be used to count by query elements, `icon` font awesome reference to icon to be displayed (i.e. `fas fa-book fa-3x`), `label` text key to be displayed together with the icon, `link`(optional) link to be followed when icon is clicked.

It is possible to get the configuration using the specific REST endpoint **layout/sections** to obtain all the configured sections and specifically the REST endpoint **layout/sections/:id** to obtain the configuration of a single section. The id to be set for this last endpoint is one of the keys configured in the map managed by `CrisLayoutSectionServiceImpl`.

On Angular side, the explore page corresponds to the **explore.component.ts** component, while the components that make up the various sections are implemented through the **browse-section.component.ts**, **facet-section.component.ts**, **search-section.component.ts** and **top-section.component.ts**. In the explore.component.html page, the various components returned by the call to the layout/sections/:id endpoint are arranged by row and the choice of which component to display is made based on the value of the componentType attribute.

#### Extension of components

To extend the list of components that can be placed within the exploration pages, the following changes must be made:

- REST side:
  - add a class that implements the **org.dspace.layout.CrisLayoutSectionComponent** interface
  - add an inner class into **org.dspace.app.rest.model.CrisLayoutSectionRest** that implements the **CrisLayoutSectionComponentRest** interface;
  - add a Spring bean that implements the **org.dspace.layout.CrisLayoutSectionComponent** interface; this new bean will convert between the class implementing CrisLayoutSectionComponent and its REST representation that implements CrisLayoutSectionComponentRest
  - use the new component in the cris-sections.xml configuration

- Angular side:
    - add a new component/template pair for the new component to be created
    - edit the explore.component.html template to add a new case to the switch by componentType to display the new component.

## The Discovery module

The Discovery Module enables faceted searching & browsing for your repository.

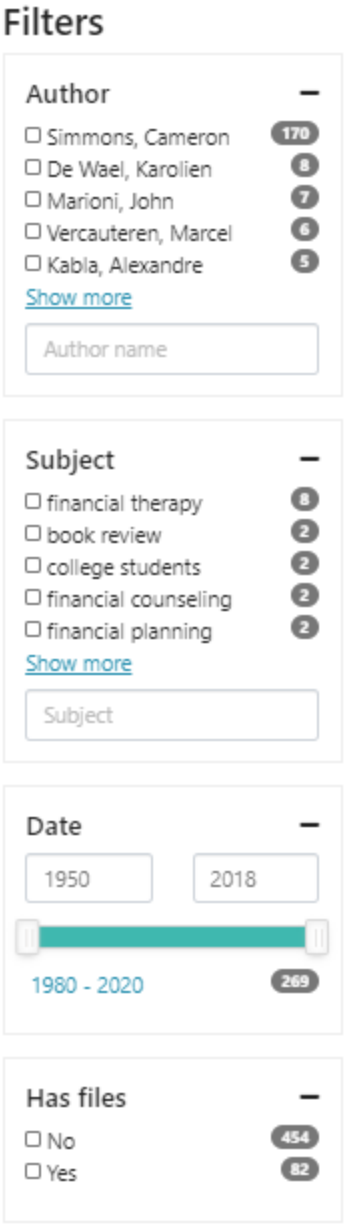
These techniques  might feel familiar from other platforms like Aquabrowser or Amazon, where facets help you to select the right product according to facets like price and brand. DSpace Discovery offers very powerful browse and search configurations that were only possible with code customization in the past.

#### Sidebar Facet

From the user perspective, faceted search breaks up search results into multiple categories, typically showing counts for each, and allows the user to "drill down" or further restrict their search results based on those facets.

When you have successfully enabled Discovery in your DSpace, you will notice that the different enabled facets are visualized in a "Filters" section in your sidebar.

It's important to know that multiple metadata fields can be included in one facet.




Another important property of Sidebar Facets is that their contents are automatically updated to the context of the page. On collection homepages or community homepages it will include information about the items included in that particular collection or community.

**Search Filter**

In a standard search operation, a user specifies his complete query prior to launching the operation. If the results are not satisfactory, the user starts over again with a (slightly) altered query.

In a faceted search, a user can modify the list of displayed search results by specifying additional "filters" that will be applied on the list of search results.

## Configuration files

The configuration for discovery is located in 2 separate files.

- General settings: The `discovery.cfg` file located in the `[dspace-install-dir]/config/modules directory`.
- User Interface Configuration: The `discovery.xml` file is located in `[dspace-install-dir]/config/spring/api/` directory.

## General Discovery settings (config/modules/discovery.cfg)

The discovery.cfg file is located in the [dspace]/config/modules directory and contains following properties. Any of these properties may be overridden in your local.cfg

| Property: | **discovery.search.server** |
|---|---|
| Example Value: | `discovery.search.server=[http://localhost:8080/solr/search]` |
| Informational Note: | Discovery relies on a Solr index for storage and retrieval of its information. This parameter determines the location of the Solr index.<br><br>If you are uncertain whether this property is set correctly, you can use a commandline tool like "wget" to perform a query against the Solr index (and ensure Solr responds). For example, the below query searches the Solr index for "test" and returns the response on standard out:<br><br>`wget -O - http://localhost:8080/solr/search/select?q=test` |
| Property: | **discovery.index.authority.ignore[.field]** |
| Example Value: | `discovery.index.authority.ignore=true`<br><br>`discovery.index.authority.ignore.dc.contributor.author=false` |
| Informational Note: | By default, Discovery will use the authority information in the metadata to disambiguate homonyms. Setting this property to false will make the indexing process the same as the metadata doesn't include authority information. The configuration can be different on a field (<schema>.<element>.<qualifier>) basis, the property without field set the default value. |
| Property: | **discovery.index.authority.ignore-prefered[.field]** |
| Example Value: | `discovery.index.authority.ignore-prefered=true`<br><br>`discovery.index.authority.ignore-prefered.dc.contributor.author=false` |
| Informational Note: | By default, Discovery will use the authority information in the metadata to query the authority for the preferred label. Setting this property to false will make the indexing process the same as the metadata doesn't include authority information (i.e. the preferred form is the one recorded in the metadata value). The configuration can be different on a field (<schema>.<element>.<qualifier>) basis, the property without field set the default value. **If the authority is a remote service, disabling this feature can greatly improve performance.** |
| Property: | **discovery.index.authority.ignore-variants[.field]** |
| Example Value: | `discovery.index.authority.ignore-variants=true`<br><br>`discovery.index.authority.ignore-variants.dc.contributor.author=false` |

| | |
|---|---|
| Informational Note: | By default, Discovery will use the authority information in the metadata to query the authority for variants. Setting this property to false will make the indexing process the same, as the metadata doesn't include authority information. The configuration can be different on a per-field (<schema>.<element>.<qualifier>) basis, the property without field set the default value. If authority is a remote service, disabling this feature can greatly improve performance. |

## Modifying the Discovery User Interface (config/spring/api/discovery.xml)

**Structure Summary**

The configurations are organized together in beans, depending on the purpose these properties are used for.

This purpose can be derived from the class of the beans. Here's a short summary of classes you will encounter throughout the file and what the corresponding properties in the bean are used for.

| | |
|---|---|
| Class: | **DiscoveryConfigurationService** |
| Purpose: | Defines the mapping between separate Discovery configurations and individual collections/communities |
| Default: | All communities, collections and the homepage (key=default) are mapped to defaultConfiguration, also controls the metadata fields that should not be indexed in the search core (item provenance for example). |
| Class: | **DiscoveryConfiguration** |
| Purpose: | Groups configurations for sidebar facets, search filters, search sort options and recent submissions |
| Default: | There is one configuration by default called defaultConfiguration |
| Class: | **DiscoverySearchFilter** |
| Purpose: | Defines that specific metadata fields should be enabled as a search filter |
| Default: | dc.title, dc.contributor.author, dc.creator, dc.subject.* and dc.date.issued are defined as search filters |
| Class: | **DiscoverySearchFilterFacet** |
| Purpose: | Defines which metadata fields should be offered as a contextual sidebar browse options, each of these facets has also got to be a search filter |
| Default: | dc.contributor.author, dc.creator, dc.subject.* and dc.date.issued |
| Class: | **HierarchicalSidebarFacetConfiguration** |
| Purpose: | Defines which metadata fields contain hierarchical data and should be offered as a contextual sidebar option |
| Class: | **DiscoverySortConfiguration** |
| Purpose: | Further specifies the sort options to which a DiscoveryConfiguration refers |
| Default: | dc.title and dc.date.issued are defined as alternatives for sorting, other than Relevance (hard-coded) |

| Class: | **DiscoverySortFunctionConfiguration** |
|---|---|
| Purpose: | Allow to use solr functions to define results sort (i.e. termfreq) |
| Default: | Default is not provided as a solr function should be specified. This Bean takes as properties: name of the solr function (“function”), id of the Bean (“id”) and list of arguments to be passed to the solr function (“arguments”) |
| Class: | **DiscoveryHitHighlightingConfiguration** |
| Purpose: | Defines which metadata fields can contain hit highlighting & search snippets |
| Default: | dc.title, dc.contributor.author, dc.subject, dc.description.abstract & full text from text files. |
| Class: | **TagCloudFacetConfiguration** |
| Purpose: | Defines the tag cloud appearance configuration bean and the search filter facets to appear in the tag cloud form. You can have different "**TagCloudFacetConfiguration**" per community or collection or the home page |

#### Default Settings

In addition to the summarized descriptions of the default values, following details help you to better understand these defaults. If you haven't already done so, download the configuration file and review it together with the following parameters.

The file contains one default configuration that defines following sidebar facets, search filters, sort fields and recent submissions display:

- Sidebar facets
  - **searchFilterAuthor:** groups the metadata fields dc.contributor.author & dc.creator with a facet limit of 10, sorted by occurrence count
  - **searchFilterSubject:** groups all subject metadata fields (dc.subject.*) with a facet limit of 10, sorted by occurrence count
  - **searchFilterIssued:** contains the dc.date.issued metadata field, which is identified with the type "date" and sorted by specific date values
- Search filters
  - **searchFilterTitle:** contains the dc.title metadata field
  - **searchFilterAuthor:** contains the dc.contributor.author & dc.creator metadata fields
  - **searchFilterSubject:** contains the dc.subject.* metadata fields
  - **searchFilterIssued:** contains the dc.date.issued metadata field with the type "date"
- Sort fields
  - **sortTitle**: contains the dc.title metadata field
  - **sortDateIssued:** contains the dc.date.issued metadata field, this sort has the type date configured.
- defaultFilterQueries
  - The default configuration contains no defaultFilterQueries
  - The default filter queries are disabled by default but there is an example in the default configuration in comments which allows discovery to only return items (as opposed to also communities/collections).
- Recent Submissions
  - The recent submissions are sorted by dc.date. accessioned which is a date and a maximum number of 5 recent submissions are displayed.
- Hit highlighting
  - The fields dc.title, dc.contributor.author & dc.subject can contain hit highlighting.
  - The dc.description.abstract & full text field are used to render search snippets.
- Non indexed metadata fields
  - **Community/Collections:** dc.rights (copyright text)
  - **Items:** dc.description.provenance

Many of the properties contain lists that use references to point to the configuration elements. This way a certain configuration type can be used in multiple discovery configurations so there is no need to duplicate them.

#### Non indexed metadata fields

The discovery.xml file has configuration to not index certain metadata fields for communities/collections/items. The configuration is handled in the "toIgnoreMetadataFields" property located in the "org.dspace.discovery.configuration.DiscoveryConfigurationService" bean. Below is an example configuration that excludes dc.description.provenance for items & dc.rights for communities/collections:

```
<property name="toIgnoreMetadataFields">
    <map>
        <entry>
            <key><util:constant static-field="org.dspace.core.Constants.COMMUNITY"/></key>
            <list>
                <!--Introduction text-->
                <!--<value>dc.description</value>-->
                <!--Short description-->
                <!--<value>dc.description.abstract</value>-->
                <!--News-->
                <!--<value>dc.description.tableofcontents</value>-->
                <!--Copyright text-->
                <value>dc.rights</value>
                <!--Community name-->
                <!--<value>dc.title</value>-->
            </list>
        </entry>
        <entry>
            <key><util:constant static-field="org.dspace.core.Constants.COLLECTION"/></key>
            <list>
                <!--Introduction text-->
                <!--<value>dc.description</value>-->
                <!--Short description-->
                <!--<value>dc.description.abstract</value>-->
                <!--News-->
                <!--<value>dc.description.tableofcontents</value>-->
                <!--Copyright text-->
                <value>dc.rights</value>
                <!--Collection name-->
                <!--<value>dc.title</value>-->
            </list>
        </entry>
        <entry>
            <key><util:constant static-field="org.dspace.core.Constants.ITEM"/></key>
            <list>
                <value>dc.description.provenance</value>
            </list>
```

```
        </entry>
    </map>
</property>
```

By adding additional values to the appropriate lists additional metadata can be excluded from the search core, a reindex is required after altering this file to ensure that the values are removed from the index.

#### Search filters & sidebar facets Customization

This section explains the properties for search filters & sidebar facets. Each sidebar facet must occur in the reference list of the search filters. Below is an example configuration of a search filter that is not used as a sidebar facet.

```
<bean id="searchFilterTitle" class="org.dspace.discovery.configuration.DiscoverySearchFilter">
    <property name="indexFieldName" value="title"/>
    <property name="metadataFields">
        <list>
            <value>dc.title</value>
        </list>
    </property>
</bean>
```

The id & class attributes are mandatory for this type of bean. The properties that it contains are discussed below.

- **indexFieldName** (Required)**:** A unique search filter name, the metadata will be indexed in Solr under this field name.
- **metadataFields** (Required): A list of the metadata fields that need to be included in the facet.

Sidebar facets extend the search filter and add some extra properties to it, below is an example of a search filter that is also used as a sidebar facet.

```
<bean id="searchFilterAuthor" class="org.dspace.discovery.configuration.SidebarFacetConfiguration">
        <property name="indexFieldName" value="author"/>
        <property name="metadataFields">
            <list>
                <value>dc.contributor.author</value>
                <value>dc.creator</value>
            </list>
        </property>
        <property name="facetLimit" value="10"/>
        <property name="sortOrder" value="COUNT"/>
        <property name="type" value="text"/>
    </bean>
```

Note that the class has changed from **DiscoverySearchFilter** to **SidebarFacetConfiguration** this is needed to support the extra properties.

- **facetLimit** (optional)**:** The maximum number of values to be shown. This property is optional, if none is specified the default value "10" will be used. If the filter has the type **date**, this property will not be used since dates are automatically grouped together.
- **sortOrder** (optional)**:**The sort order for the sidebar facets, it can either be COUNT or VALUE. The default value is COUNT.
    - **COUNT** Facets will be sorted by the amount of times they appear in the repository
    - **VALUE** Facets will be sorted alphabetically

- **type**(optional): the type of the sidebar facet it can either be "date" or "text", "text" is the default value.
    - **text:** The facets will be treated as is
    - **date**: Only the year will be stored in the Solr index. These years are automatically displayed in ranges that get smaller when you select one.

*Hierarchical (taxonomies based) sidebar facets*

Discovery supports specialized drill down in hierarchically structured metadata fields. For this drill down to work, the metadata in the field for which you enable this must be composed out of terms, divided by a splitter. For example, you could have a dc.subject.taxonomy field in which you keep metadata like "CARTOGRAPHY::PHOTOGRAMMETRY", in which Cartography and Photogrammetry are both terms, divided by the splitter "::". The sidebar will only display the top level facets, when clicking on view more all the facet options will be displayed.

```
<bean id="searchFilterSubject" class="org.dspace.discovery.configuration.HierarchicalSidebarFacetConfiguration">
   <property name="indexFieldName" value="subject"/>
   <property name="metadataFields">
      <list>
         <value>dc.subject</value>
      </list>
   </property>
   <property name="sortOrder" value="COUNT"/>
   <property name="splitter" value="::"/>
   <property name="skipFirstNodeLevel" value="false"/>
</bean>
```

Note that the class has changed from **SidebarFacetConfiguration** to **HierarchicalSidebarFacetConfiguration** this is needed to support the extra properties.

- **splitter** (required)**:** The splitter used to split up the separate nodes
- **skipFirstNodeLevel** (optional)**:** Whether or not to show the root node level. For some hierarchical data there is a single root node. In most cases it doesn't need to be shown since it isn't relevant. **This property is true by default.**

**Sort option customization for search results**

This section explains the properties of an individual SortConfiguration, like sortTitle and sortDateIssued from the default configuration. In order to create custom sort options, you can either modify specific properties of those that already exist or create a totally new one from scratch.

Here's what the sortTitle SortConfiguration looks like:

```
<bean id="sortTitle" class="org.dspace.discovery.configuration.DiscoverySortFieldConfiguration">
      <property name="metadataField" value="dc.title"/>
      <property name="type" value="text"/>
 </bean>
```

The id & class attributes are mandatory for this type of bean. The properties that it contains are discussed below.

- **metadataField** (Required): The metadata field indicating the sort values
- **type** (optional): the type of the sort option can either be date or text, if none is defined text will be used.

**DiscoveryConfiguration**

The DiscoveryConfiguration Groups configurations for sidebar facets, search filters, search sort options and recent submissions. If you want to show the same sidebar facets, use the same search filters, search options and recent submissions everywhere in your repository, you will only need one DiscoveryConfiguration and you might as well just edit the defaultConfiguration.

The DiscoveryConfiguration makes it very easy to use custom sidebar facets, search filters, ... on specific communities or collection homepage. This is particularly useful if your collections are heterogeneous. For example, in a collection with conference papers, you might want to offer a sidebar facet for conference date, which might be more relevant than the actual issued date of the proceedings. In a collection with papers, you might want to offer a facet for funding bodies or publisher, while these fields are irrelevant for items like learning objects.

A DiscoveryConfiguration consists out of five parts

- The list of applicable sidebarFacets
- The list of applicable searchFilters
- The list of applicable searchSortFields
- Any default filter queries (optional)
- The configuration for the Recent submissions display
- The configuration of the tag cloud facet

### *Configuring lists of sidebarFacets and searchFilters*

Below is an example of how one of these lists can be configured. It's important that each of the bean references corresponds to the exact name of the earlier defined facets, filters or sort options.

```
<property name="sidebarFacets">
    <list>
        <ref bean="sidebarFacetAuthor" />
        <ref bean="sidebarFacetSubject" />
        <ref bean="sidebarFacetDateIssued" />
    </list>
</property>
```

### *Configuring and customizing search sort fields*

The search sort field configuration block contains the available sort fields and the possibility to configure a default sort field and sort order.

Below is an example of the sort configuration.

```
<property name="searchSortConfiguration">
    <bean class="org.dspace.discovery.configuration.DiscoverySortConfiguration">
        <!--<property name="defaultSort" ref="sortDateIssued"/>-->
        <!--DefaultSortOrder can either be desc or asc (desc is default)-->
        <property name="defaultSortOrder" value="desc"/>
        <property name="sortFields">
            <list>
                <ref bean="sortTitle" />
                <ref bean="sortDateIssued" />
            </list>
        </property>
```

```
    </bean>
</property>
```

The property name & the bean class are mandatory. The property field names are discusses below.

- **defaultSort** (optional): The default field on which the search results will be sorted, this must be a reference to an existing search sort field bean. If none is given relevance will be the default. Sorting according to the internal relevance algorithm is always available, even though it's not explicitly mentioned in the sortFields section.
- **defaultSortOrder** (optional): The default sort order can either be asc or desc.
- **sortFields** (mandatory): The list of available sort options, each element in this list must link to an existing sort field configuration bean.

### *Access Rights Awareness - technical details*

The *DSpaceObject* class has an *updateLastModified()* method which will be triggered each time an authorization policy changes. This method is only implemented in the item class where the last_modified timestamp will be updated and a modify event will be fired. By doing this we ensure that the discovery consumer is called and the item is reindexed. Since this feature can be switched off a separate plugin has been created: the *SolrServiceResourceRestrictionPlugin.* Whenever we reindex a DSpace object all the read rights will be stored in the read field. We make a distinction between groups and users by adding a '*g*' prefix for groups and the '*e*' prefix for epersons.

When searching in discovery all the groups the user belongs to will be added as a filter query as well as the users identifier. If the user is an admin all items will be returned since an admin has read rights on everything.

### *"More like this" configuration*

The "more like this"-configuration element contains all the settings for displaying related items on an item display page.

Below is an example of the "more like this" configuration.

```
<property name="moreLikeThisConfiguration">
    <bean class="org.dspace.discovery.configuration.DiscoveryMoreLikeThisConfiguration">
        <property name="similarityMetadataFields">
            <list>
                <value>dc.title</value>
                <value>dc.contributor.author</value>
                <value>dc.creator</value>
                <value>dc.subject</value>
            </list>
        </property>
        <!--The minimum number of matching terms across the metadata fields above before an item is found as related -->
        <property name="minTermFrequency" value="5"/>
        <!--The maximum number of related items displayed-->
        <property name="max" value="3"/>
        <!--The minimum word length below which words will be ignored-->
        <property name="minWordLength" value="5"/>
    </bean>
</property>
```

The property name & the bean class are mandatory. The property field names are discussed below.

- similarityMetadataFields: the metadata fields checked for similarity
- minTermFrequency: The minimum number of matching terms accross the metadata fields above before an item is found as related
- max: The maximum number of related items displayed
- minWordLength: The minimum word length below which words will be ignored

"More like this" technical details

The *org.dspace.discovery.SearchService* object has received a *getRelatedItems()* method. This method requires an item & the more-like-this configuration bean from above. This method is implemented in the *org.dspace.discovery.SolrServiceImpl* which uses the item as a query & uses the default Solr parameters for more-like-this to pass the bean configuration to solr (https://cwiki.apache.org/confluence/display/solr/MoreLikeThis) . The result will be a list of items or if none found an empty list. The rendering of this list is handled in the *org.dspace.app.xmlui.aspect.discovery.RelatedItems* class.

*"Did you mean" spellcheck aid for search technical details*

Similar to the More like this configuration, SOLR's spell check component is used with default configuration values. Any of these values can be overridden in the solrconfig.xml file located in dspace/solr/search/conf/. Following links provide more information about the SOLR SpellCheckComponent:

http://wiki.apache.org/solr/SpellCheckComponent

https://cwiki.apache.org/confluence/display/solr/Spell+Checking

## Discovery Solr Index Maintenance

| Command used: | `[dspace]/bin/dspace index-discovery [-cbhf[r <item handle>]]` |
|---|---|
| Java class: | org.dspace.discovery.IndexClient |
| Arguments (short and long forms): | Description |
| | called without any options, will update/clean an existing index |
| • b | (re)build index, wiping out current one if it exists |
| • c | clean existing index removing any documents that no longer exist in the db |
| • f | if updating existing index, force each handle to be reindexed even if uptodate |
| • h | print this help message |
| • i <object handle> | Reindex an individual object (and any child objects). When run on an Item, it just reindexes that single Item. When run on a Collection, it reindexes the Collection itself and all Items in that Collection. When run on a Community, it reindexes the Community itself and all sub-Communities, contained Collections and contained Items. |
| • o | optimize search core |
| • r <item handle> | remove an Item, Collection or Community from index based on its handle |
| • s | Rebuild the spellchecker, can be combined with -b and -f. |

t is recommended to run maintenance on the Discovery Solr index occasionally (from crontab or your system's scheduler), to prevent your servlet container from running out of memory:

[dspace]/bin/dspace index-discovery -o

(Since Solr 4, the underlying optimize operation has been discouraged as mostly unnecessary and renamed. See https://issues.apache.org/jira/browse/SOLR-3141).

## Advanced Solr Configuration

Discovery is built as an application layer on top of the Solr open source enterprise search server. Therefore, Solr configuration can be applied to the Solr cores that are shipped with DSpace.

The DSpace Solr instance currently runs several cores (which means indexes in Solr parlance). The "statistics" core is for collection of DSpace usage events for statistical purposes (if you have been collecting statistics for multiple years, you may have chosen to use sharding and you will see one core per each year collected). The "search" core is used by Discovery for for search and faceting, for displaying the collection /community hierarchy and item counts. The "authority" core is used by SolrAuthority to store information about authors, including their data imported from the ORCID registry.

solr

solr.xml

search

conf

admin-extra.html

elevate.xml

protwords.txt

schema.xml

scripts.conf

solrconfig.xml

spellings.txt

stopwords.txt

synonyms.txt

xslt

DRI.xsl

example.xsl

example_atom.xsl

example_rss.xsl

luke.xsl

...

statistics

conf

admin-extra.html

elevate.xml

protwords.txt

schema.xml

scripts.conf

solrconfig.xml

spellings.txt

stopwords.txt

synonyms.txt

xslt

example.xsl

example_atom.xsl

example_rss.xsl

luke.xsl

### Internationalization

Discovery has its own messages.xml file, located at dspace-xmlui/src/main/resources/aspects/Discovery/i18n/messages.xml. To add your own labels for new fields and facets in a Maven overlay, copy this file to dspace/modules/xmlui/src/main/resources/aspects/Discovery/i18n/messages.xml and modify this file. Alternatively, you may add them to the main messages.xml file. Same goes for translations - it's encouraged to submit a single messages_XX.xml file including messages from all the separate messages.xml files in DSpace.

Advanced search related keys (change "author" to desired field)

| | |
|---|---|
| Filter name | xmlui.ArtifactBrowser.SimpleSearch.filter.author |
| Facet heading | xmlui.ArtifactBrowser.AdvancedSearch.type_author |
| "Filter by" page heading | xmlui.Discovery.AbstractSearch.type_author |

Search & Faceting

The Search & Faceting system of DSpace-CRIS extends the basic Discover module of DSpace inheriting all its configuration and capacity and adding more. Here we will give just a quick overview of the basic concepts and will document the specific DSpace-CRIS extension. Please refer to the DSpace discover documentation for more details about the basic configuration.

The discovery module has been extended to be able to manage also Entities. New special entries can be used in the definition of the DiscoveryConfigurationService in the `[installDir]/config/spring/api/discovery.xml` file to allow specific configuration for entity type:

```
<bean id="org.dspace.discovery.configuration.
DiscoveryConfigurationService"
 class="org.dspace.discovery.configuration.
DiscoveryConfigurationService">
 <property name="map">
  <map>
   <entry key="default" value-ref="defaultConfiguration" />
   <entry key="publication" value-ref="publication"/>
   <entry key="person" value-ref="person"/>
   <entry key="project" value-ref="project"/>
   <entry key="organization" value-ref="organization"/>
    ....
  </map>
  ....
```

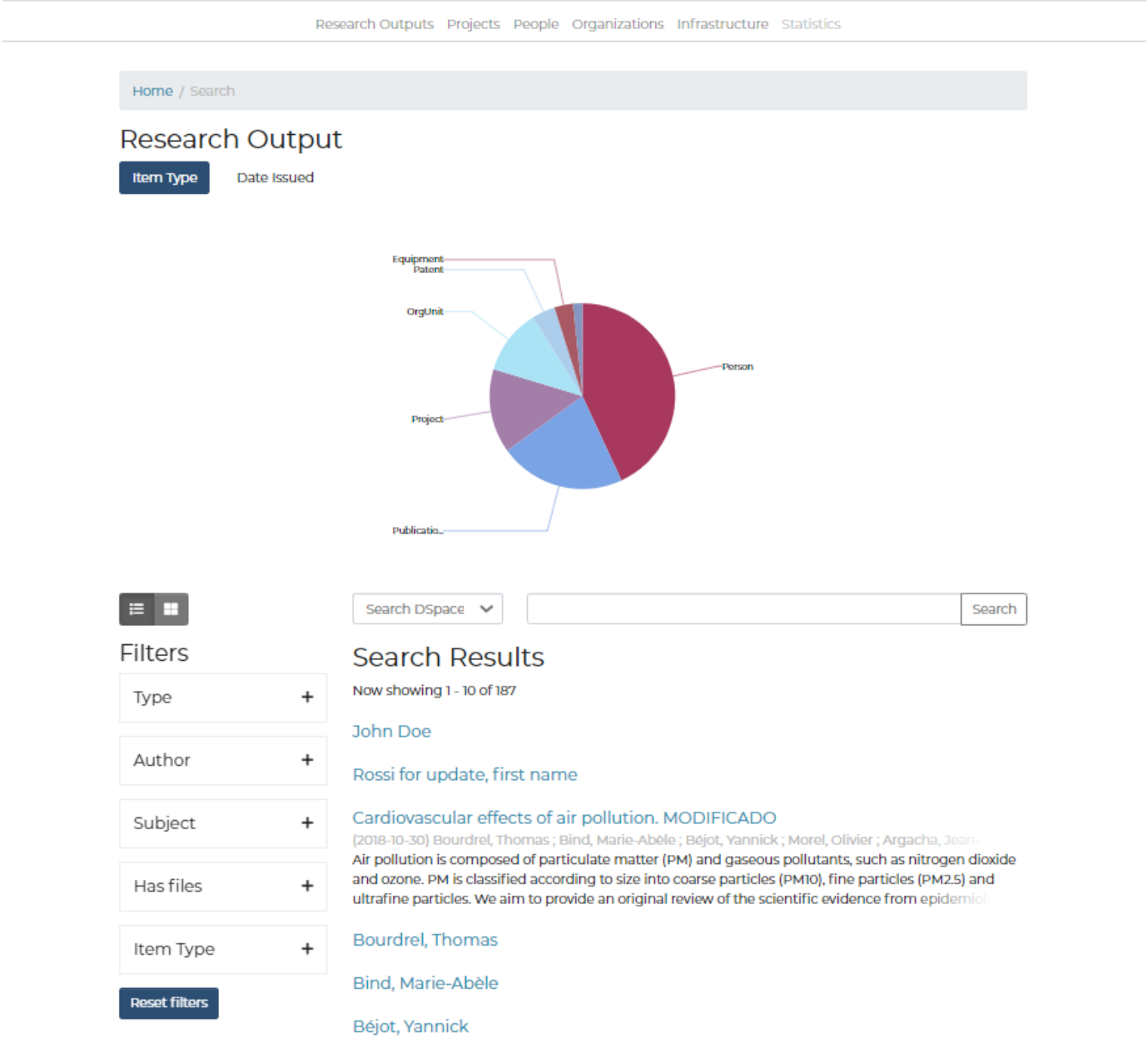
The map containing all the settings, the key is used to refer to the page/scope of the search, the "site", a community/collection handle or an entity type, the value-ref is a reference to a spring bean that actually define the DiscoveryConfiguration format. Below are some of the configurations present:

- **default** is the configuration key used if a not more specific one exist for the current search/indexing scope
- **publication** is the configuration key used searching/indexing a Publication
- **person** is the configuration key used searching/indexing a ResearcherPage
- **project** is the configuration key used searching/indexing a Project
- **organization** is the configuration key used searching/indexing an OrgUnit

The searching scope is defined by the UI implicitly when the search is performed from a “specific page” as a community or collection home page or explicitly when the user choose to restrict the search to a specific subset.

#### Graphical faceting

DSpace-CRIS supports the definition of facets that are visualized as charts. In the default layout they are positioned just over the search box and presented as a sequence of tabs so that the opened chart has always a full row for display.




The user can interact with the chart, it is possible

- to filter the result clicking on data point in the chart

- to scroll the chart when there are more data that the ones allowed in the default view and the left to right or right to left scroll is enabled in the configuration (only apply to bar charts)

To define a graphical facet it is necessary to configure a facet bean in the `discovery.xml` as an instance of the `GraphDiscoverSearchFilterFacet` class, below an example

```
<bean id="graphPublicationByType"
    class="org.dspace.discovery.configuration.
GraphDiscoverSearchFilterFacet">
        <property name="facetLimit" value="10"/>
        <property name="indexFieldName" value="graphitemtype" />
        <!-- the splitter needs to be a not empty string -->
        <property name="splitter" value="::" />
        <!-- <property name="maxLevels" value="3" /> -->
        <!-- <property name="onlyLastNodeRelevant" value="true" /> -->
        <property name="onlyLastNodeRelevant" value="true" />
        <!-- isDate cannot be true if the splitter is set -->
        <property name="isDate" value="false" />
        <!-- <property name="fillDateGaps" value="true" /> -->
        <!-- <property name="sortOrderSidebar" value="VALUE"/> -->
        <!-- <property name="inverseDirection" value="true" /> -->
        <property name="metadataFields">
            <list>
                <value>dc.type</value>
            </list>
        </property>
        <property name="graphType" value="pie" />
        <property name="exposeMissing" value="true" />
        <property name="exposeMore" value="true" />
        <property name="exposeTotalElements" value="true" />
</bean>
```

The bean definition has the following properties

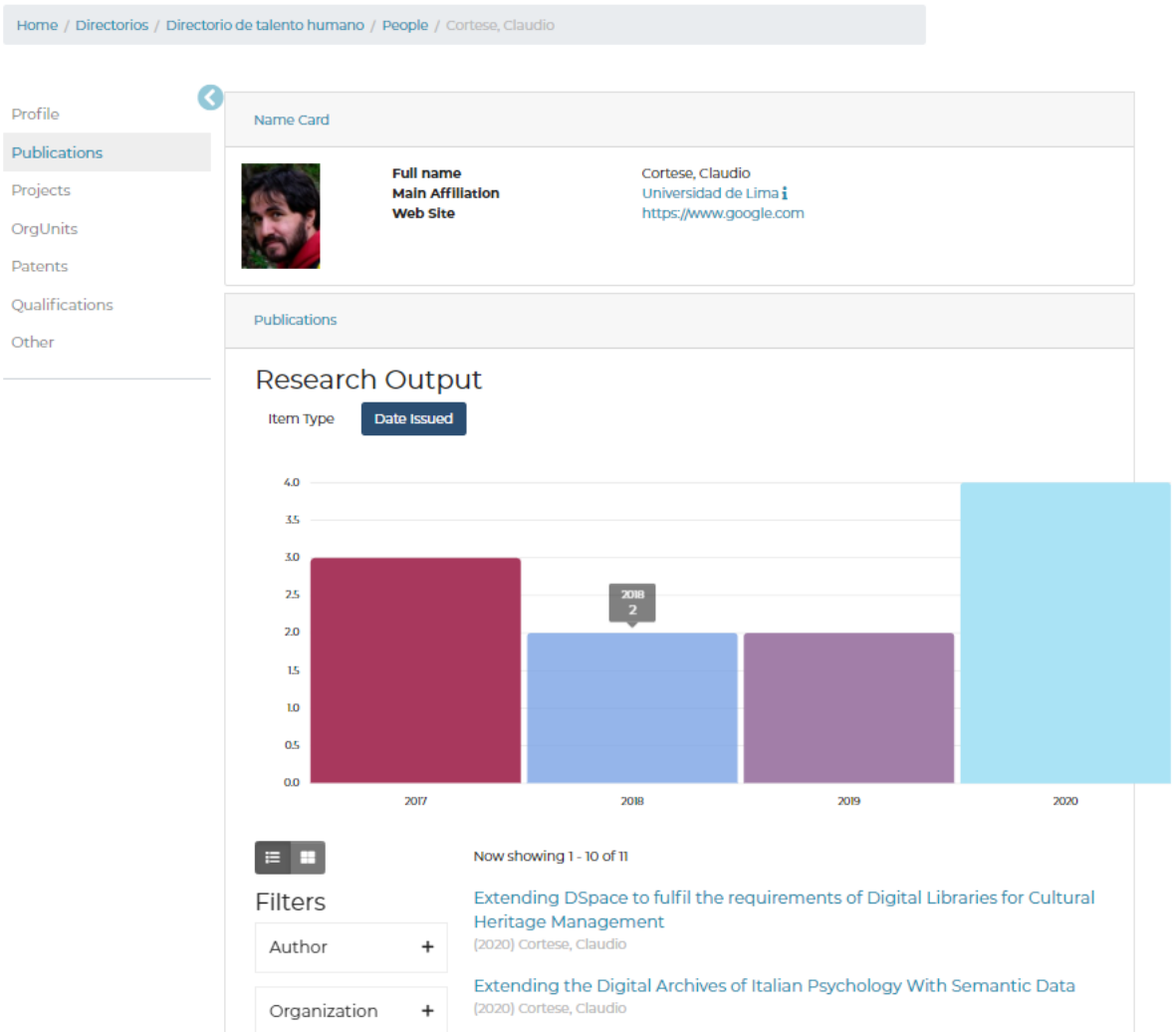
- `graphType`: it defines which type of chart will be drawn. The supported values are `bar`, and its specializations (`bar.left-to-right` and `bar.right-to-left`), `pie` or `line`
- `indexFieldName`: it must be an unique string that will be used to prefix all the fields in the SOLR documents needed by the facet
- `splitter`: it can be used to split the metadata values retaining only a substring. It works in combination with the `maxLevels` and `onlyLastNodeRelevant` properties. If the `onlyLastNodeRelevant` is true only the last valid fragment is used, otherwise the text up to the specified `maxLevels` is keep. With the default configuration (`maxLevels` unlimited, `onlyLastNodeRelevant` false) the value are stored as is. The `splitter` must be null if the `isDate` (see below) is set to true
- `isDate`: if true the metadata are parsed as a date and the year is extracted.
- `fillDateGaps`: can be only used when `isDate` is true. It forces to return all the years between the first and last years in the result so that there is no gap in the visualization (i.e. the gap is filled with the missing year at 0 count)
- `sortOrderSidebar`: it can be VALUE or COUNT (default). Values are sorted in ascending order by default, Count in descending order (the most frequent terms first)
- `inverseDirection`: it can be used to reverse the direction of the sorting
- `metadataFields`: it contains the list of metadata that are indexed in the facet
- `exposeMissing`: if enabled the system return the number of documents without any value for the facet
- `exposeMore`: if enabled the system return how many documents have values other than the one listed in the current facet page
- `exposeTotalElements:` if enabled the system return how many different values exist in the facet (with at least 1 document)

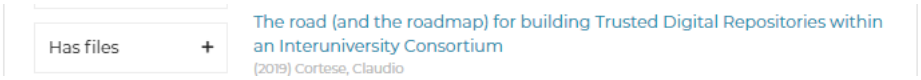
The facet bean must be referenced in one or more configurations that will determine in which scenario the graph is shown.

For example, in the default configuration of the inverse relation between research outputs and people we have

```
<bean id="relationAuthorResearchOutputsConfiguration" class="org.dspace.
discovery.configuration.DiscoveryRelatedItemConfiguration">
     <!--Which sidebar facets are to be displayed-->
     <property name="sidebarFacets">
         <list>
             <ref bean="graphPublicationByType" />
             <ref bean="graphPublicationByDate" />
             <ref bean="searchFilterIssued" />
             <ref bean="searchFilterAuthor" />
             <ref bean="searchFilterEditor" />
                 <ref bean="searchFilterOrgUnit" />
                 <ref bean="searchFilterSubject" />
                 <ref bean="searchFilterFunding" />
                 <ref bean="searchFilterContentInOriginalBundle"/>
                 <ref bean="searchFilterType" />
         </list>
     </property>
     ...
</bean>
```

that result in two chart facets to be included in the components






If the configuration is used by a search form, such as the general site search or one from a specific explore section the charts will be shown also, see for instance the screen at the start of this paragraph.

### Browse system

The browse system can be accessed via the explore section (see dedicated documentation) or the community and collection' home page (see at the bottom of this page for the specific configurations)

The browse indexes for DSpace-CRIS can be extensively configured. This section of the configuration allows you to take control of the indexes you wish to browse, and how you wish to present the results. The configuration is broken into several parts: defining the indexes, defining the fields upon which users can sort results, defining truncation for potentially long fields (e.g. authors), setting cross-links between different browse contexts (e.g. from an author's name to a complete list of their items), how many recent submissions to display, and configuration for item mapping browse.
There are two types of browse:

- **Full**, single level browse, which just list in a specific order the instances of an Entity class: Researchers, OrgUnits, etc.
- **Metadata**, two levels browse, that provide a first page listing the values of a configured metadata leading to a second page where the instances that have the clicked value for that metadata are listed

All configuration properties described in this section have to be set into configuration (.cfg) files, default properties are in `dspace.cfg` file.

#### Full, single level browse

The syntax to configure full browse (single level) is

```
webui.browse.index.<n> = <index-name>:<display-type>:<sort-name>[:DESC]
```

- **index-name** is used to refer to further configurations as the list of columns to show and generate the i18n keys for the navigation (menu links, page header, etc.)
- **display-type** can be anything except metadata and metadataAuthority that are reserved word for the two level browse. Using the browse name it is possible to define filter to apply to the general SOLR query, see “Apply filters to the browse indexes”
- **sort-name** is used to refer to the sorting configuration described below *DESC* if used make the descending order the default for that browse

For example

```
webui.browse.index.1 = dateissued:itemPublication:dateissued
```

The syntax to configure a sort option is

```
webui.itemlist.sort-option.<n> = <sort-name>:<metadata>:<value-type>
```

- **sort-name** is the name by which the sort option will be identified. This is the name by which it is referred in the "webui.browse.index" settings
- **metadata** is the field to be sorted on in the index
- **value-type** refers to the datatype of the field;  can be one of title, text, date or any other alias used to configure a Sort Plugin:
  - `date` the sort field will be treated as a date object
  - `text` the sort field will be treated as plain text.
  - `title` the sort field will be treated like a title, which will include a link to the item page

For example

```
webui.itemlist.sort-option.1 = title:dc.title:title
```

#### Metadata, two levels browse

It is possible to define browse over a metadata (i.e. authors of an item). In this way the system will produce a two levels browse, the first level will show in a paginated way all the values of the metadata (i.e., all the authors) clicking on a single value will show the list of items that match the selection. Applying this concept to the CRIS entities, you can for example build a two level browse showing all the departments of the researchers and for any department the list of researchers affiliated

```
webui.browse.index.<n> = <index-name>:<display-type>:<schema.element.
qualifier>:<text|date>
```

- **index-name** is used as reference in the column definition configuration to apply specific configuration for that browse
- **display-type** can have the following values
  - metadata is used to build a browse on any values used with or without authority
  - metadataAuthority limit the browse to only the value with an authority key
  - metadataXXXX where XXXX can be anything behave as metadata allowing separe definition of default filtering for the browse (see next section)
- **schema.element.qualifier** defines the field upon which the browse is build. It is possible to specifying multiple metadata fields in one index separating them with an ESCAPED comma (,).
- **text|date** specify if the values must be interpreted as String or Dates for sorting

For example

```
webui.browse.index.6 = webui.browse.index.6 = type:metadata:dc.type:
text
```

#### Apply filters to the browse indexes

It could be useful to restrict the set of objects for a specific browse applying additional SOLR filter query. To configure a filter for a specific browse you can define the following configuration property

```
browse.solr.bi_<display-type>.filter = <your-solr-filter-query>
```

- **display-type** is the value of the second part of the browse configuration. It is metatadata, metadataAuthority or metadata<Something> for two levels browse or something else for the configuration of a full browse index.

For example the following configuration will limit the browse to the items published after the 2000

```
browse.solr.bi_item.filter = dateissued:[2000 TO *]
```

When you are limiting a two level browse you need to configure, typically the same filter, also for the second level. In such case the browse index is used

```
browse.solr.bi_<n>_dis.filter = <your-solr-filter-query>
```

For example the following configuration will limit the browse to contains the authors names of only item published from the 2000 on and to list under such names only these items

```
webui.browse.index.2 = author:metadataauthor:dc.contributor.*,dc.
creator:text
browse.solr.bi_metadataauthor.filter = dateissued:[2000 TO *]
browse.solr.bi_2_dis.filter = dateissued:[2000 TO *]
```

### Add Communities' or Collections' Browse boxes

The buttons displayed in the Browse boxes of the communities and collections are determined by the following fields:

- **browse.community**: a list of index name related to the browse by to show for the communities
- **browse.collection**: a list of index name related to the browse by to show for the collections
- **browse.collection.<entity-type>**: a list of index name related to the browse by to show for the collections with the given entity-type

Once a collection of a specific type <entity-type> is selected, the browse.collection.<entity-type> property will be read to show the correct browse by button list; if a collection does not have an explicit entity-type or if the property for that entity type is not configured then the button configuration will derive from the browse.collection property.

For example

```
browse.collection = author
browse.collection = dateissued
browse.collection = type
browse.collection.OrgUnit = ouname
```

with the previous configuration:

- the communities page does not show browse by buttons
- the Publication collections page shows 3 browse by (author, dateissued and type)
- the OrgUnit collections page shows only a browse by name
- the collections without entity type shows 3 browse by (author, dateissued and type)

Following this example BrowseBy “author” is related to following index

`webui.browse.index.2 = author:metadata:dc.contributor.*\,dc.creator:text`

which represents a metadata index of text type, built using `dc.contributor.*` and `dc.creator` metadata, and it has an extra filter

`browse.solr.bi_2_dis.filter= entityType:Publication`

which restricts the index number 2 (bi_2 means index number 2) to browse only entity of Publication type. With those settings, “author” browse won’t index entities different than Publication, thus will allow browsing only between publications.

⚠ If a new property of type browse.collection.<entity-type> is added, it must be added to the properties that can be exposed through the configuration REST endpoint. To do this you need to add a property **rest.properties.exposed** to the **rest.cfg** with a value equal to that of the new property

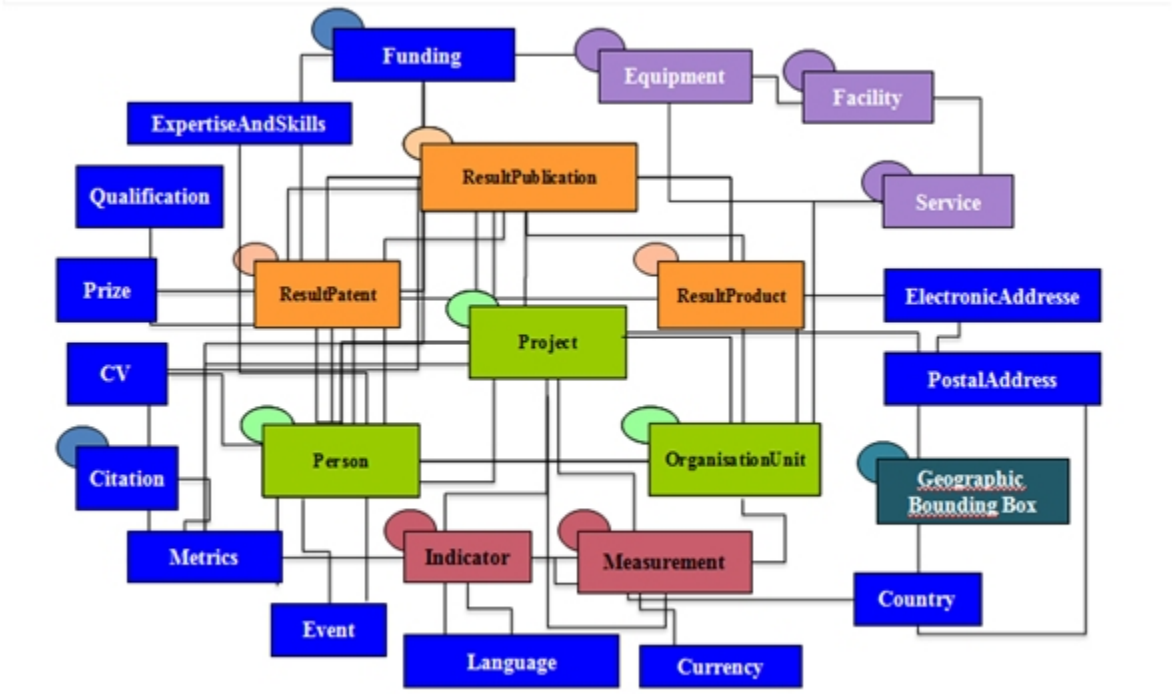
All BrowseBy needs to be reported in angular **environment** configuration file into the array “browseBy.types”. Here every entry has its *id*, which is the same reported in REST configuration file and its *type* which can be one of: `BrowseByType.Title`, `BrowseByType.Metadata` or `BrowseByType.Date`, representing the type of browsing to be proposed to the user: item title, date or metadata.

For all BrowseBy, on angular side, following 118n keys must be provided `browse.comcol.by.<browse by name>` which will hold the value to be displayed in browse page buttons and `browse.metadata.<browse by name>.breadcrumbs` holding the text to be displayed into page breadcrumbs when this browsing has been selected.

## DSpace Items and CRIS Entities

While in previous versions of DSpace CRIS, output-related entities were modeled through specific data structures, in DSpace CRIS 7 all the entities are modeled as DSpace items.

The distinction between result-items (publications, products, patents) and CRIS entities items is made through the use of the metadata `dspace.entity.type` that will hold the type of Entity that the item represent whether it's a result entity, a base entity, a 2nd level entity or an infrastructure entity according to the CERIF entity classification.




**Entity metadata**

Starting from DSpace-CRIS 7, collections are used to configure different entities; therefore, you can relate each collection with one of the entities defined within the repository. DSpace-CRIS 7 allows also to relate a single entity with more than one submission form; so the user has to choose also the submission type the collection will be associated to. Each entity can be, therefore related with multiple metadata structures, according to needs of the different institutions.

An Entity in DSpace-CRIS can be described using metadata from different schemas. Institutions are free to add more metadata and schemas to their installation to meet local needs. To keep easier the reuse of the configuration fragments between different installations, allowing a more easy maintenance and share of customizations, the following practice is usually followed:

- the **dublin-core** schema (dspace/config/registries/dcterms-types.xml) is preferred when the metadata is available to express a specific concept, also instead of a more specific metadata in other schema, such as the name in the Person schema.org, because the Dublin Core is the DSpace default schema and all the configurations out-of-box are built on it
- a **schema.org** for additional attributes specific of an entity that have been already standardized in Schema.org, for instance the https://schema.org/Person, https://schema.org/Organization or others. Such schema are named schema-<entity>-types.xml (i.e. dspace/config/registries/schema-person-types.xml)
- the OpenAIRE literature v4 schema for metadata described in the guidelines and not available in the above schemas
- the OpenAIRE CRIS schema for metadata described in the guidelines and not available in the above schemas
- a DSpace-CRIS entity specific schema for any further attributes of common usage.

Other than the descriptive metadata the platform also use some feature specific metadata from specific schema

- `eperson-types.xml` used for DSpace account and group management
- `relationship-types.xml` used for entity characterization
- `dspace-types.xml` used by DSpace standard features (i.e. configurable entities, process execution)
- `cris-types.xml` used for DSpace-CRIS additional features such as the object owner, the sourceId of imported records, etc.

Institutions are welcome to add their own schema to manage additional semantic or technical metadata. We recommend to create a separate schema for feature specific metadata based on the name of the project, such as `perucris-types.xml`. The out-of-box empty local-types.xml file can be used for descriptive metadata

#### How to manage relationships between items

The relationships between items are managed in DSpace-CRIS via the Authority Framework. In the dspace configuration it is possible to configure which metadata relates one dspace item to another dspace item or to external records. When the relationship is managed internally, i.e between dspace items the uuid of the linked item is stored as authority value in the metadata holding the relationship from the source item. For instance, given a publication Item A the metadata dc.contributor.author will have the value “Mario Rossi” and the authority of such metadata value will be the UUID of a Person Item B that represents the person “Mario Rossi”.

Relationships in DSpace-CRIS are always implicitly bidirectional with one preferred side that will be used to actually store the data.

The side holding the relationship is usually identified as the one that changes more frequently compared to the other or the one that usually is created later. For instance to store the “author” relationship between publication - person it is better to use metadata in the publication record so that when new publications are added to the system for an existing person only the new publication record is touched and there is no need to edit also the person record. The same operations will be performed to the “investigator” relationship between project - person (the information will be stored in the project metadata) or the “affiliation” relationship between person - organisation (the information will be stored in the person metadata).

`ItemAuthority` are configured using the `authority.cfg` file; a configuration example is shown below:

```
plugin.named.org.dspace.content.authority.ChoiceAuthority = \
 org.dspace.content.authority.ItemAuthority = AuthorAuthority,\
 org.dspace.content.authority.ItemAuthority = AuthorAuthority,\
 org.dspace.content.authority.ItemAuthority = DataSetAuthority,\
 org.dspace.content.authority.ItemAuthority = JournalAuthority,\
 org.dspace.content.authority.ItemAuthority = OrgUnitAuthority,\
 org.dspace.content.authority.ItemAuthority = ProjectAuthority,\
 org.dspace.content.authority.ItemAuthority = PublicationAuthority,\
 ...

cris.ItemAuthority.AuthorAuthority.relationshipType = Person
cris.ItemAuthority.DataSetAuthority.relationshipType = Product
cris.ItemAuthority.JournalAuthority.relationshipType = Journal
cris.ItemAuthority.OrgUnitAuthority.relationshipType = OrgUnit
cris.ItemAuthority.ProjectAuthority.relationshipType = Project
cris.ItemAuthority.PublicationAuthority.relationshipType = Publication

choices.plugin.dc.contributor.author = AuthorAuthority
choices.presentation.dc.contributor.author = lookup
authority.controlled.dc.contributor.author = true

...
```

From this configuration we can understand that the metadata `dc.contributor.author` is linked to an `AuthorAuthority` and to a `Person` type entity, while the metadata `dc.relation.project` is linked to `ProjectAuthority` and to a `Project` type entity.

When relations are used to drive specific features or business workflows such as correction requests, selected list of objects, etc. In this case, relations are driven by DSpace7 relations framework.

### Projects and Funding

As default data model, DSpace-CRIS adopts the information space representation recommended by the OpenAIRE guidelines for CRIS Manager https://openaire-guidelines-for-cris-managers.readthedocs.io/en/latest/cris_elements_openaire.html

These guidelines are based on the CERIF model, which formalizes two separate entities to represent Project and Funding information.

It is relevant to note that Project and Funding are often in a 1:1 relation. This leads to ambiguous reference where researchers, from a formal perspective, wrongly mention one as a replacement of the other.

Generally speaking, a Project represent the scientific aspects, e.g. investigation and development activities that will be conducted. A Funding represents the economical and contractual aspects.

There are scenarios where Funding exists without an explicit Project or where the “ideal” project connected to the Funding is not relevant for the purpose of the CRIS system. This is usually the case for awards, external collaboration contracts, scholarships.

There are also scenarios where a single Project receives multiple Funding, e.g. from national and international Funders and from the local Institution.

DSpace-CRIS extends the representation of the Funding to include direct reference to the involved Organisations and Persons, where the current OpenAIRE guidelines assume that such relations are tracked only at the Project level. This choice explicitly supports the scenario where a Funding exists without a Project. Moreover, the ORCID definition of a Funding explicitly requires such relations. As, when a Project is in place, such relations are usually borrowed from the Project, the DSpace-CRIS UI prefills them with the Project data as soon as a Project is linked to the Funding. The user is still free to modify such values to attribute the right contractual obligations to the different Organisations and Persons.

### Custom query filters for ItemAuthority query

Standard solr query that performs the lookup of items during ItemAuthority (`org.dspace.content.authority.ItemAuthority`) `getMatches()` method can be extended with custom query filters.

To facilitate this, standard abstract class `org.dspace.content.authority.CustomAuthorityFilter` is provided. To define custom filter, this class needs to be extended and instrumented as Spring Bean.

Extending classes must implement `appliesTo()` method, that checks if custom filter queries need to be applied, according to its own custom internal logic.

Two ready-to-be configured extensions are provided by classes `org.dspace.content.authority.EntityTypeAuthorityFilter` and `org.dspace.content.authority.AuthorityNameAuthorityFilte` that allow users to define some static queries which don't need to use particular runtime logic.

`org.dspace.content.authority.EntityTypeAuthorityFilter` provides a way to filter out the elements according to their Entity type. The supported entities need to be listed using the `supportedEntities` field, while the filtering parameters are specified with the `customQueries` field.

`org.dspace.content.authority.AuthorityNameAuthorityFilter` provides a way to filter out the elements according to their Authority name. The supported entities need to be listed using the `supportedEntities` field, while the filtering parameters are specified with the `customQueries` field.

`customQueries` , `supportedAuthorities` and `supportedEntities` fields can be set directly while configuring the Spring bean.

Following snippet is an example of filtering by Entity type:

```
<bean class="org.dspace.content.authority.EntityTypeAuthorityFilter">
      <property name="supportedEntities">
                        <util:list>
                  <value>Person</value>
                  <value>Project</value>
                  <value>Funding</value>
                  <value>OrgUnit</value>
                  <value>Publication</value>
                  <value>Patent</value>
                  <value>Equipment</value>
                      </util:list>
      </property>
      <constructor-arg name="customQueries">
                        <util:list>
                                          <value>dc.type:mytype</value>
                        </util:list>
      </constructor-arg>
</bean>
```

Following snippet is an example of filtering by Authority name:

```
<bean class="org.dspace.content.authority.AuthorityNameAuthorityFilter">
      <property name="supportedAuthorities">
                        <util:list>
                    <value>AuthorAuthority</value>
                    <value>EditorAuthority</value>
                    <value>DataSetAuthority</value>
                    <value>OrgUnitAuthority</value>
                    <value>ProjectAuthority</value>
                    <value>EquipmentAuthority</value>
                    <value>GroupAuthority</value>
                        </util:list>
      </property>
      <constructor-arg name="customQueries">
                        <util:list>
                                        <value>dc.type:mytype</value>
                        </util:list>
      </constructor-arg>
</bean>
```

In order to enable one of these custom filters, the corresponding Spring configuration xml needs to be put in any spring configuration file under the following path `config/spring/api`.

### Creation of linked CRIS entities

During the submission of an item it is usual to link this new item to others. This is for instance the case of a publication item that during the submission is linked with person items (authors, editors, etc.), project item, organisation item, journal item, etc

When the item to link is not yet present in the system, i.e. the person item for an author doesn't exist or the project item, etc. it is possible to configure the system to automatically create a new item for him using the data provided in the publication item under submission and/or enriched with extra information retrieved from external sources with a pluggable logic (i.e. an author identified in the ORCID Registry can get a person item filled with data from the ORCID Registry).

The linked items are created upon the acceptance of the submitted item into the archive. This means that no linked items are created when the submitted item is in the workspace or workflow.

The creation is performed by a DSpace consumer named **CrisConsumer** that listen for INSTALL and MODIFY events over items. Again the normal flow is related to the INSTALL event that is triggered when the submitted item is accepted in the repository, the MODIFY events are listed to process edits of the submitted item done after that the item has been archived.

#### Configuration

The CrisConsumer is configured in the **dspace.cfg**

```
event.consumer.crisconsumer.class = org.dspace.authority.CrisConsumer
event.consumer.crisconsumer.filters =
Item+Install|Modify|Modify_Metadata
```

Once that an item has been accepted in the repository the consumer will scan all his metadata looking for the ones associated with an ItemAuthority, i.e. used to link the item with other items.

For instance with a default configuration for a publication item the **dc.contributor.author** is used to link the publication with person items. In such case the CrisConsumer will create an item for the authors in the publication that doesn't have one yet or in some case, see below, will associate the author string to an existing person item. The entity type (**relationship.type**), Person, to assign to the item to be created is identified analyzing the configuration of the ItemAuthority instance associated with the dc.contributor.author metadata

```
cris.ItemAuthority.AuthorAuthority.relationshipType = Person
...
choices.plugin.dc.contributor.author = AuthorAuthority
...
```

#### Link the metadata value to an existing items

For each metadata value that must be processed the consumer will verify if an item for the same value was created before. This could be the case when several submissions are sitting in the workspace / workflow referencing the same “not yet existing” item (person, project, etc) and at a later point the items are progressively accepted. After the first ones, some of the references items would have been created and we expect the same to be reused by the other incoming items.

To do that the consumer will generate a sourceId from the metadata that will be used in the subsequent lookup. The use of a separate sourceId than the exact metadata value allows a more granular control about when an existing item should be used in the association.

Out of box many algorithms are provided to generate the sourceId

- if the authority starts with the prefix “**will be generated::**” or “**will be referenced::**” the sourceId will be the remaining part of the authority
- default, the sourceId will be the md5 hash of the metadata textual value uppercase. In such way the same, case insensitive, textual value will be always linked to the same target item
- uuid, the sourceId will be a random UUID. This strategy guarantee that no match with previous created items will never occur so a new item will be created for each value

To enable the uuid strategy it is needed to add the following property in the configuration

```
cris.import.submission.strategy.uuid.{field_key} = true
```

please note that **the property use the field key and not the dot representation** of the metadata so to enable it over the metadata dc.contributor.author the following property should be added to the dspace configuration (in the dspace.cfg or any overriding file such the local.cfg)

```
cris.import.submission.strategy.uuid.dc_contributor_author = true
```

If the CrisConsumer generates a new item this new item will be created at least with the following metadata

- **cris.sourceId** containing the generated sourceId
- **relationship.type** with the entity-type of the generated item
- **dc.title** with the textual value of the metadata that have generated the item

The lookup will also check the entity type of the item to link to identify valid match so that the same sourceId for a Person and a Project will lead to two different items.

For example, with the default hash strategy, the submission of a publication item with the metadata dc.contributor.author = “Mario Rossi” will lead to the calculation of a cris.sourceId as md5sum of “MARIO ROSSI” and a lookup for an item with the following metadata cris.sourceId = md5sum (MARIO ROSSI) AND relationship.type = “Person”

#### Consumer’s Logic

Once an item has been archived the CrisConsumer performs the following operations:

1. identifies the metadata that can be associated to entities by consulting the configuration of the Authorities
2. calculates the **cris.sourceId** by adopting one of the strategies described above
3. calculates the **relationship.type** using the Authority configuration linked to the specific metadata
4. starting from the calculated cris.sourceId and relationship.type searches for an already existing item using the org.dspace.authority.service.**ItemSearchService** (specific “ItemSearchService and ItemSearcherMapper“ for more details):
    - if such an item exists, its UUID is used to set the metadata authority, in order to link the already existing item to the metadata of the just archived one;
    - if such an item does not exist and the authority didn’t have the prefix “will be referenced::” then the consumer creates a new one and then uses its UUID to enhance the metadata authority of the newly stored item.
5. enriches/modifies the related item identified using a specific **AuthorityImportFiller**

#### New Item's Creation

If it is necessary to create a new item to associate with the metadata, the CrisConsumer configures it in the following way::

- the **submitter** of the original item that has just been archived is set as the item's submitter
- as **owning collection** is set the "closest" collection to that of the archived item that has, among its metadata, a relationship.type equal to that calculated for the item under creation. In order to search for such a collection, all the collections present in the community of the archived item are examined: if among them there is a collection with the right relationship.type, it is used, otherwise, in a recursive way, the procedure goes up to the higher community and within it, even among the sub-communities, a collection is searched for, that has the value of the metadata relationship.type searched for. The search ends when there are no longer any higher level communities to examine, For example, assuming the following communities/collections structure
    - Community A
        - Collection A publication
        - Collection A person
    - Community B
        - Collection B publication
        - Collection B person

  a submission in Collection A publication that gives rise to a new "person" item should create it in Collection A person while a submission in Collection B publication should use Collection B person..
  If with the strategy described above the consumer has not been able to identify a proper collection, the creation of the CRIS item is not performed and a warning message is shown in the console. However, this does not preclude the creation of other items to be associated with other possible metadata.
- adds the **cris.sourceId** metadata to the item
- adds the **relationship.type** metadata to the item (if the item hasn't already inherited it from the collection identified with the algorithm described above)

Once the item is configured the consumer can decide whether to install it (default) or to start a workflow associated with it. The choice is made according to the **cris.import.submission.enabled.entity.{field}** configuration, or, if not present, according to the generic **cris.import.submission.enabled.entity** property.

#### AuthorityImportFiller

The system provides a mechanism to dynamically enrich the items created using additional logic. This mechanism is called the "ImportFiller framework" and is configured through a spring configuration present by default in the cris-plugin.xml file. A possible configuration is shown below.

```
<util:constant id="CrisConsumer-SOURCE_INTERNAL"
      static-field="org.dspace.authority.CrisConsumer.SOURCE_INTERNAL"
/>

<bean id="org.dspace.authority.filler.AuthorityImportFillerHolder"
      class="org.dspace.authority.filler.AuthorityImportFillerHolder">
        <property name="fillers">
                <map>
                        <entry key-ref="orcid">
                                <bean class="org.dspace.authority.
filler.OrcIdImportFiller"
                                        parent="
fullItemMetadataConfiguration" />
                        </entry>
                        <entry key-ref="CrisConsumer-SOURCE_INTERNAL">
                                <bean class="org.dspace.authority.
filler.ItemMetadataImportFiller"
                                        parent="
fullItemMetadataConfiguration" />
                        </entry>
                </map>
        </property>
</bean>
```

The **AuthorityImportFillerHolder** class collects all AuthorityImportFiller configured in the system in a map..

This map associates the possible implementations of the Filler to the value of an authority key, whose value depends on the metadata related to the CRIS item:

- If the metadata that caused the creation of the new item had no authority, the authority key is the **SOURCE_INTERNAL** value and the **ItemMetadataImportFiller** filler is used for enrichment.
- If the starting metadata has an authority that starts with "will be generated::orcid::" then the selected filler will be OrcIdImportFiller (to be implemented)

#### ItemMetadataImportFiller

The ItemMetadataImportFiller is configured as Spring's bean and represents the fillers used if the metadata associated with the item has no particular authority. This filler allows you to enrich the CRIS item by copying metadata present in the archived/modified item; the filler also allows you to update CRIS items previously created.

Below is a possible configuration example:

```
<bean class="org.dspace.authority.filler.ItemMetadataImportFiller"
      id="fullItemMetadataConfiguration" abstract="true">
        <property name="itemService" ref="org.dspace.content.
ItemServiceImpl" />
        <property name="allowsUpdateByDefault" value = "true|false" />
        <property name="configurations">
                <map>
                        <entry key="dc.contributor.author">
                                ....
                        </entry>
                        <entry key="dc.contributor.editor">
                                ....
                        </entry>
                </map>
        </property>
</bean>
```

where for each entry on the **configurations** map there is such a configuration:

```
<bean class="org.dspace.authority.filler.MetadataConfiguration">
        <property name="updateEnabled" value="true" />
        <property name="mapping">
                <map>
                        <entry key="oairecerif.editor.affiliation">
                                <bean class="MetadataConfiguration.
MappingDetails">
                                        <property name="visibility"
value="1|0" />
                                        <property name="useAll" value="
true|false" />
                                        <property name="appendMode"
value="true|false" />
                                </bean>
                        </entry>
                </map>
```

```
        </property>
</bean>
```

The **configurations** map shows the possible metadata that can generate the CRIS item, to be enriched with additional metadata. Each of these configured metadata refers to a **MetadataConfiguration** that has the following properties:

- **updateEnabled**: in the presence of a CRIS item that has been found in the database through the cris.sourceId search, this property indicates if the filler can act on it or if the update is not enabled. If this property is not set as default, the value of the **allowsUpdateByDefault** property of the **ItemMetadataImportFiller** is considered (which if not set, has a false value as default)
- **mapping:** iindicates the metadata to be copied from the item stored, in the CRIS item

Il mapping che consente di riportare i metadati dall'item archiviato è rappresentato da una mappa in cui la chiave rappresenta il metadato da copiare e il valore la configurazione con cui effettuare l'operazione:

The mapping that allows you to copy the metadata from the archived item is represented by a map in which the key represents the metadata to be copied and the value represents the configuration by which perform the operation:

- **visibility**: allows to specify the visibility of the metadata to be added to the CRIS item; the eligible values are private (0) or public (1) (to be implemented)
- **appendMode:** indicates whether the new metadata should be added to the set of metadata of the same type eventually already present in the CRIS item, or they should overwrite them (default = false)
- **useAll:** indicates whether all the metadata of the type referred to should be reported in the item to be enriched (true) , or only the one which has the same position as the metadata that gave raise to the CRIS item should be reported ((false, default)

With the reported configuration:

- if **useAll = false:** to the CRIS item related to the first dc.contributor.author only the first metadata oairecerif.editor.affiliation of the archived item would be added and so on.
- if **useAll = true:** to each CRIS metadata item dc.contributor.author would be added all metadata oairecerif.editor.affiliation of the archived item

#### ItemSearchService and ItemSearcherMapper

The cris-plugin.xml contains the definition of a bean of type org.dspace.authority.service.**ItemSearchService** which is used to search an item with a specific strategy. The methods of this interface allow to search for an item for a specific parameter and for relationship type.

In particular, the search done by the org.dspace.authority.service.**ItemSearchServiceImpl** implementation consists of the following steps, executed in the order indicated until an item is found:

1. if the given searchParam is a valid uuid the item is searched by uuid; if the search is also made for relatioship.type, any item found is discarded if the type does not match.
2. a search by cris.sourceId and relatioship.type is done using the given searchParam and relationship type
3. if the search param has the form <identifier>::<value> a search is made using the bean org.dspace.authority.service.**ItemSearcherMapper**

This ItemSearcherMapper handle a Map of <String, ItemSearcher> in which the keys represent the various strategy identifier that can be used and the value corresponds to a particular implementation of the org.dspace.authority.service.**ItemSearcher** interface which, through the searchBy (Context context, String searchParam) method, allows to search the item with a specific strategy. The current provided implementations of the ItemSearcher interface are:

- org.dspace.authority.service.impl.**ItemSearcherById**: search the item by uuid
- org.dspace.authority.service.impl.**ItemSearcherByMetadata**: search the item by the metadata provided during the bean instantiation

The ItemSearcherMapper can be configured with a default ItemSearcher.

An example configuration is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<bean class=org.dspace.authority.service.ItemSearcherMapper" id="
itemSearcherMapper">
   <constructor-arg index="0">
      <map>
         <entry key="UUID">
            <bean class="org.dspace.authority.service.impl.
ItemSearcherById" />
         </entry>
```

```
         <entry key="ORCID">
            <bean class="org.dspace.authority.service.impl.
ItemSearcherByMetadata">
               <constructor-arg ref="org.dspace.discovery.
SearchService" />
               <constructor-arg value="person.identifier.orcid" />
            </bean>
         </entry>
         <entry key="RID">
            <bean class="org.dspace.authority.service.impl.
ItemSearcherByMetadata">
               <constructor-arg ref="org.dspace.discovery.
SearchService" />
               <constructor-arg value="person.identifier.rid" />
            </bean>
         </entry>
         <entry key="ISNI">
            <bean class="org.dspace.authority.service.impl.
ItemSearcherByMetadata">
               <constructor-arg ref="org.dspace.discovery.
SearchService" />
               <constructor-arg value="person.identifier.isni" />
            </bean>
         </entry>
         <entry key="DOI">
            <bean class="org.dspace.authority.service.impl.
ItemSearcherByMetadata">
               <constructor-arg ref="org.dspace.discovery.
SearchService" />
               <constructor-arg value="dc.identifier.doi" />
            </bean>
         </entry>
      </map>
   </constructor-arg>
   <constructor-arg index="1">
      <bean class="org.dspace.authority.service.impl.ItemSearcherById"
/>
   </constructor-arg>
</bean>
```

#### Skip metadata with empty authority

The **cris-consumer.skip-empty-authority** property allows to configure or not the CrisConsumer to skip the metadata that have an empty authority: in this way, therefore, the creation of related items can only be carried out if the authority has the prefixes **will be generated::** or **will be referenced::** .

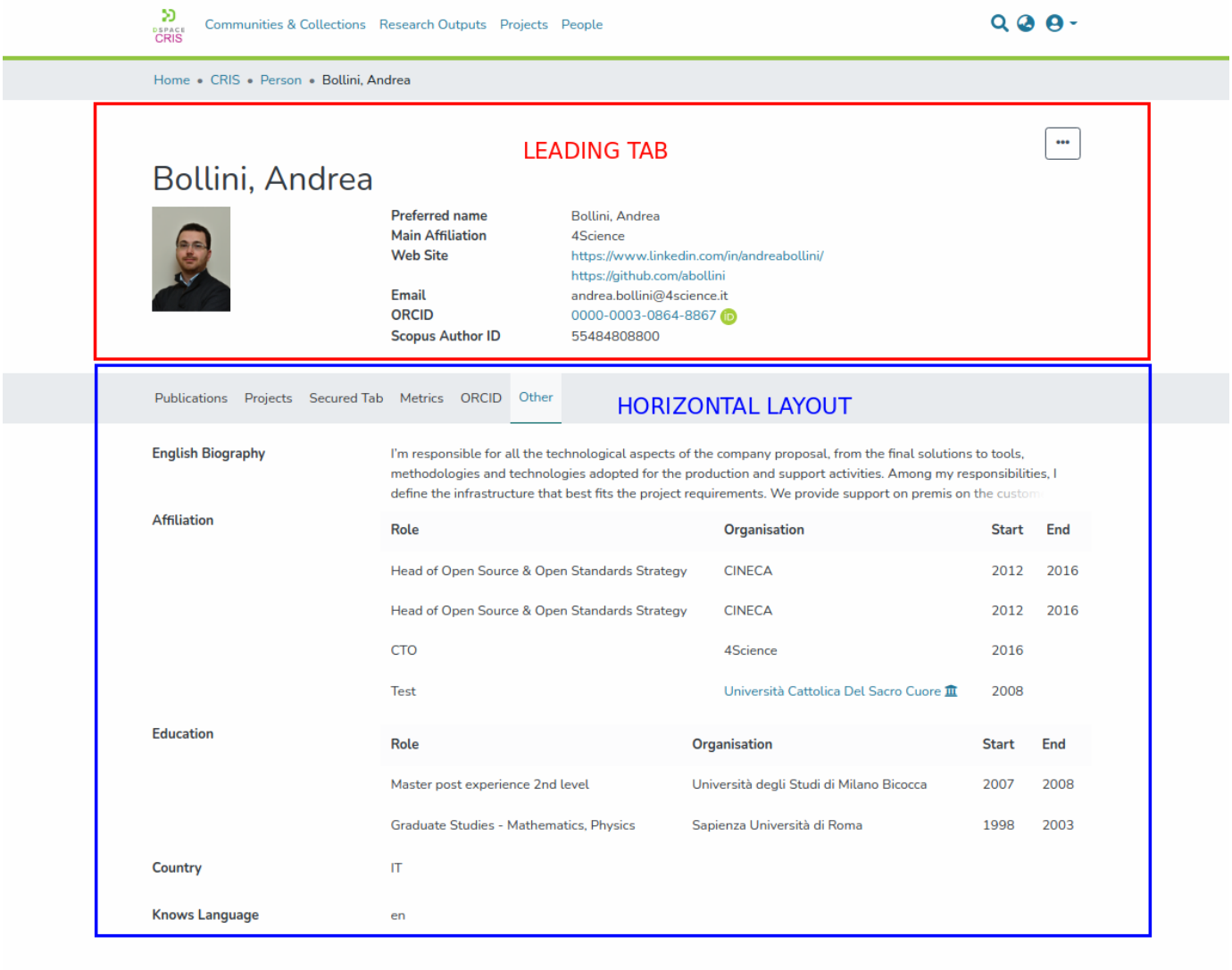
### Item details: layout & security

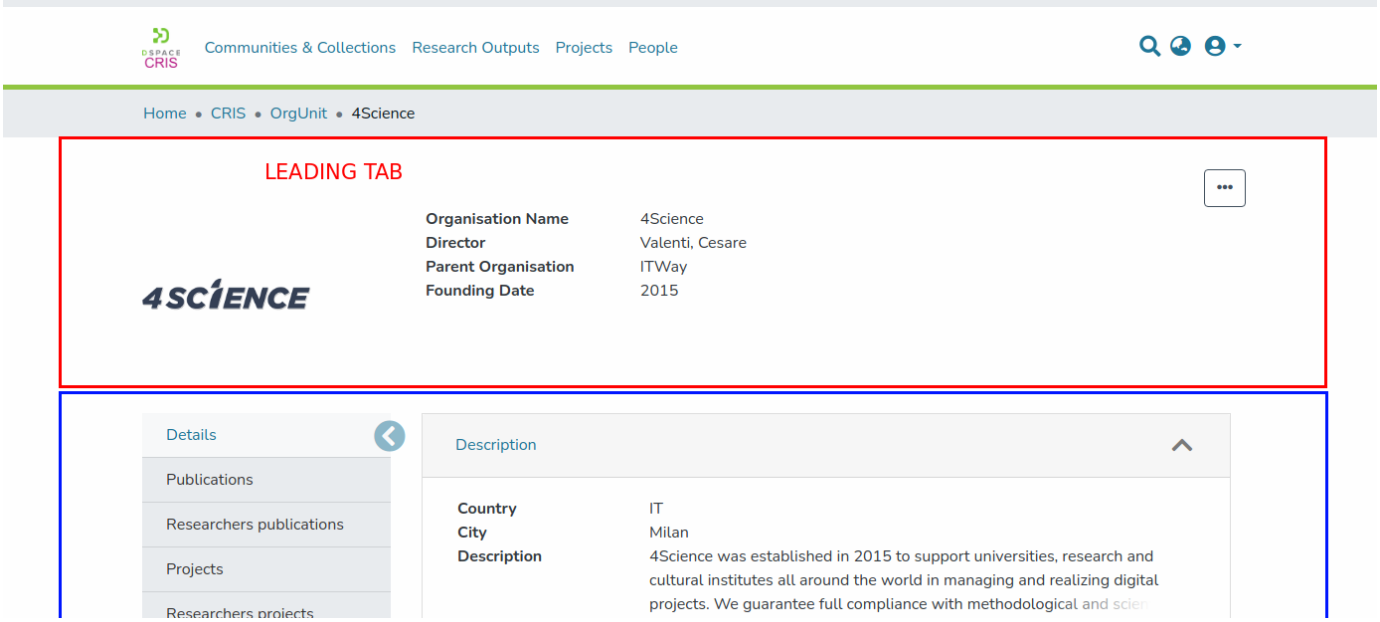
DSpace-CRIS allows to manage the visualization and access to the item data in a more fine-grain way than the default DSpace.

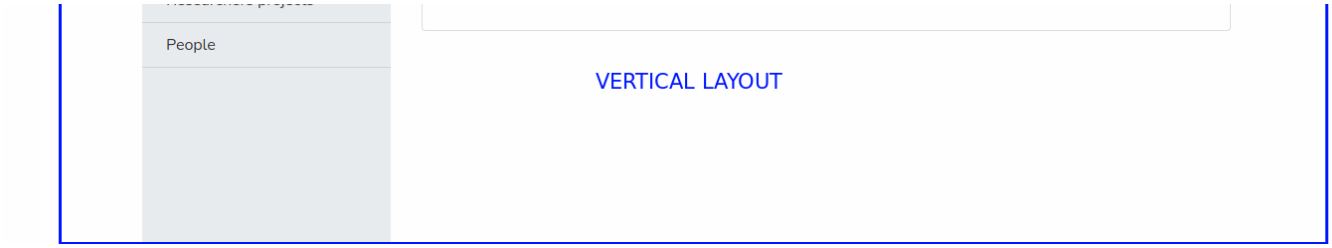
Continuing in the wake of the previous DSpace-CRIS versions the item information is organized over two levels, tabs and boxes. Each tab can contain one or more boxes organized in a sort of grid composed by rows and cells. The same box can be eventually shared between multiple tabs with a specific order in each tab. A box represents the minimal unit of information about an item that can be visualized and protected.

## Vertical and horizontal Layout

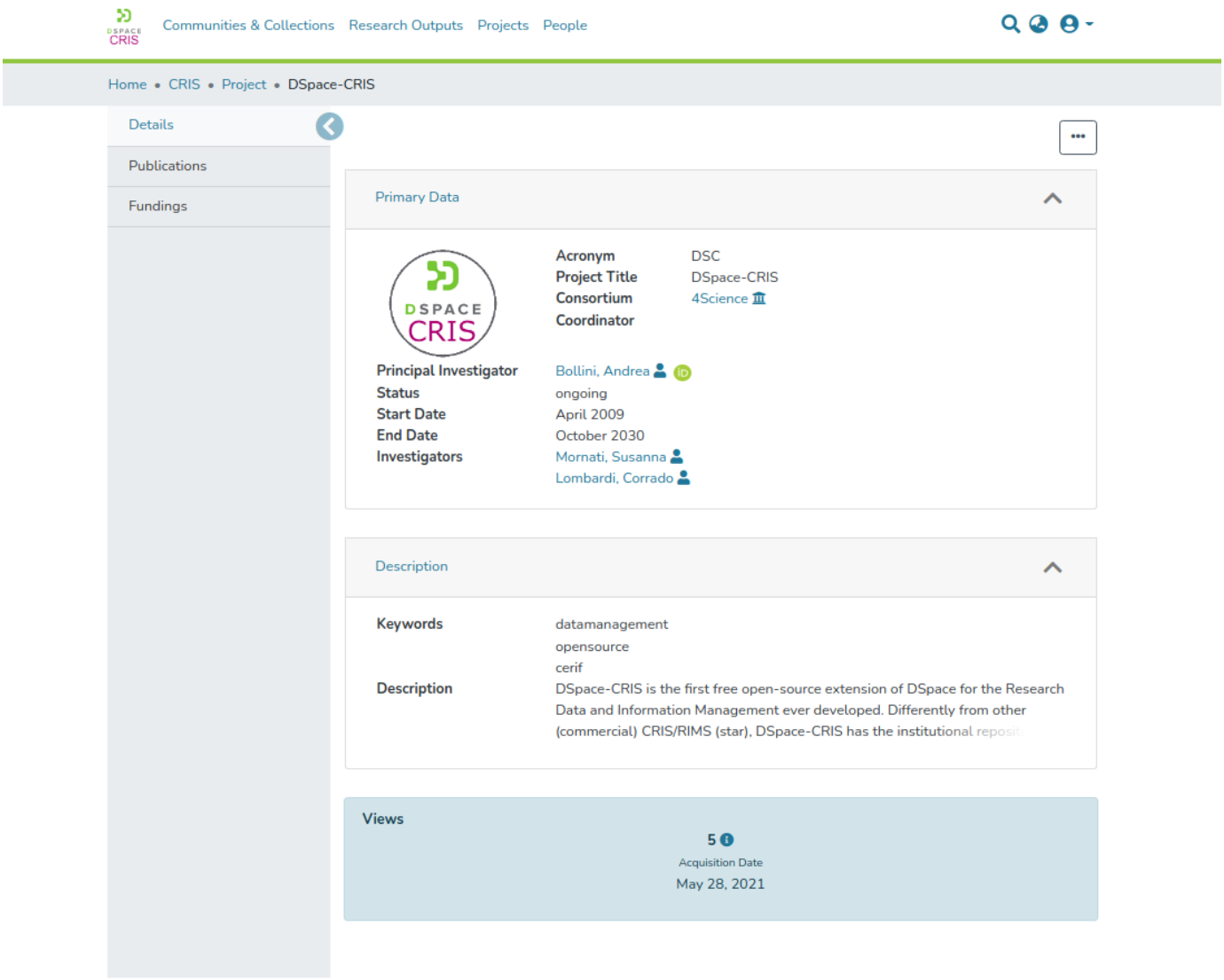
Unlike previous versions, however, it is now possible to have a main **leading tab** always visible and to organize the remaining tabs with a layout having two different orientations : **vertical** and **horizontal.** In the horizontal layout the tabs are arranged inside a top navbar while in the vertical one are arranged within a lateral sidebar



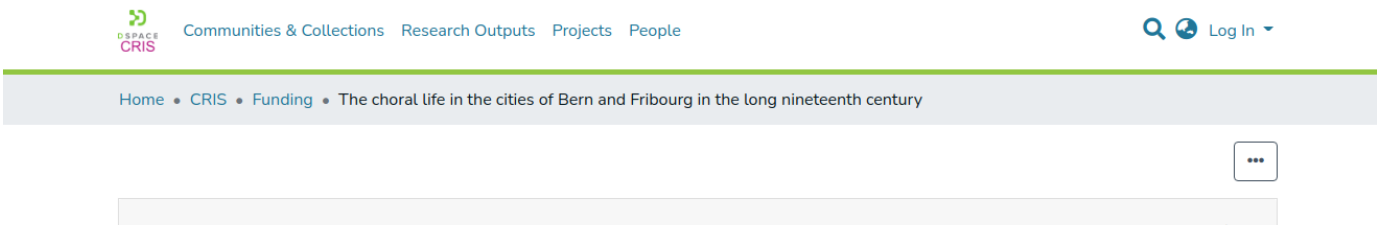


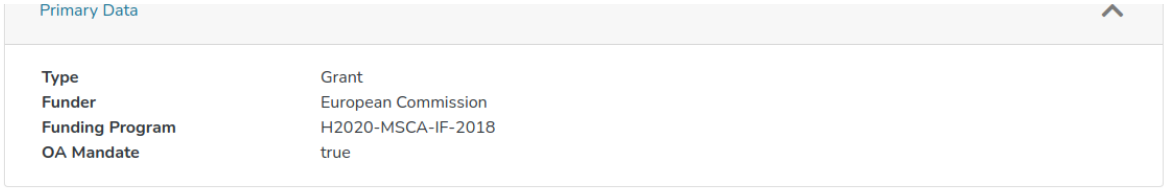

if the leading tab is not set only the chosen layout is displayed

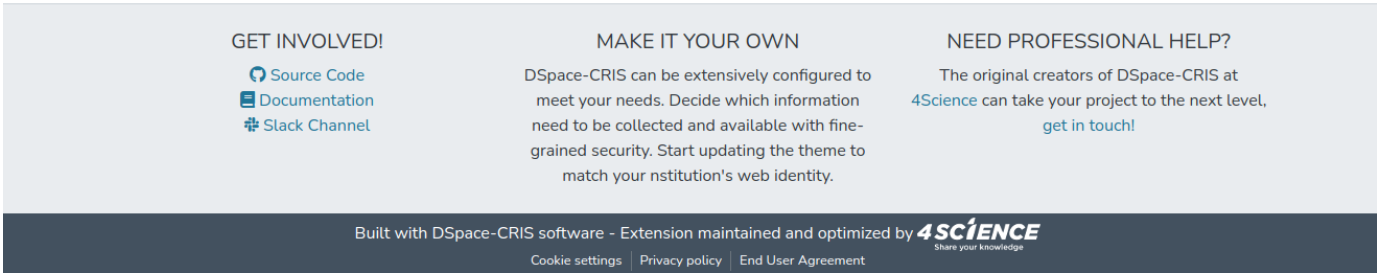



Only tabs and boxes that contain data visible to the current user are listed. If a tab only contains boxes without visible data or that are flagged as **minor** such tab is not listed at all.

In both `vertical` and `horizontal` layout when only one tab is available sidebar and navbar are not displayed.






It's possible to use one of the two layout (**vertical** or **horizontal**) differently depending on the type of entity. To select the layout the dsapce angular application provide an `itemPage` settings under the `cris-laoyut` property in the environment file (`src/environments/environment.common.ts`), e.g. :

```
itemPage: {
  Person: {
      orientation: 'horizontal'
  },
  default: {
      orientation: 'vertical'
  },
},
```

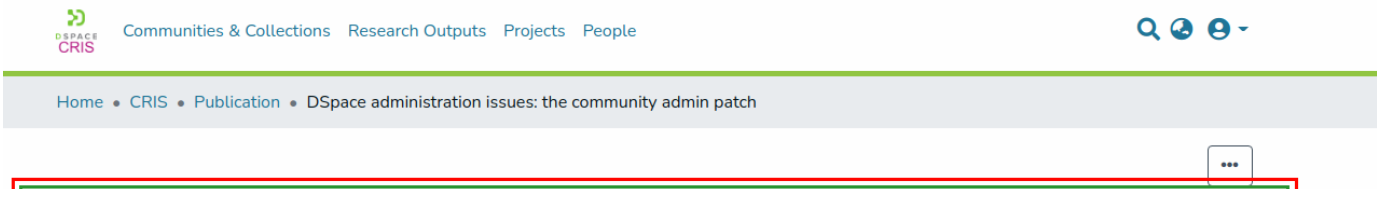
the previous configures an `horizontal` layout for the entity type `Person` while use the `vertical` one by default for all other entities type. It's possible to specify no entity type, in this case the default value is used for all entities.

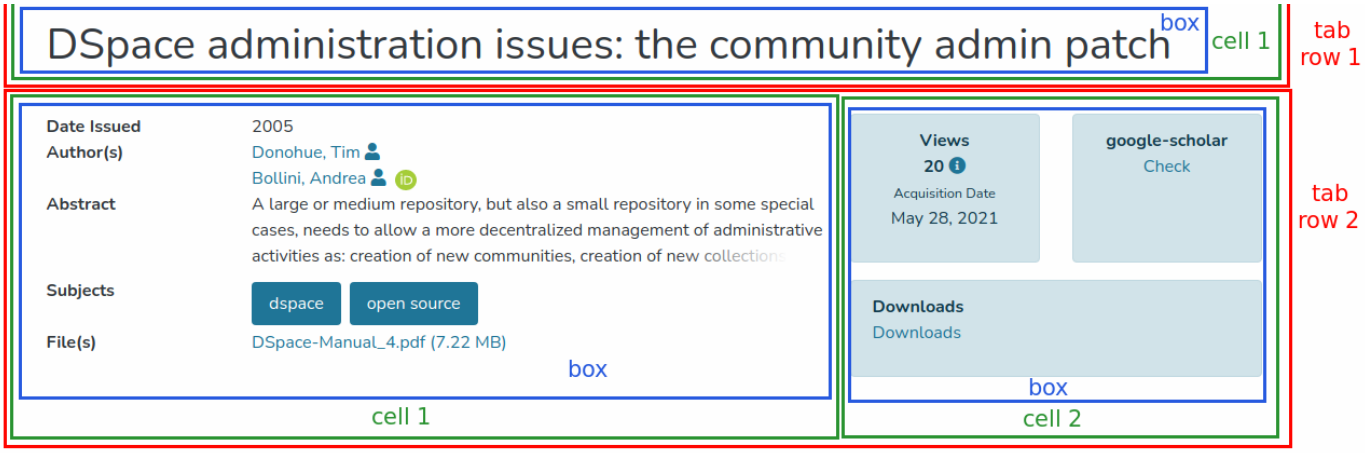
## Tab grid system

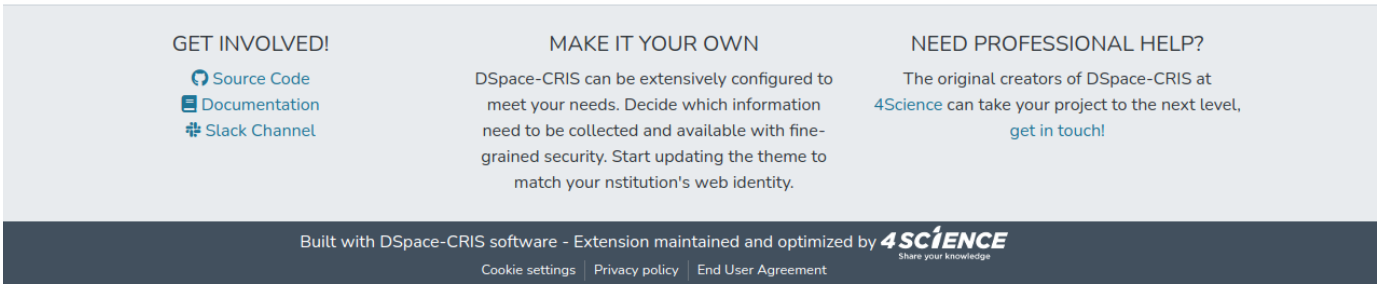
In order to have a more flexible way to arrange boxes within a tab a new grid system is used.  The grid system uses a series of rows and cells to layout and align boxes, following these rules :

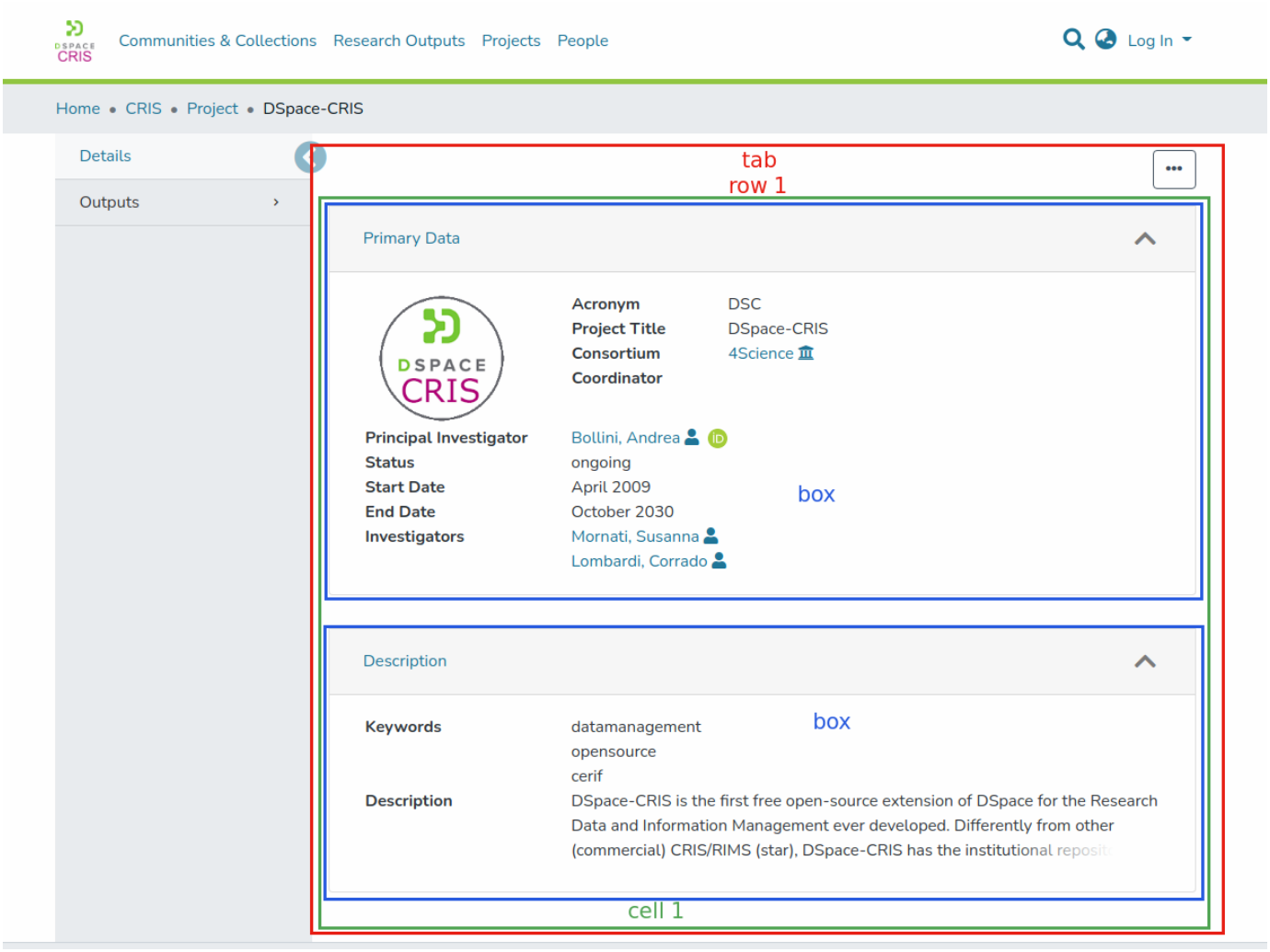
- Every row can contains one or more cells
- Every cell can contain one or more boxes.
- All cells inside the same row are displayed one next to the other
- All boxes inside the same cell are displayed one below the other

Here some practical examples :

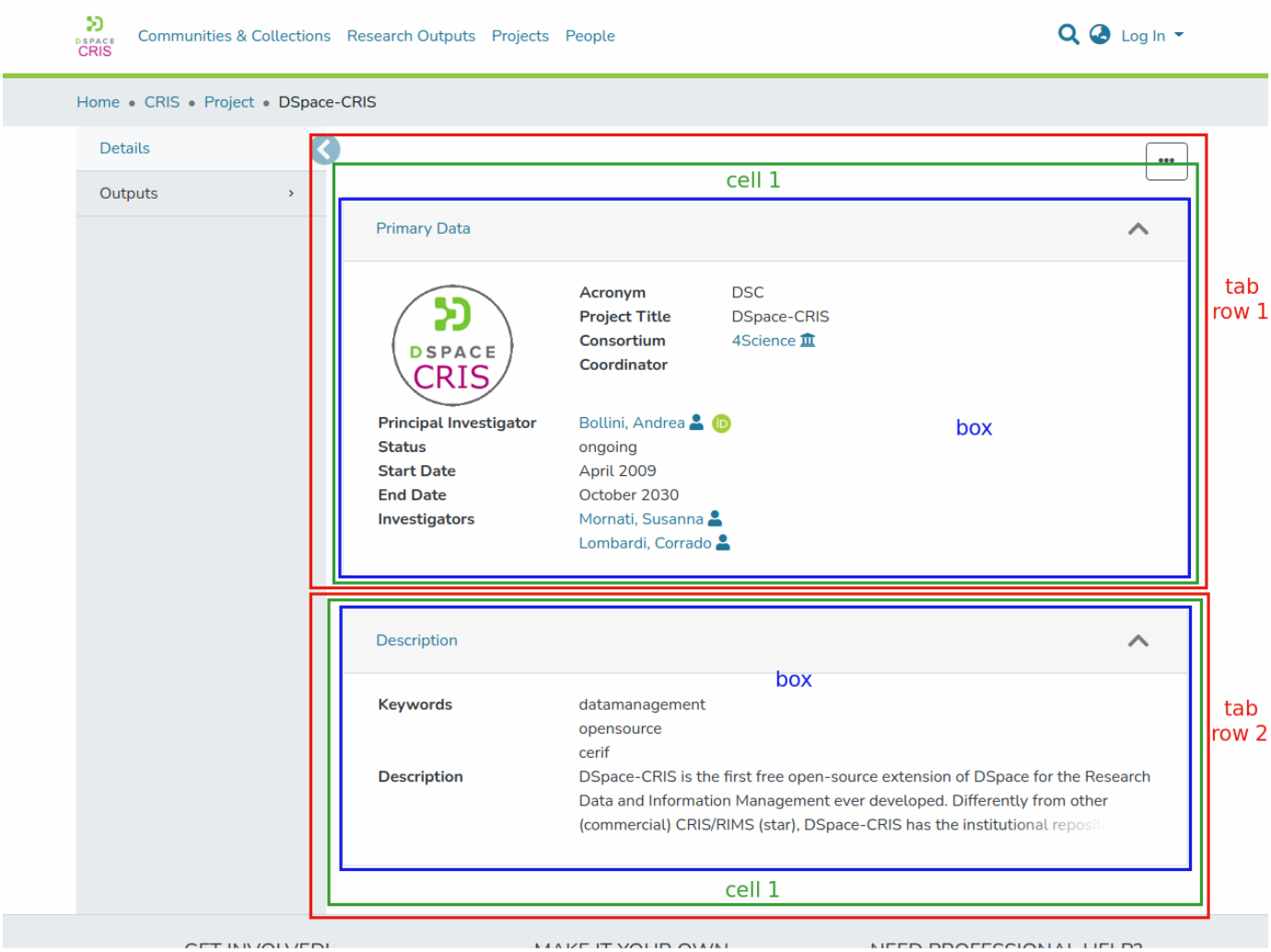




the same result can be achieved in a different way :




## Tab and Box models

The tabs are represented by the `org.dspace.layout.CrisLayoutTab` java class and exposed in the REST layer via the `org.dspace.app.rest.model.CrisLayoutTabRest`. The boxes are represented by the `org.dspace.layout.CrisLayoutBox` java class and exposed in the REST layer via the `org.dspace.app.rest.model.CrisLayoutBoxRest` and `org.dspace.app.rest.model.CrisLayoutBoxConfigurationRest`. The later one in particular is the extension point to plug the different types of boxes in the platform.

The REST contract for tabs and boxes can be found here:

https://github.com/4Science/Rest7Contract/blob/dspace-cris-7/tabs.md

Tabs and Boxes are bound to a specific Entity Type and share the following attributes :

- **shortname**. It is an alias of the Id used to refer to the tab / box from configuration files without the need to hardcode the database generated Id
- **label**. It is the label or the i18n key to use to present the section to the user. In case you want to use the header as a translation key the complete i18n key used by the system has the prefix `layout.tab.header.` for the tab and `layout.box.header.` for the box
- **security**. It can have one of the following values
  - 0 public. Everyone can access the data contained in the tab (if the security is not overridden by the box security) or box
  - 1 administrator. Only system administrator can access the data
  - 2 owner only. Only the owner of the item can access the data. Please note that the concept of item owner is specific of DSpace-CRIS and it is different from the submitter. The item owner is defined by the cris.owner metadata
  - 3 owner and administrator. Only the owner of the item and the system administrator can access the data
  - 4 custom policy. The list of people and groups that can access the data are defined in other metadata of the item itself. The metadata to use are defined in the **securityMetadata** attribute that contains a list of reference to metadata fields that are expected to be configured with the `EPersonAuthority` or `GroupAuthority`

Other than the common attributes above the tabs have also these extra attributes :

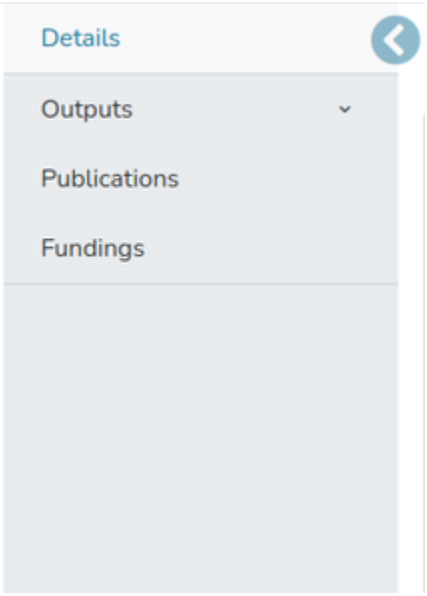
- **leading.** It can be true or false. If true the tab is shown on the top of the item's page and remains there even if the user browse the other tabs
- **priority** attribute that is used to sort them in ascending order.
- **rows.** It contains the configuration of the grid used to display boxes beloging to the current tab. Each row is composed by **cells** that contains the list of boxes. Boxes have a specific order inside each cell that include them.

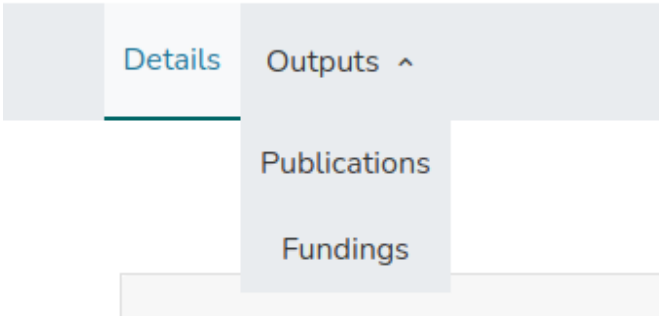
Other than the common attributes above the boxes have also these extra attributes :

- **container.** It can be true or false. If true the box is show as a collapsible panel otherwise it has no container and is always visible
- **collapsed**. When true and box container property is set to true the box panle start collapsed and the user need to open it to see the actual data
- **minor**. It is true when the box should be not used to determine if a tab actually has content or not
- **style.** It is added to the CSS classes of the generated html element that contains the tab or box to allow further customization via CSS
- **configuration**. It lcontains additional information used to render the data in the appropriate way

*Nested tabs*

It's possible to have two or more tabs grouped by the same top menu entry. In this case the tab entries are displayed with a dropdown in the sidebar and navbar :




to configure the nested two or more tabs the **shortname** property and **label** may contain the parent and child tabs concatenated by `::`.

Following the previous example where we have two tabs `publications` and `fundigs` grouped by a the top level tab `outputs` , the configuration is like :

| shortname | label |
|---|---|
| outputs::publications | Outputs::Publications |
| outputs::fundings | Outputs::Fundings |

When relating boxes to nested tab the shortname of the tab to use must include also the top level ( e.g. `outputs::publications`)

## Box types

Different types of boxes exist, the rest contract https://github.com/4Science/Rest7Contract/blob/dspace-cris-7/boxes-types.md details the different endpoints used by each type to expose the configuration details.

*Metadata Box*

The most simple and used is named **Metadata Box**. A metadata box is a collection of item metadata fields and selection criteria over the item bitstreams organized in rows each of which can contain one or more fields. Three types of fields exist (metadata and bitstream), their additional configuration options are exposed in an attribute with the same name than the `fieldType`

- **METADATA**. The field holds the values stored in an item metadata identified with the `<schema>.<element>[.<qualifier>]` syntax that is exposed in the `metadata` attribute

- **METADATAGROUP**. The field holds the values stored in a group of nested metadata identified with the `<schema>.<element>[.<qualifier>]` syntax that is exposed in the `metadata` attribute
- **BITSTREAM**. The field holds the bitstreams in a specific item bundle optionally matching a specific value for a metadata. The `bitstream` attribute is an object containing the
    - `bundle`, the name of the bundle
    - `metadataField` and `metadataValue`, optional, the value of a specific bitstream metadata that will be used to filter which bitstreams are included in the field

Regardless to the `fieldType` each field has the following attributes

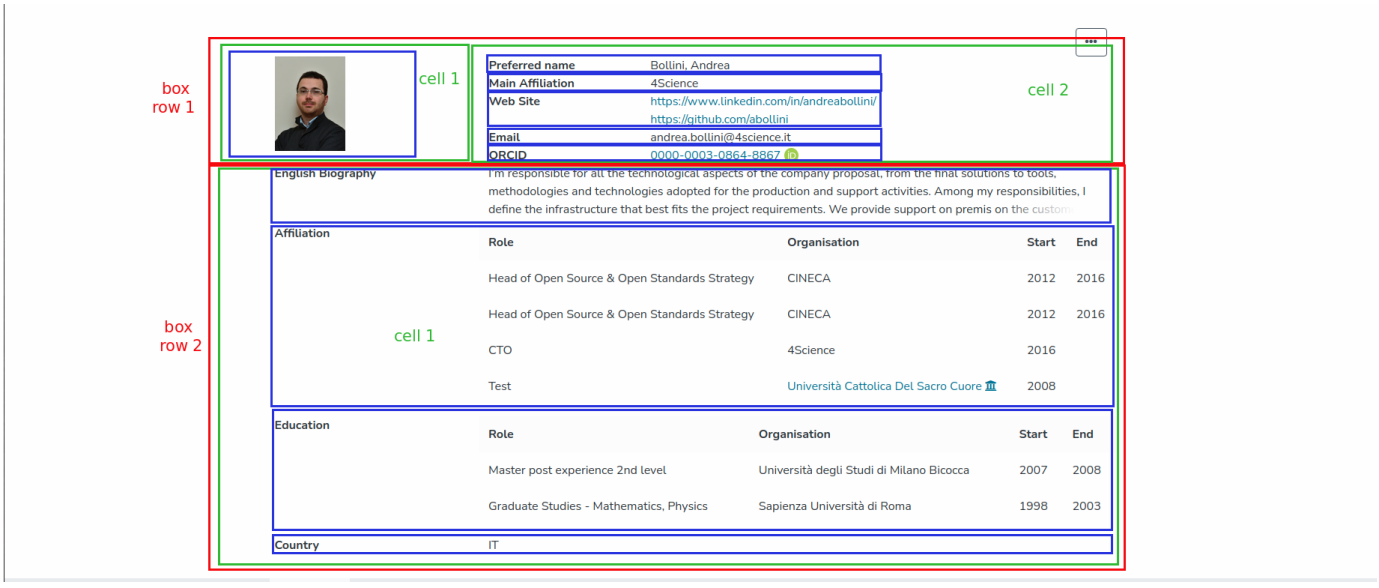
- **label**. the textual label or i18n key to use as label for the field. In case you want to use the header as a translation key the complete i18n key used by the system has the prefix `layout.field.header.`
- **rendering**. the rendering strategy for the field. Examples are heading, text, longtext, crisref, identifier, date, link etc. for metadata field and preview, thubmnail for bitstream field
- **styleLabel**. the style attribute allows to set arbitrary css styles to the metadata's label
- **styleValue.** the style attribute allows to set arbitrary css styles to the metadata's value
- **labelAsHeading.** if true the metadata label is displayed below the metadata label. If false metadata label is displayed along the metadata label.


- **valuesInline.** if true when metadata has multiple values they are displayed one along the others, if false they are displayed one belong the others.


*Box grid system*

In a very similar way to what we have for tabs, the grid system is used also to locate metadata within a metadata box. The grid system uses a series of rows and cells to layout and align metadata, following these rules :

- Every row can contains one or more cells
- Every cell can contain one or more metadata.
- All cells inside the same row are displayed one next to the other
- All metadata inside the same cell are displayed one below the other




*Rendering Types*

DSpace Cris 7 provides some types of ready-to-use rendering but it is possible to create new ones; for further details refer to the paragraph **Defining a new rendering.**

The list of default renderings:

- **heading**
- **text**
- **longtext**

- **link**
- **link.label**
- **date**
- **identifier**
- **crisref**
- **thumbnail**
- **attachment**

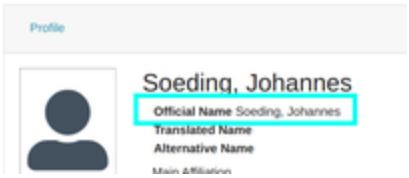
to show a certain field with one of renderings listed it is necessary to set one of names indicated in the “rendering” property of the field.

Heading

type of rendering commonly used for headings. It shows the metadata value in a container with css class h2.


Text

used to show short text information. If there is a label in the database record that defines the field, it will be shown, in bold, before the metadata value.




Longtext

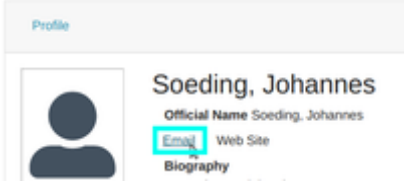
longtext rendering is used to display very long texts. It provides a "show more" mechanism that allows you to view, as a preview, the first 3 lines of the text and by clicking on it you can show / hide the additional lines present.


Link

generates a link that has as displayed text and href the value of the metadata associated with the field

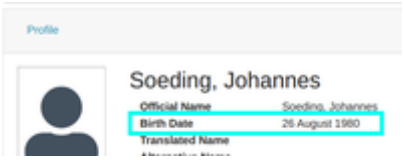

link.label

generates a link that has the metadata value as href and as text a i18n value if the label contains an i18n key, the label text otherwise



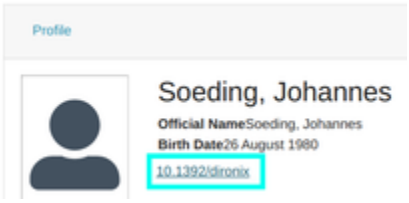
Date

formats the metadata value as a date in the current locale, ex. 2020-08-25 -> 25 August 2020




Identifier

This rendering type build a dynamic link from the identifier present in the metadata. If the metadata value is an http, https, ftp or ftps url the component shows an html anchor with the metadata value in href and text, otherwise if the value is an external identifier (ex. doi:xxxxxxx) the component shows an anchor with href valorized with url of external source and the identifier value for text.
It's possible to force a certain type of identifier using the related subtype, ex. if the metadata value contains a doi identifier in the shape xxxxxxxx (without the URN "doi:") to force doi rendering, use the subtype **identifier.doi** for the rendering field.

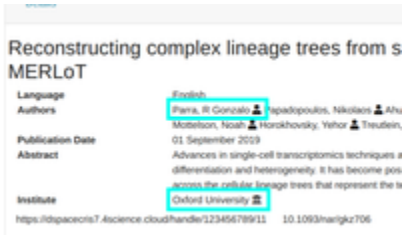



The mapping between the urn and the base url used by the resolver is defined in the `environment.common.ts` file, see the excerpt below

```
crisLayout: {
    urn: [
      {
        name: 'doi',
        baseUrl: 'https://doi.org/'
      },
```

CrisRef

This rendering build a dynamic link to the authority associated at metadata, it shows an icon based on the referenced entity type. To configure an icon for specific entity type you can add an entry at path **layout.crisRef** of client configuration file. The new entry must has two properties, entityType and icon. These properties must set with the entity type name (ex. Person) and font awesome icon (ex. fa fa-user), respectively. If for a specific entity type isn't configured an icon the default will be showed.




**Thumbnail**

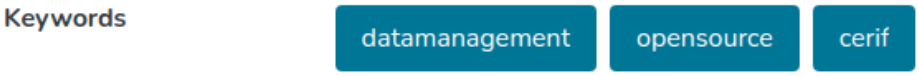
This rendering is used to show a thumbnail of the item, if there is one or more fields of this type, the content will be shown on the left side of the row in which it is contained
as shown in the following image:


**Attachment**

This rendering creates a link to the files attached to the item


**Tag**

This rendering creates a button for each metadata value




**Valuepair**

This rendering should be used for metadata associated to a value-pair in order to render the display value instead of the stored value. The rendering must be provided using the valuepair name as subtype for instance, **valuepair.common_language**


#### Relation Box

Out of box another type of box named **Relation Box** is available. This box is bound to a `DiscoveryConfiguration` that can be parameterized with the `uuid` of the item. The `defaultFilterQueries` of the `DiscoveryConfiguration` can contains the placeholder `{0}` that will be replaced at runtime with the `uuid` of the item. Please refer to next paragraph for further details about how to set and configure such queries.

*Relation Discovery Configuration*

A powerful out-of-the-box box type is provided: **Relation.** This box is populated with Item's linked objects found via a Discovery query.

Discovery queries are configured via `discovery.xml` file (`<dspace-install-dir>/config/spring/api/discovery.xml`) , in `org.dspace.discovery.configuration.DiscoveryConfigurationService` bean. A map's entry for each relation is provided, standard pattern for key is `RELATION.<Entity>.<relationName>`. Entity ans relationName must match to what reported in xls file, box sheet, 'ENTITY' and 'SHORTNAME' entries for each relation. For example, `RELATION.Project.researchoutputs` identifies the relation that will be used to populate 'researchoutupts' box configured in excel file for entity 'Project'.

Map's entry is a reference to a `DiscoveryConfiguration` instance that has to be properly configured in an ad-hoc section. In this section informations about sidebar's facets and search filters to be included in the box, results sorting, results per page, …

The **core part of** `DiscoveryConfiguration`, for relation's set up is `defaultFilterQueries`. This section contains one or more filter queries to be performed, given Item's uuid, to find linked Items. In case many filter queries are provided, such queries are executed in sequence: the second query filters first query's results and so on.

This is the query that retrieves projects related to a person: `projectinvestigators_authority:{0}` where {0} is a placeholder for UUID of the person. In case of inverse relations, queries are more complex and a subquery is needed. For example this is the query that finds Projects belonging to every person affiliated to an OrgUnit, given OrgUnit UUID: `'{'!join from=search.resourceid to=projectinvestigators_authority fromIndex=search'}'person.affiliation.name_authority:{0}`

A full example of how relation that finds a Person pubblications is configured:

```
<bean id="relationAuthorPublicationsConfiguration" class="org.dspace.
discovery.configuration.DiscoveryRelatedItemConfiguration">
    <!--Which sidebar facets are to be displayed-->
    <property name="sidebarFacets">
        <list>
            <ref bean="searchFilterAuthor" />
            <ref bean="searchFilterEntityType"/>
            <ref bean="searchFilterSubject" />
```

```
        </list>
    </property>
    <!-- Set TagCloud configuration per discovery configuration -->
    <property name="tagCloudFacetConfiguration" ref="
defaultTagCloudFacetConfiguration"/>
    <!--The search filters which can be used on the discovery search
page-->
    <property name="searchFilters">
        <list>
            <ref bean="searchFilterAuthor" />
            <ref bean="searchFilterEntityType"/>
            <ref bean="searchFilterSubject" />
        </list>
    </property>
    <!--The sort filters for the discovery search-->
    <property name="searchSortConfiguration">
        <bean class="org.dspace.discovery.configuration.
DiscoverySortConfiguration">
            <!--<property name="defaultSort" ref="sortDateIssued"/>-->
            <!--DefaultSortOrder can either be desc or asc (desc is
default)-->
            <property name="defaultSortOrder" value="desc"/>
            <property name="sortFields">
                <list>
                    <ref bean="sortTitle" />
                </list>
            </property>
        </bean>
    </property>
    <!--Any default filter queries, these filter queries will be used
for all
            queries done by discovery for this configuration -->
    <property name="defaultFilterQueries">
        <list>
            <!--Only find related items. The placeholder {0} will be
replaced with scope (UUID of item)-->
            <value>author_authority:{0} AND entityType_keyword:
Publication</value>
        </list>
    </property>
    <!--Default result per page  -->
    <property name="defaultRpp" value="10" />
    <property name="hitHighlightingConfiguration">
        <bean class="org.dspace.discovery.configuration.
DiscoveryHitHighlightingConfiguration">
            <property name="metadataFields">
                <list>
                    <bean class="org.dspace.discovery.configuration.
DiscoveryHitHighlightFieldConfiguration">
                        <property name="field" value="relationship.type"
```

```
/>
                        <property name="snippets" value="5"/>
                    </bean>
                </list>
            </property>
        </bean>
    </property>
</bean>
```

### *Metrics Box*

The metrics box is responsible to display metrics values for the displayed item.

When a box of type metric is detected, his configuration is fetched. The metrics box configuration simply contains an array of types of metrics that belong to the box. The order is important because existing metrics are displayed following such order.

Then Metrics values are processed by a service method named *getMatchingMetrics*. It filters by types, sorts and organizes metrics in rows as defined by the box field 'maxColumn' which specifies how many metrics must appear in each single row.

Finally layout components are instantiated based on the type of each metric.

The *metricLoaderService* keeps the mapping between metric type and component type and, occasionally, an external script that must be loaded to display the metric correctly.

For bundles sizes reasons, scripts are lazily loaded once the first time they're needed.

Simple metrics types extend the abstract *BaseMetricComponent*. Metrics which require external script extend the *BaseEmbeddedMetricComponent* which takes care to manage the script execution. Since the script could take time to be loaded, the baseEmbeddedMetric follows a retry strategy. This strategy can be driven through two global variables.

***METRIC_SCRIPT_TIMEOUT_MS*** = 500;
***METRIC_SCRIPT_MAX_RETRY*** = 3;

which specify the max number of attempts and a delay between each.

Currently 4 types of components exists:

- MetricDspacecrisComponent (BaseMetricComponent):

  Display a generic dspacecris metric using all the information coming from the server. This type is also the default in case no mapping with the metricType exists.
- Type: **google-scholar**

  MetricGooglescholarComponent (BaseMetricComponent)

  Display a Google Scholar Metric, by using the link inside the metric.remark field.
- Type: **altmetric**

  MetricAltmetricComponent (BaseEmbeddedMetricComponent)

As per metadata boxes, metric boxes are returned by the server only if content to visualize exists. At the moment possible existing metrics are visible only to logged in users, so for anonymous sessions metric boxes are always hidden.

#### Customize the default items layout

In DSpace-CRIS 7 the default layout for displaying items can be overwritten with customized Angular components.

To make the layout customization flexible, it is possible to overwrite the layout in different levels:

- Customization of the layout by overwriting style
- Overwriting a specific box
- Defining a new field rendering type

Within the dspace-angular project there is the `CrisLayoutModule` module under the path `src/app/cris-layout`, which is responsible for managing the layout of the items.

### *Customization of the layout by overwriting style*

The most important properties of the new layout have css variables that can be changed, or overwritten in case you're using your custom DSpace theme.

All the variables are available in the file `src/styles/_custom_variables.scss` (all the ones that start with the prefix `--ds-cris-layout`) and allow to change color, width or height of some layout's elements.

### *Customization of a specific box*

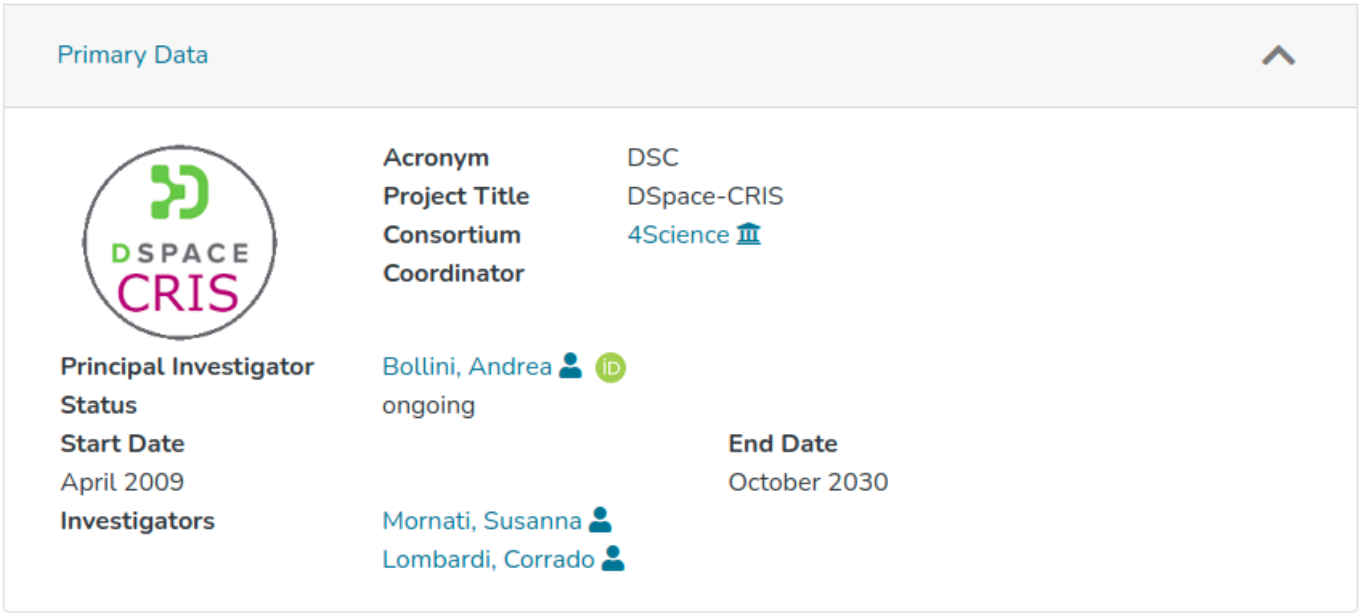
All the components used for rendering a specific type of bos are collected in the folder `src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/boxes`

Every type of box has in common a component container (`src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/cris-layout-box-container.component.ts`) that has the aim of rendering the box within a collapsible accordion

So it is possible to customize `cris-layout-box-container.component.ts` in order to change the container for all the boxes or to customize only a specific type of boxes. To deal with it every box components are using the `@RenderCrisLayoutBoxFor` decorator that has two params :

- `boxType` : that defines for wich type of box the component is used
- `hasOwnContainer` : that defines if the box should use the common container or its own one

This level of customization will allow to overwrite, for example, the section like in the following image:




### *Defining a new field rendering type*

To define a new field rendering the following steps are needed:

- add the new rendering type into the enumeration `FieldRenderingType` (contained in `src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/boxes/metadata/rendering-types/metadata-box.decorator.ts`)
- create new component uder the path `src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/boxes/metadata/rendering-types`
- extend the `RenderingTypeValueModelComponent`object (present in `src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/boxes/metadata/rendering-types`) in case the new rendering should handle only one matadata value per time
- extend the `RenderingTypeStructuredModelComponent`object (present in `src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/boxes/metadata/rendering-types`) in case the new rendering should handle all the matadata values per time
- add the decorator `@MetadataBoxFieldRendering(FieldRendetingType.NEW_RENDERING_TYPE)`(contained in `src/app/cris-layout/cris-layout-matrix/cris-layout-box-container/boxes/metadata/rendering-types/metadata-box.decorator.ts`)

The new component will inherit the `box`, `item` , `field` and `metadataValue` (only when extending `RenderingTypeValueModelComponent`object ) variables, valorized with the information of item to display and the current field, respectively.

**Update the Tab/Box content**

The content of a tab is normally updated on navigation events. Sometimes it could be necessary to update the content from within a tab/box when specific events occurs. This can be achieved programmatically calling specific event emitters on the abstract component `CrisLayoutTabModelComponent` and `CrisLayoutBoxModelComponent`.

`refreshBox` must be used to reload a single box content.

`refreshTab` must be used to reload the entire opened tab.

### Item reference resolution

As a metadata value's authority it is possible to specify that that metadata value will be linked to a particular item when it is submitted to the system. To do this, you can set the authority with a value that has the syntax **will be referenced::<reference-type>::<value>**, where:

- **reference-type** indicates the type of the reference (for example ORCID or DOI)
- **value** indicates the value for which to search for the item to be referenced

For example, if a metadata dc.contributor.author has will be referenced::ORCID::0000-0001-2345-6789 as authority it have to be resolved by setting the uuid of the item which has a person.identifier.orcid equal to 0000-0001-2345-6789.

The resolution of the reference can take place in two moments:

- when the item with a metadata with authority will be referenced is deposited and the item to be referenced is already present in the system
- when an item is deposited and other items have a reference to an item that matches one of the metadata of the deposited item

In the first case the reference is resolved by the **CrisConsumer**, that through all the implementations of the interface org.dspace.authority.service.**ItemSearcher** defined in the map handled by the class org.dspace.authority.service.**ItemSearcherMapper** tries to find an item that matches the reference.

In the other case, instead, the references are resolved by a special consumer, implemented by the org.dspace.authority.**ItemReferenceResolverConsumer** class, which uses all the beans with type org.dspace.authority.service.**ItemReferenceResolver** to search for items with different strategies.

#### *Update previously referenced items with new metadata value*

When an item is added and other pre-exsisting items have a reference to this newly inserted, it is possible to update the metadata field of the pre-exsisting items (Those ones added prior to the referencig item) with the value of the newly added.

To perform this metadata substitution it is needed to set the `cris.item-reference-resolver.override-metadata-value` property to true (default value is false). This property can be found under the **cris.cfg** file at the path: ***dspace/config/modules***

An example of metadata replace (property set to true) is: 3 pubblications are added and liked to a non-existing author named: "John S.". This author is then added with him full name "John Smith". Setting up `cris.item-reference-resolver.override-metadata-value` to ***true*** will replace in each pubblications of his (any item linked to this author) the name from "John S." to "John Smith" when the refecences are updated.

##### ItemSearcher

The classes that implement the org.dspace.authority.service.**ItemSearcher** interface are used to locate items according to a certain strategy. There are currently two implementations available:

- org.dspace.authority.service.impl.**ItemSearcherById** search an item by UUID
- org.dspace.authority.service.impl.**ItemSearcherByMetadata** search for an item that has a metadata related to the metadata field configured for the particular bean instance with the given value. The search is done on Solr also searching for items not yet archived.

The ItemSearcher are collected in a map handled by the org.dspace.authority.service.**ItemSearcherMapper** class that associates each of them with a particular reference type. Configuration example (cris-plugin.xml):

```
<bean class="org.dspace.authority.service.ItemSearcherMapper" name="org.
dspace.authority.service.ItemSearcherMapper">
        <constructor-arg index="0">
                <map>
                        <entry key="UUID">
                                <bean class="org.dspace.authority.
service.impl.ItemSearcherById"></bean>
                        </entry>
                        <entry key="ORCID" value-ref="
```

```
itemSearcherByORCID"/>
                        <entry key="RID" value-ref="itemSearcherByRID"/>
                        <entry key="ISNI" value-ref="itemSearcherByISNI"
/>
                        <entry key="DOI" value-ref="itemSearcherByDOI"/>
                </map>
        </constructor-arg>
        <constructor-arg index="1">
                <bean class="org.dspace.authority.service.impl.
ItemSearcherById"></bean>
        </constructor-arg>
</bean>

<bean class="org.dspace.authority.service.impl.ItemSearcherByMetadata"
name="itemSearcherByORCID">
        <constructor-arg value="person.identifier.orcid"></constructor-
arg>
        <constructor-arg value="ORCID"></constructor-arg>
</bean>

<bean class="org.dspace.authority.service.impl.ItemSearcherByMetadata"
name="itemSearcherByRID">
        <constructor-arg value="person.identifier.rid"></constructor-
arg>
        <constructor-arg value="RID"></constructor-arg>
</bean>

<bean class="org.dspace.authority.service.impl.ItemSearcherByMetadata"
name="itemSearcherByDOI">
        <constructor-arg value="dc.identifier.doi"></constructor-arg>
        <constructor-arg value="DOI"></constructor-arg>
</bean>

<bean class="org.dspace.authority.service.impl.ItemSearcherByMetadata"
name="itemSearcherByISNI">
        <constructor-arg value="person.identifier.isni"></constructor-
arg>
        <constructor-arg value="ISNI"></constructor-arg>
</bean>
```

#### ItemReferenceResolver

The classes that implement org.dspace.authority.service.**ItemReferenceResolver** are instead used to search for any items that refer to an item that has just been deposited. All the bean instances that implement this interface are collected by the org.dspace.authority.service.impl.**ItemReferenceResolverServiceImpl** class which allows to cycle on them to attempt to resolve the reference with different strategies.

Currently only one class implements the interface and it is class **ItemSearcherByMetadata** which also implements the **ItemSearcher** interface. In this way, therefore, this class allows both to resolve the references on one side and on the other and, once the metadata to search for has been defined, it does not require further configurations. This class identifies the items that have a reference to the given one by searching on solr all the items with a metadata having authority of the type will be referenced::<reference-type>::<value>, using the configured reference type and taking the value from a specific metadata of the item. The metadata to search for are all those authority controlled that are associated with an entity type consistent with the given item.

For example, given the ItemSearcherByMetadata configured for ORCID and a Person item with a person.identifier.orcid equals to 0000-0001-2345-6789, that ItemSearcher will search for all the items that have a metadata authority controlled related to Person (as dc.contributor.author or dc.contributor.editor) with an authority equals to will be referenced::ORCID::0000-0001-2345-6789.

## Content Subscription

DSpace-CRIS 7 users can subscribe to Communities, Collections and Items. Once an user is subscribed, he / she will receive via email periodical updates.

#### Subscriptions types

A subscription can be of two types:

- **CONTENT:** The user will receive periodical emails about content updates affecting subscribed Communities, Collections or Items, i.e. new items into put into a collection, updated items, etc.
- **STATISTICS:** The user will receive periodical emails about subscribed content statistics, i.e. how many views the contend had, how many downloads, etc. Statistics values are absolute, not related to the notification frequency: for example if a user subscribe to a Publication statistics updates with a weekly frequency (see next paragraph), and for this Publication there are number of views available, notification will contain number of views such publication had so far, not how many views it had in last week. When available, notification will contain also the value the same statistic indicator had the previous month and the previous week.

#### Subscriptions frequencies

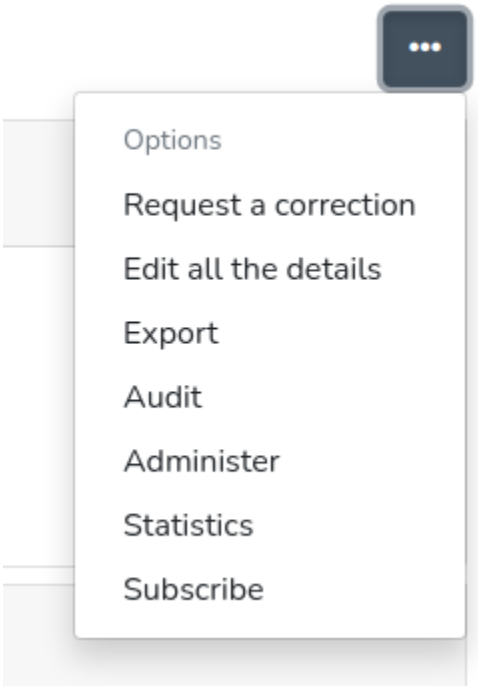
The subscribing user can select the frequency he / she wants to receive notifications. Available frequencies are:

- **DAILY**: The user will receive every day an email containing last day content updates, or statistics related to the Community, Collection or Item to which he / she is subscribed.
- **MONTHLY**: The user will receive every month an email containing last month content updates, or statistics related to the Community, Collection or Item to which he / she is subscribed.
- **WEEKLY**: The user will receive every week a notification containing last week content updates, or statistics related to the Community, Collection or Item to which he / she is subscribed.

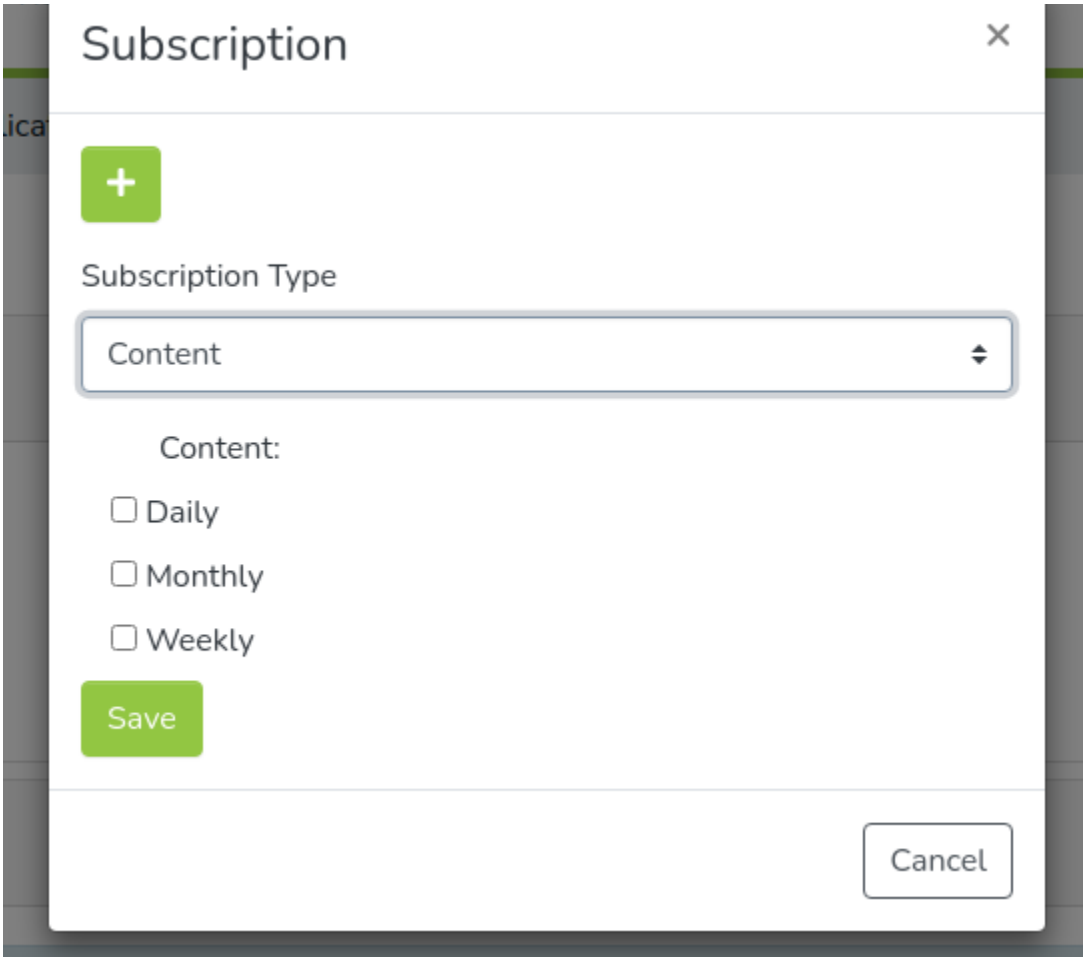
Despite of how many Communities, Collections or Items the user has subscribed to, updates will be grouped. This means that the user will receive a single email per subscription type (content or statistics) and frequency containing all updates. For example, if he / she has subscribed for daily updates regarding content of 3 communities, 4 collections and 5 items, and for daily updates regarding statistics of 8 items, every day two emails will be sent to this subscriber: one with 12 content updates, and one with statistics information of 8 items. The same applies for monthly and weekly subscriptions.

#### Subscriptions Management

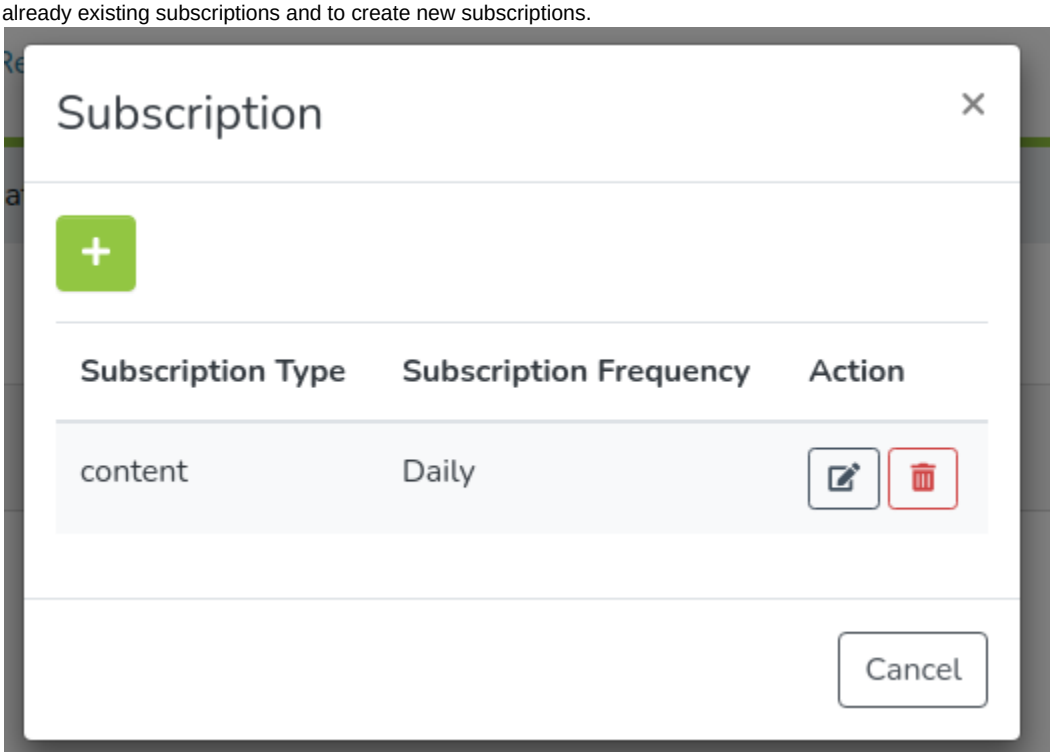
Content subscription can be done from Communities, Collections or Item context menu, “Subscribe” option



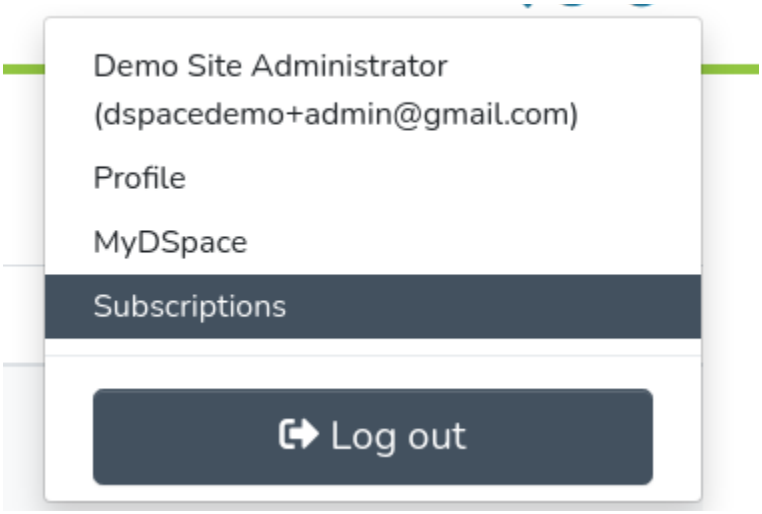

once selected, a modal is shown where user can select type and frequency. Multiple frequency selection is possible.




In case user already has a subscription in place for the Community, Collection or Item he / she is subscribing to, it is possible to edit or delete already existing subscriptions and to create new subscriptions.




A summary page is reachable from the user menu. From this page the user can view, edit or delete all his / hers existing subscriptions.




| Subject | Subscription Type | Subscription Frequency | Action |
|---|---|---|---|
| item Bollini, Andrea | content | Monthly, Daily | |
| item DSpace-CRIS | content | Daily, Weekly | |

Notifications via email

Notifications are sent via email by the `subscription-send` command, which can be started as a process in the processes DSpace-CRIS7 section by an user having administrator privileges, or as command from command line.

This command has two mandatory parameters used to identify for which subscriptions notifications must be sent

- `-t` or `--Type` representing notification type to be sent, one between "content" and "statistics"
- `-f` or `--Frequency` representing the notification frequency: possible values are "D" for daily updates, "M" for monthly updates and "W" for weekly updates.

For example, `{dspace-install-dir}/bin/dspace subscription-send -t content -f D` will send notifications to all users which want to receive daily content updates, with last day updates affecting subscribed content, `{dspace-install-dir}/bin/dspace subscription-send -t content -f M` will send notifications to all users which want to receive monthly content updates, with last month updates affecting subscribed content, while `{dspace-install-dir}/bin/dspace subscription-send -t statistics -f W` will send notifications to all users which want to receive weekly content statistics update

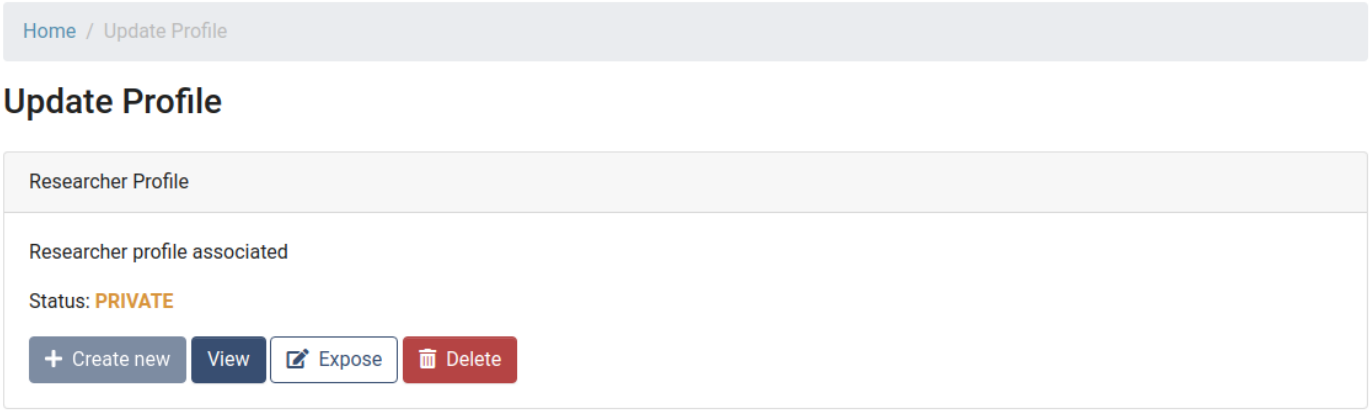
## Researcher Profile

Users can create a Reseacher profile that allow their names to be look up during submission to easily associate pubblications to their profile. In addition an user can make the profile public to have a personal page showing his research activities and publications.

A user can associate to him a researcher profile under the “Profile” section:


Once the user have created a new researcher profile he will be able to:

- View the personal page related to the profile
- Change the profile's visibility (hide or expose)
- Delete the profile




#### Profile implementation

The researcher profile is modeled using an item which, during the creation phase, will be configured with the following metadata:

- **cris.sourceId:** metadata whose value is the id of the eperson who created the profile
- **cris.owner:** metadata that has as value the full name of the eperson associated with the profile and as authority the id of the same eperson

The item is created in a specific collection which is identified using the following strategy:

- if the property researcher-profile.collection.uuid is set, the given uuid is used to find the collection
- otherwise, the collection is found by searching for a specific relationship.type which can be configured using the researcher-profile.type property (or "Person" by default); the search is successful if exactly one collection has been found.

If no collection is found using the previous strategy, an error occurs and no profile is created.

The visibility of the profile is handled adding or removing the ANONYMOUS group policy on the related item. By default the item is created without the READ policy associated with the ANONYMOUS group and therefore the profile visibility is private.

#### Profile deletion

The deletion of the profile can be of two types, soft or hard:

- the soft (or logical) deletion of the profile does not cause a real deletion of the item associated with the profile but releases this item to the user by deleting the cris.owner metadata.
- the hard (or physical) deletion instead causes the actual deletion of the item in an irreversible way.

By default, hard deletion is disabled: to change this behavior you can enable it using the boolean property **researcher-profile.hard-delete.enabled**

#### Automatic claim

During login, if the user is not yet related with a Researcher Page, the application looks for a match based on the current user's email or ORCID; The system could be also configured to accept other rules for matching.

The automatic claim is done, if possible, immediately after login using a mechanism that allows to invoke the loggedIn methods of all the beans present in the context that implement a specific interface called **PostLoggedInAction**. Therefore, by adding other beans that implement this interface, it is possible to add further automatic claim strategies or perform other actions always after the user login.

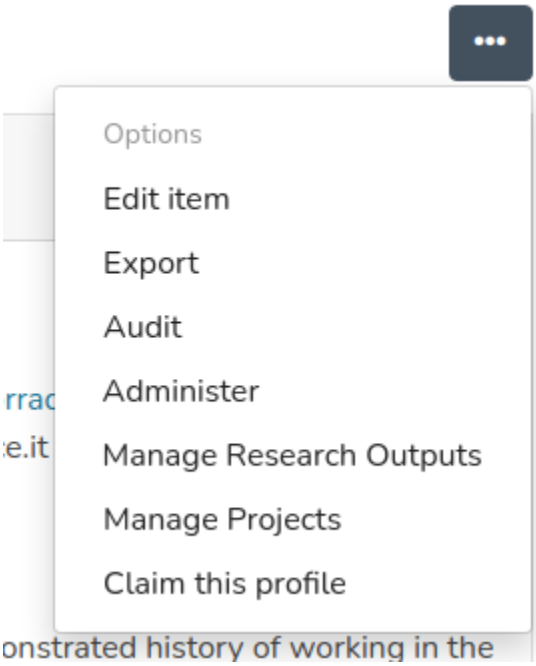
##### Claim by user's email

The automatic claim by email is done searching an unique item that has a **crisrp.email** metadata equals to the current user's email. The coupling is done if the current user does not already have an associated researcher profile and if the user identified through the email is unique. This strategy is implemented by the **ResearcherProfileClaimByEmail** class, that implements the PostLoggedInAction interface and that is marked as a Spring Component.

#### Manual claim

If the user is not yet related with a Researcher Page, it is possible to create a new Researcher Profile by claiming an existing Item in a defined collection (i.e. a "Person" collection) that has not been yet claimed by other users.

If profile can be claimed, a “Claim this profile” option will be available in item’s menu:




In this case, the claim is proposed to the logged user via the contextual menu. Once the claim is performed, the user will have claimed profile linked to his Researcher Profile and no other users are able to claim the same profile. This link is created by setting metadata **cris.owner** in claimed item. Once Researcher profile is deleted, the claimed object, by default, will remain archived and previously set **cris.owner** metadata will be removed.

To enable deletion of both Researcher profile and linked items, following the configuration key “**researcher-profile.hard-delete.enabled**” must be set to true (**CAUTION**: setting this key to true will cause previously claimed Item removal from the repository)

To enable this feature, at least one entity type must be enabled as “claimable”, by setting following key in configuration file. Many values could be set.

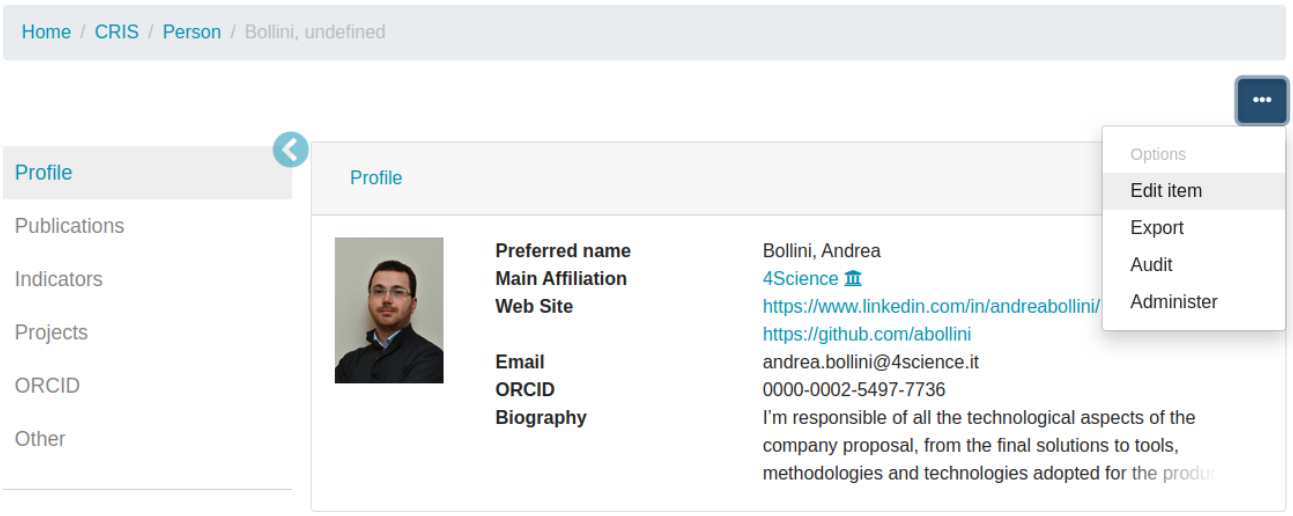
**claimable.entityType=**<entity type of item that could be claimed>

## Edit Item in Submission mode

DSpace-Cris 7 introduce, in addition to administrative edit function, the item edit in submission mode.

### Access to the Functionality

This modality is accessible from the item details page for the items that have configured the dynamic layout of DSpace-Cris 7.
In the item details page, one or more menu entries are showed within the context menu.



The menu entry name showed in the dropdown is configurable with a i18n label on angular side. The label key is `context-menu.actions.edit-item.btn.<configuration_name>` where `<configuration_name>` must be replaced with the name of the edit configuration. For more details read the "Edit Modes Configuration" paragraph.

#### Edit Modes Configuration

to configure one or more edit modalities for a specific entity type is necessary add the configuration in the property ***editModesMap*** of bean ***org.dspace.content.edit.service.impl.EditItemModeServiceImpl*** inside ***edititem-service.xml*** file.

The first step is the creation of a new submission-process definition for the entity type in the `item-submission.xml` file. For example :

```
    <submission-process name="edit_admin_publication">
      <step id="extraction" />
      <step id="publication" />
      <step id="publication_indexing" />
      <step id="publication_bibliographic_details" />
      <step id="publication_references" />
      <step id="upload" />
    </submission-process>
```

Once the submission definitions are created you have to add a new entry in the property ***editModesMap*** of the bean ***org.dspace.content.edit.service.impl.EditItemModeServiceImpl***.

In an instance without edit modes configuration the property editModesMap is configured as follow:

```
<bean class="org.dspace.content.edit.service.impl.
EditItemModeServiceImpl">
</bean>
```

The editModesMap property is a Map that has as key the entity type name (in lowercase) and as value a list of edit modes:

```
<bean class="org.dspace.content.edit.service.impl.
EditItemModeServiceImpl">
    <property name="editModesMap">
        <entry key="publication">
            <list>
                <bean class="org.dspace.content.edit.EditItemMode">
                    <property name="name" value="MODE1" />
                    <property name="label" value="edititem.mode.test2"
/>
                    <property name="security">
                      <value type="org.dspace.content.edit.
EditItemModeSecurity">
                         ADMIN
                      </value>
                    </property>
```

```
                    <property name="submissionDefinition" value="
edit_admin_publication" />
                </bean>
                <bean class="org.dspace.content.edit.EditItemMode">
                    <property name="name" value="FULL" />
                    <property name="security">
                      <value type="org.dspace.content.edit.
EditItemModeSecurity">
                        OWNER
                      </value>
                    </property>
                    <property name="submissionDefinition" value="
edit_owner_publication" />
                </bean>
            </list>
        </entry>
    </property>
</bean>
```

Below is the detail of the configuration of a specific edit mode:

```
<bean class="org.dspace.content.edit.EditItemMode">
    <property name="name" value="MODE1" />
    <property name="label" value="edititem.mode.test2" />
    <property name="security">
      <value type="org.dspace.content.edit.EditItemModeSecurity">
        ADMIN
      </value>
    </property>
    <property name="submissionDefinition" value="
edit_admin_publication" />
</bean>
```

As you can see, an edit mode is represented by class ***org.dspace.content.edit.EditItemMode***, this class has follow properties:

- **name**: this is a unique identifier for the edit mode at entity type level;
- **label**: this is an optional property, if is valorized the UI will use its value as i18n key otherwise the UI will use the value of name property;
- **submissionDefinition**: contains the name of the submission definition to use with the current edit mode;
- **security**: defines the security level for the current edit mode and its visibility. This property is an Enum (*org.dspace.content.edit.EditItemMode*) and accepts one of the following values:
  - **ADMIN**: this value allows only the Administrator to use this edit mode
  - **OWNER**: this value allows only the item owner to use this edit mode
  - **ADMIN_OWNER**: this value allows the Administrator and item owner to user this edit mode
  - **CUSTOM**: this value is used to define a fine-grained security access. This security level is explained follow

*Configure a custom security level*

This security level allows to use the edit mode only to the users/groups that are present inside a list of metadata associated to the item. When configuring the CUSTOM security levels you need to set two other properties within the configuration: users and groups. These properties contains a list of metadata used to check if a specific user/group are allowed to use this edit mode

```
<bean class="org.dspace.content.edit.EditItemMode">
    <property name="name" value="MODE1" />
    <property name="label" value="edititem.mode.test2" />
    <property name="security">
      <value type="org.dspace.content.edit.EditItemModeSecurity">
        CUSTOM
      </value>
    </property>
    <property name="submissionDefinition" value="
edit_admin_publication" />
    <property name="groups">
        <list>
            <value>cris.groups</value>
            <value>dspace.groups</value>
        </list>
    </property>
    <property name="users">
        <list>
            <value>cris.users</value>
        </list>
    </property>
</bean>
```

previous configuration are visible only to users that are present in the ***cris.users metadata*** and the users that are part of the groups present in ***cris.groups*** or ***dspace.group*** metadata.

### Automatic suggestion of new publications

This feature is the result of the OpenAIRE Advance Open Call for Innovation Project “Enrich local data via the OpenAIRE Graph” awarded to 4Science see https://www.4science.it/en/2020/09/07/openaire-advance-premia-4science-per-il-progetto-enrich-local-data-via-the-openaire-graph-fase-2/

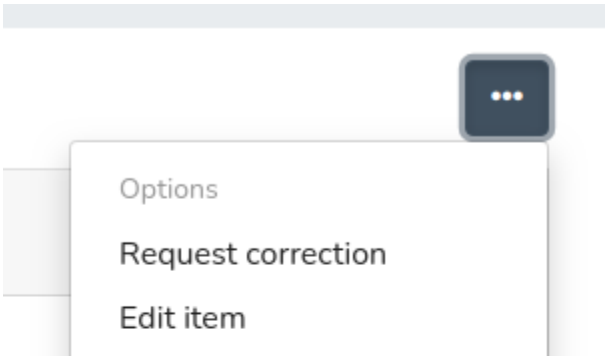
The detailed documentation is maintained on a dedicated project website https://4science.github.io/oaire-eld/#/

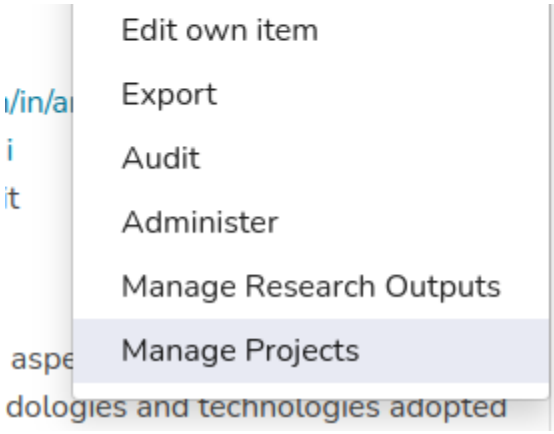
### Entities hide, sort and selection functionality

This functionality is driven by DSpace7 Relations framework, used to drive specific features or business workflows such as correction requests, selected list of objects, etc.

This relations framework is different than the authority framework (see “How To Manage Relationships between items paragraph in DSpace Items and CRIS Entities page, https://4science.atlassian.net/wiki/spaces/DTD/pages/1495695372/DSpace+Items+and+CRIS+Entities#How-to-manage-relationships-between-items) used to describe or add context to an object.

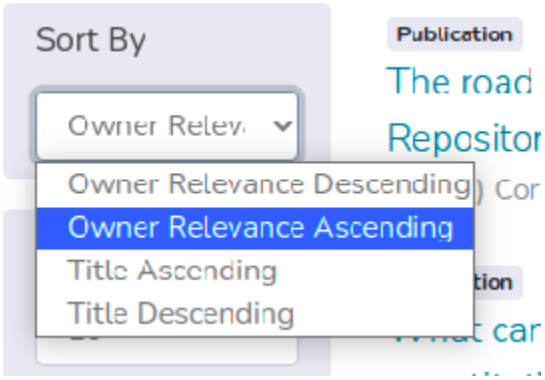
#### Selected lists of objects



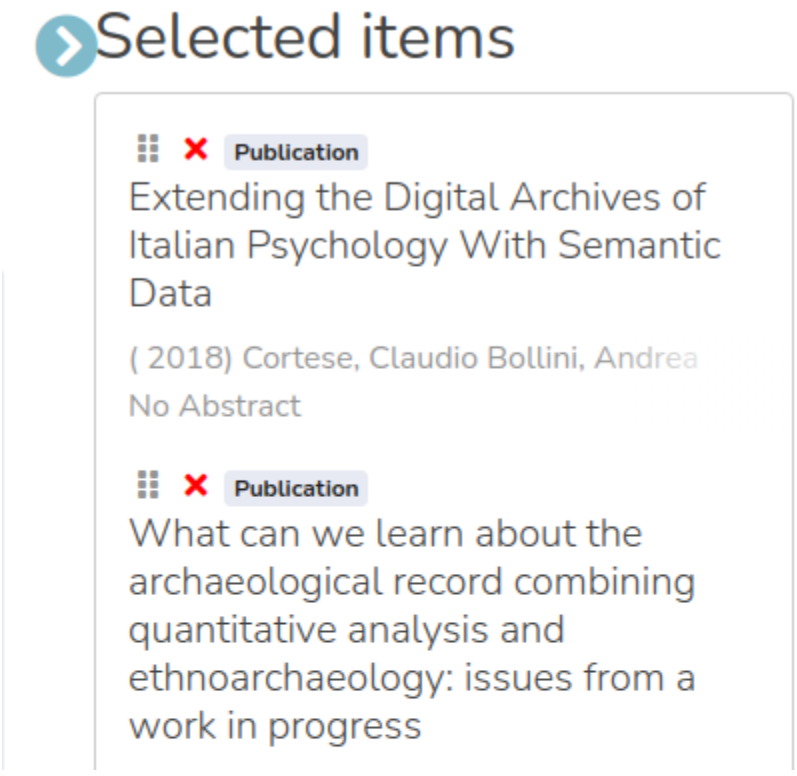



Users who have edit permissions on a given DSpace-CRIS entity can use "management" feature to perform operations on DSpace-CRIS entities related to this entity by mean of DSpace-CRIS inverse relations mechanism, following operations are allowed:

- **Hide**: a related entity won't be visible within list of related items
- **Select / Sort:** selected entities will appear on top of list of related entities, when "Owner Relevance Ascending" sorting criteria (currently set as default) is selected.




The sorting order of selected entities can be changed by dragging and dropping selected entities in management page.




Needed relationship types should be initialized by importing file "**hide-sort-relationships.xml**" with following command, see next “Relationship type set-up” paragraph for a quick introduction to DSpace7 relationship set up.

```
{dspace.install.dir}/bin/dspace initialize-entities -f ../config/entities/hide-sort-relationship-types.xml
```

#### New inverse relation hide and sort settings

Out of the box, relationship types to perform hide and sorting of all entities taking part to inverse relations defined in **discovery.xml** (keys of type "RELATION.<target entity>.<name>) file are defined and ready to be imported with above command.

In case new inverse relations are defined, with a new “target entity” or a new “name”, new relationship types needs to be created to perform hide and sorting, with following syntax

**select/sort relationship**
*left_type*: null*
*right_type*: <inverse relation target type>
*leftward_type*: is<relationship name>SelectedFor
*rightward_type*: hasSelected<relationship name>
*left and right min cardinalities*: 0

*see “Relationship types involving multiple entity types” paragraph

For this kind of relationships, entry in configuration file to use only right place (relationship.place.onlyright) needs to be provided (see Relationship place management paragraph for further details about this functionality)

**hide relationship**
*left_type*: null
*right_type*: <inverse relation target type>
*leftward_type*: is<relationship name>HiddenFor
*rightward_type*: notDisplaying<relationship name>
*left and right min cardinalities*: 0

For example, given inverse relation "*RELATION.Person.researchoutputs*", following relationship types are defined

**select/sort relationship**
*left_type*: null
*right_type*: Person
*leftward_type*: isResearchoutputsSelectedFor
*rightward_type*: hasSelectedResearchoutputs
*left and right min cardinalities*: 0

**hide relationship**
*left_type*: null
*right_type*: Person
*leftward_type*: isResearchoutputsHiddenFor
*rightward_type*: notDisplayingResearchoutputs
*left and right min cardinalities*: 0

and the entry

*relationship.places.onlyright=null::Person::isResearchoutputsSelectedFor::hasSelectedResearchoutputs is*

inserted in application configuration

### Relationship types set-up

DSpace7 relationship types can be imported into database with CLI command
`dspace initialize-entities -f {xml-file}`
where xml-file is an xml representation of Relationship type structure. A sample can be found at path
`{dspace.install.dir}/config/entities/relationship-types.xml`

### Relationship types involving multiple entity types

It is possible to define relationship with null type as left or right type (null on both sides is forbidden) to identify relationship on which only left (right) entities taking part to them must be of a defined type, while other entity can be of different types, thus a relation type defined as

*left_type*: null
*right_type*: Person
*leftward_type*: isResearchoutputsHiddenFor
*rightward_type*: notDisplayingResearchoutputs
*left and right min cardinalities*: 0

might be used to define relationships involving many items hidden by “right” Person, like the following:

| left item type | left item id | right item type | right item id | leftward_type | rightward_type |
|---|---|---|---|---|---|
| Patent | 123 | Person | 1 | isResearchoutputsHiddenFor | notDisplayingResearchoutputs |
| Publication | 333 | Person | 1 | isResearchoutputsHiddenFor | notDisplayingResearchoutputs |

### Relationship place management

For each DSpace7 relationship established between entities, "leftPlace" and "rightPlace" fields can be defined, representing the placement of this relation among other of the same kind involving one or both of same items taking part to the relation.

For relationship of some types, it is possible to define that placement is to be tracked only for one of the entities taking part to the relation. This behavior is used by relations used to perform selection and sorting, where left and right item represent, respectively, entity related via inverse relation and target item to which this entity is related.

To define this logic, following properties must be defined in application configuration files:
**relationship.places.onlyright** : to define relationship where only right places value is used
**relationship.places.onlyleft** : to define relationship where only left places value is used

Many properties with same key can be defined, with following syntax
<left entity type>::<right entity type>::<relationship leftward type>::<relationship rightward type>

For example, following relationship type is used to keep track of selected research outputs of a person, starting from relation "RELATION.Person.researchoutputs" defined in discovery.xml file

*left_type*: null
*right_type*: Person
*leftward_type*: isResearchoutputsSelectedFor
*rightward_type*: hasSelectedResearchoutputs

This means that for a Person entity, we might have one or many other DSpace-CRIS7 entities related to it (selected), taking part to a relation of the same type. To keep track of this related entities sorting as defined by the user via UI (first selected research output, second selected research output, etc.) positional value will be stored in “rightPlaces”, meaning that this research output is the “nth” selection of this user. Being this relationship 1:n, the following entry in configuration file
`relationship.places.onlyright=null::Person::isResearchoutputsSelectedFor::hasSelectedResearchoutputs`
means that for relationship of this type, only right place value is used.

## ORCID Integration

DSpace-CRIS provides a full integration with ORCID based on the ORCID API v3.0. The full range of ORCID API is supported ranging from the Public API to the Membership and Premium API

- ORCID Authentication
- ORCID Synchronization
- ORCID Registry Lookup
- ORCID Imports
- ORCID Webhook

## Configuration

Integration with ORCID requires the following configuration properties:

- **orcid.domain-url** ORCID domain url
- **orcid.authorize-url** ORCID endpoint to get an OAuth Authorization Code for the specified scopes
- **orcid.token-url** ORCID endpoint to exchange the authorization code for an access token
- **orcid.api-url** The root of the ORCID registry API url
- **orcid.redirect-url** the complete url of the DSpace side rest endpoint on which the browser must redirect the user after logging in on the orcid registry (during the “3 legged OAuth”)
- **orcid.webhook-url** The root of the ORCID registry webhook endpoints
- **orcid.public-url** the url of the public endpoints of ORCID registry
- **orcid.application-client-id** the id credential provided by ORCID after the application registration
- **orcid.application-client-secret** the secret provided by ORCID after the application registration
- **orcid.scope** the list of the access scopes that the application requires; these scopes are used in the OAuth authenticate process where the user grants the specific permission asked for.

The reference configuration file of the features linked to orcid is the **orcid.cfg** file placed in config/modules, while the configuration of the main beans used for the functionalities related to orcid is defined in the file config/spring/api/**orcid-services.xml**.

In this file, in addition to the properties listed above, there are also the default configurations for all the functions related to orcid (webhook, mapping between DSpace entities and ORCID entities, etc.).

For more details about the application registration on ORCID click here.

### ORCID Authentication

DSpace-CRIS allows users to log into the system using their ORCID account and, subsequently, to be able to synchronize data on DSpace with the ORCID registry and vice versa. Furthermore, users who already have a profile on DSpace-CRIS can still connect this profile with their ORCID account by logging into ORCID using the features present on the profile page.

Authentication on DSpace-CRIS via ORCID, as well linking an existing profile, use the **3-legged OAuth** mechanism offered by the ORCID API. The next paragraphs describe how DSpace-CRIS exploits this mechanism to integrate with ORCID; the complete guide on how to integrate with ORCID is available here.

#### Login with ORCID

To allow users to log into DSpace-CRIS via ORCID, it is necessary to add the ORCID authentication methods among those configured. To do this, the following property should be added to the configuration

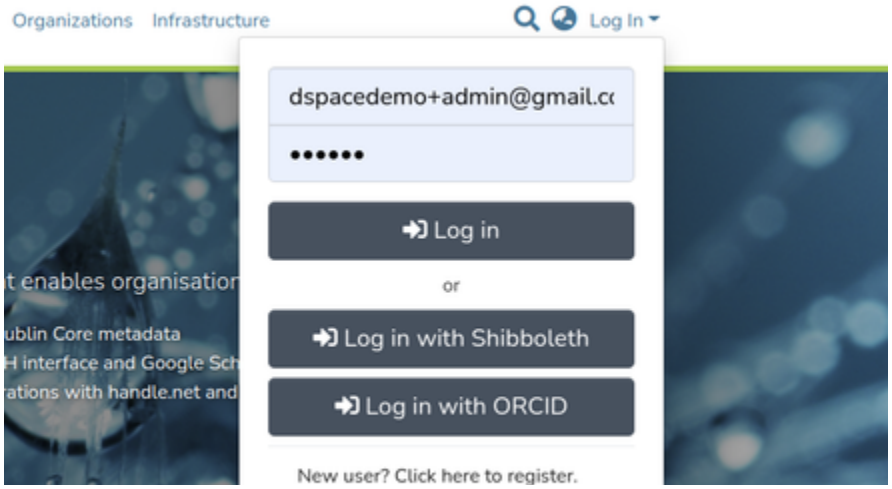
```
plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.OrcidAuthentication
```

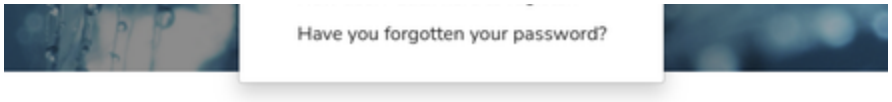
(already present but commented in the **/config/modules/authentication.cfg** file).

In addition, the following configuration properties must be set:

- orcid.authorize-url
- orcid.token-url
- orcid.redirect-url
- orcid.application-client-id
- orcid.application-client-secret
- orcid.scope (must at least have **/authenticate** among the set values)

If all the properties listed above have been set correctly, the DSpace-CRIS login panel should show an additional button to login via ORCID, as shown in the next figure.






When clicked, the user is sent to the ORCID login page, where:

- ORCID asks the user to sign in
- ORCID asks the user to grant permission to your application
- ORCID sends the user back to DSpace-CRIS with an authorization code related to the configured redirect url, which must be associated with a specific endpoint rest which starts the authentication process also on the DSpace-CRIS side.

Authentication is managed by the org.dspace.authenticate.**OrcidAuthentication** class which is triggered by a filter (org.dspace.app.rest.security.**OrcidAuthenticationFilter**) associated with requests on the url specified for the redirect (filter configuration present in the org.dspace.app.rest.security.**WebSecurityConfiguration** class).

Once the redirected request has been received the OrcidAuthentication:

- Exchanges with ORCID the **authorization code** for an access token. Along with the access token, the Orcid API also provides the ORCID iD and the name of the logged in user, the refresh token, the expiration timestamp in milliseconds (generally 20 years after issue) and the scopes related to the accepted grants.
- Check if there is already an EPerson in the system that has the **netId** equals to the ORCID iD.
    - if present it considers this user as the logged in user
    - otherwise, obtain the profile information from the ORCID register and check if there is an EPerson with the same email as the Person present on ORCID. If this search is also unsuccessful then a new EPerson is created, with netId equal to the ORCID id obtained.

The creation of a new EPerson starting from a login with ORCID is possible only if the property **authentication-orcid.can-self-register** is set to true (default), otherwise the user is considered not authenticated.

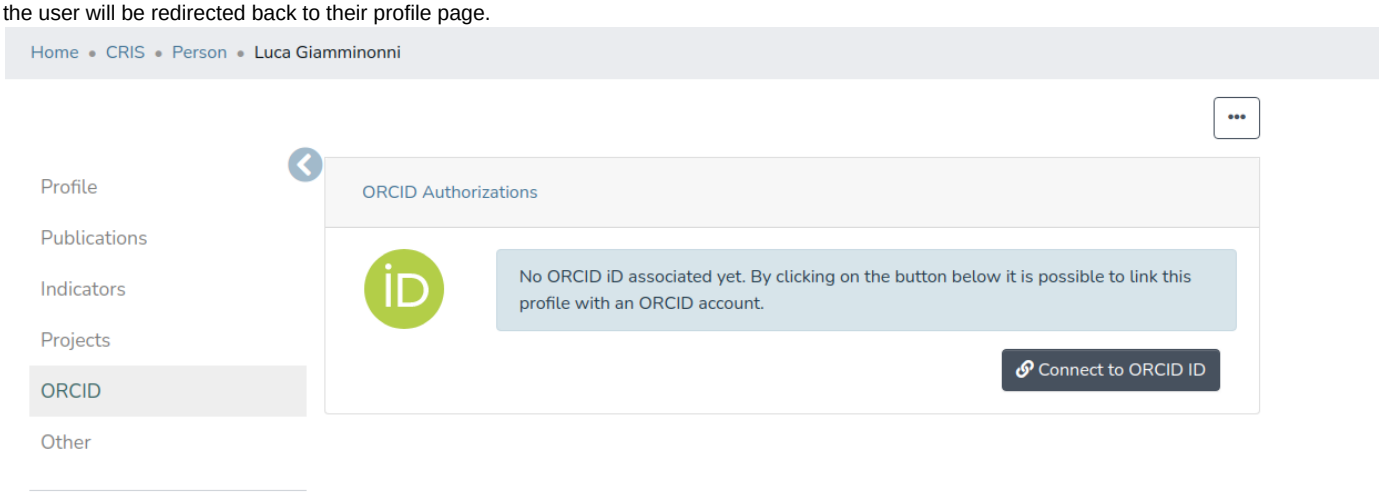
With each new login via ORCID, the ePerson identified (or created from scratch) is updated with the following metadata, populated by the data obtained from the authorization-code:

- **eperson.orcid** the ORCID iD of the logged user
- **eperson.orcid.access-token** the access token retrieved from ORCID
- **eperson.orcid.refresh-token** the refresh token retrieved from ORCID
- **eperson.orcid.scope** the list of scopes accepted by the user

Once the authentication flow is completed, the endpoint contacted with the redirect ( implemented by the class org.dspace.app.rest.**OrcidAuthenticationRestController**) takes care of redirecting the user back to the DSpace-CRIS home page. At this point the user is logged into the system and can browse it as if he had logged in normally.

#### Profile connection

It is also possible to integrate existing profiles on DSpace-CRIS with ORCID. To do this there is a specific box on the profile page under the "**ORCID**" tab: if the profile has not already been linked to an account on the ORCID register, the **ORCID authentication** box contains a button that will redirect the user to the ORCID login page and, once the authentication has been completed and the requested authorizations have been granted, the user will be redirected back to their profile page.




The mechanism used for authentication is the same that is used for login via ORCID, but in this case the redirect made after login on the ORCID register is towards the endpoint **/api/cris/orcid/{item-id}** , where the item-id is the id of the item related to the user's profile. This endpoint is implemented through the org.dspace.app.rest.**OrcidRestController** class and perform the following actions:

- Exchanges with ORCID the **authorization code** for an access token.
- Set the following metadata on the profile's item with the given id:

- **person.identifier.orcid** the ORCID iD
- **cris.orcid.access-token** the access token
- **cris.orcid.refresh-token** the refresh token
- **cris.orcid.scope** the granted scopes

#### EPerson and Profile ORCID metadata

During the login process via orcid, the information obtained after the authentication-code exchange with ORCID is stored in the metadata of the Eperson. This metadata will then be copied to the user's profile item when the profile is created. If, on the other hand, the user connects the profile to ORCID after having already logged into DSpace-CRIS (therefore for example with an existing user in DSpace) then this information will be saved directly in the profile item. For all functions concerning ORCID, the metadata taken into consideration will be only those of the profile.

### ORCID Synchronization

The synchronization process with ORCID allows to update the user publications, fundings and profile on ORCID after changes on DSpace CRIS items. To perform this synchronization, however, the user must:

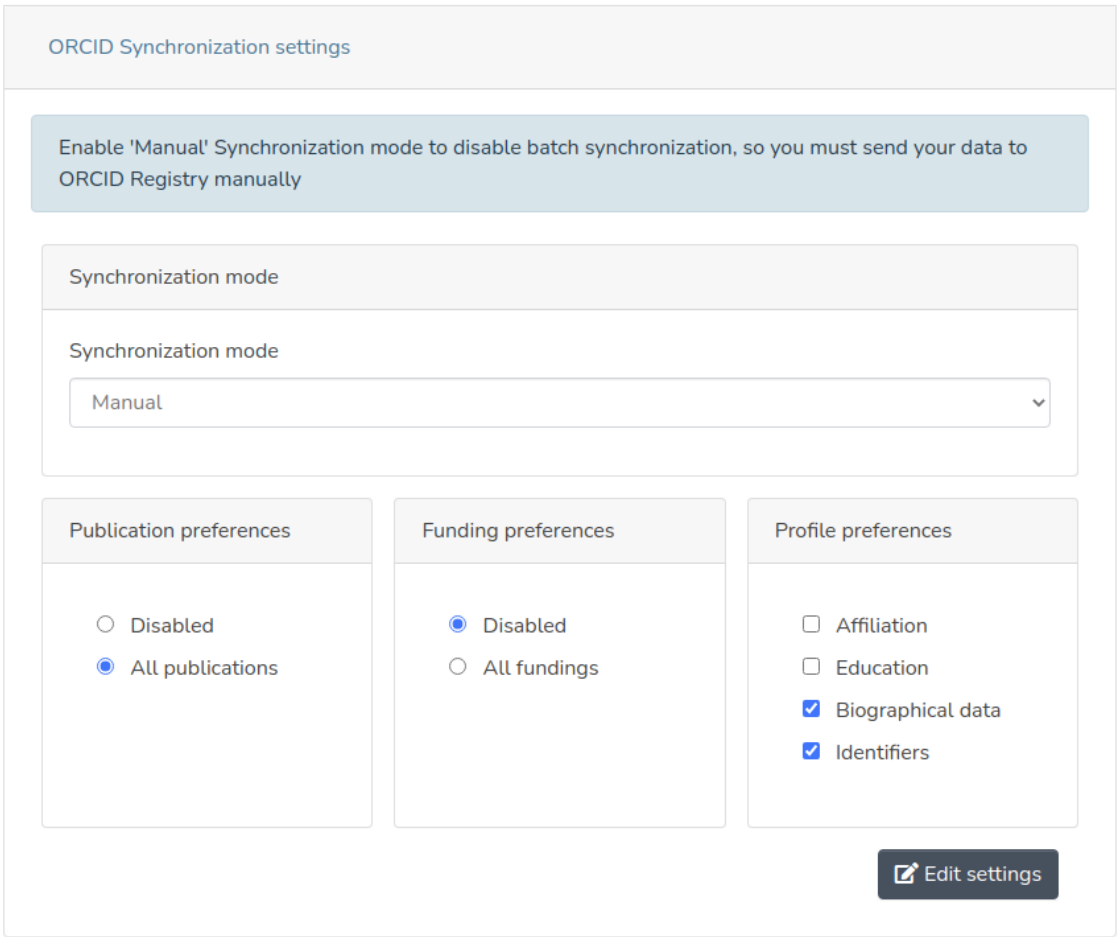
- has registered on DSpace CRIS via ORCID
- or has already connected his researcher profile with his ORCID account through the ORCID section present in the user's profile in DSpace CRIS. With this process the user from DSpace will be redirected to the ORCID website to log in and later grant the specific permission asked for.

A profile that can be synchronized with ORCID must therefore have the following metadata:

- **person.identifier.orcid:** the orcid id of the account on ORCID related to the researcher profile
- **cris.orcid.access-token**: the access token provided by the ORCID token url
- **cris.orcid.refresh-token:** the refresh token
- **cris.orcid.scope:** the scopes related to the permissions that the user has granted

#### ORCID Synchronization settings Box

It is possible to view and edit the synchronization modes and settings using the specific box under the ORCID tab. Like the ORCID tab itself and the other boxes in it, this section of the item can only be viewed by the owner of the item and only if an ORCID account was already linked to the viewing shown.




User specified synchronization settings are stored in the following metadata of the researcher profile item:

- **cris.orcid.sync-mode:** the synchronization mode
  - **MANUAL**: the synchronization on ORCID will be performed only when the user forces it manually
  - **BATCH**: the synchronization on ORCID will be done by a scheduled batch but can be also forces manually by the user
- **cris.orcid.sync-publications:** the configuration of the publications synchronization. Can have a single value among the following:
  - **DISABLED:** the synchronization of the publications is disabled
  - **ALL:** synchronize all the publications related to the profile
- **cris.orcid.sync-fundings:** the configuration of the fundigs synchronization. Can have a single value among the following:
  - **DISABLED:** the synchronization of the fundigs is disabled
  - **ALL:** synchronize all the fundigs related to the profile
- **cris.orcid.sync-profile:** the configuration of the profile synchronization. Can have many values among the following:
  - **AFFILIATION:** synchronize all the person's affiliations
  - **EDUCATION:** synchronize all the person's educations and qualifications
  - **IDENTIFIERS:** synchronize all the person's identifiers
  - **BIOGRAPHICAL:** synchronize all the person's biographical information (other names, country keywords etc ...)

The update of the synchronization preferences is done with a PATCH request to the endpoint **/api/cris/profiles/<:eperson-uuid>**, performing a REPLACE operation with one of the following paths:

- **/orcid/mode** - to update synchronization mode
- **/orcid/publications** - to update the preference relative to the publications synchronization
- **/orcid/projects** - to update the preference relative to the projects synchronization
- **/orcid/profile** - to update the preference relative to the profile synchronization; it is possible to specify multiple values using ',' as separator.

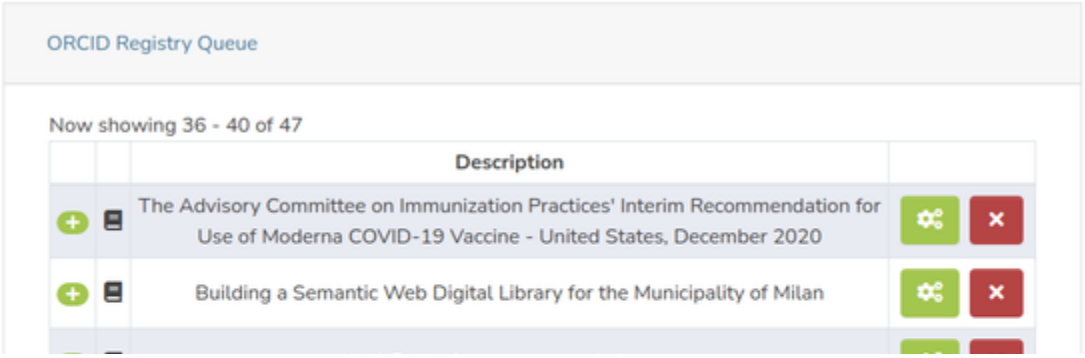
#### ORCID Registry Queue

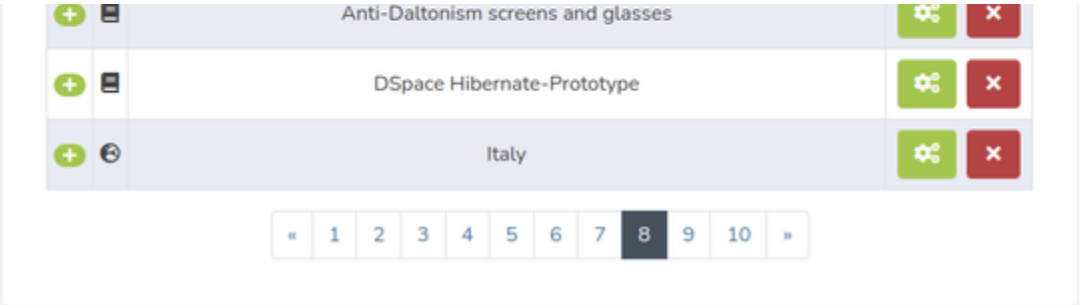
The items to be synchronized with ORCID, according to the synchronization configuration set by the user, are put in a **queue** of resources to be sent to ORCID to insert or update publications, fundings and profile informations. Once the user has forced sending to ORCID or the synchronization batch has done it, the items already synchronized will be removed from the queue.

The queue is modeled through the **orcid_queue** table, and each entry of that queue is therefore represented by a record of that table. Each record represents an item or a metadata to be synchronized on ORCID, with the reference to the item of the owner researcher profile. The columns of the **orcid_queue** table are:

- **id** id of the record
- **owner_id** the uuid of the profile item
- **entity_id** the uuid of the entity item to be synchronized; if the record refers to profile's sections, the entity_id is equals to the owner_id
- **description** the record description
- **record_type** the type of record. If the record refers to an entity distinct from the profile then the record_type represents the entity type of the item to be synchronized, otherwise it indicates the type of section of the profile to be synchronized. In the latter case, the possible values are:
  - AFFILIATION related to a nested metadata group of affiliation
  - EDUCATION related to a nested metadata group of education
  - QUALIFICATION related to a nested metadata group of qualification
  - OTHER_NAMES related to the profile's other name
  - COUNTRY related to the profile's country
  - KEYWORDS related to the profile's keywords
  - EXTERNAL_IDS related to the profile's external identifiers
  - RESEARCHER_URLS related to the profile's researcher urls
- **operation** the operation's type (INSERT, UPDATE, DELETE)
- **metadata** if the record refers to the synchronization of a section of the profile, that column the signature of the metadata referenced by the record; the signature is generated using a bean of type org.dspace.app.orcid.service.**MetadataSignatureGenerator** (see the dedicated section for further details)
- **put_code** in case of update or delete it indicates the entity to update/delete in the ORCID registry
- **attempts** number of push attempts made by the batch procedure

The records in the ORCID queue associated with a specific profile can be viewed in the specific box under the ORCID tab on the item page. From this section you can remove the records from the queue or force synchronization with the ORCID registry. For further details on manual synchronization, refer to the specific section of this page.






#### Orcid history

After each attempt to push data from DSpaceCRIS to the ORCID registry, a record is inserted in the **orcid_history** table with the detail of the response coming from ORCID. In particular, in addition to the columns present in the orcid_queue except the attempts column, the orcid_history also has the following columns:

- **response_message** the body of the response from ORCID
- **status** the status of the response from ORCID
- **timestamp_last_attempt** timestamp calculated during the push attempt

#### Orcid Queue population

The ORCID queue is in most cases populated by a specific consumer implemented by the class org.dspace.app.orcid.consumer.**OrcidQueueConsumer** which, when an archived item is modified, checks if it is necessary to update the queue of some profile connected to ORCID . In particular:

- if the modified item is a profile (Person item), check if it is linked to ORCID and if necessary recalculate the queue for each section of the profile. To do this, the metadata associated with the various sections of the profile are obtained, their signature is calculated and using that signature it is checked whether there have been any changes compared to what is present in the orcid_history.
- if the modified item is an entity of the supported type (Publication or Funding) then it is checked if among the item's metadata there are any related to a profile linked to ORCID. If a profile is identified, then a record is added to the queue, if not already present for the pair profile id/entity id. To understand if the operation associated with this new record must be an insertion rather than an update, a search in the orcid_history for an entry relating to the same entity for the same owner is done: if this record is present, the put_code present in the record of the orcid_history is set in the new queue record and the operation will be UPDATE, otherwise the operation will be INSERT.

The ORCID queue is also populated on two other occasions, in addition to the OrcidQueueConsumer:

- When a publication or funding associated with a profile linked to ORCID is deleted. In this case a DELETE record is inserted with entity id null and putCode the one taken from the orcid history.
- When publication/funding preferences are updated.

#### Publications synchronization

Publications can be synchronized with the ORCID registry to create/update Work entities. The conversion between the items representing the publications and the works in xml format to be sent to the ORCID register is managed by the bean org.dspace.app.orcid.model.factory.impl.**OrcidWorkFactory**.
This bean use a dynamic configuration of the metadata to be read, implemented through the class org.dspace.app.orcid.model.**OrcidWorkFieldMapping**.
The mapping between the publication's metadata fields and the Work attributes present in the ORCID registry can be configured through the following properties:

- **orcid.mapping.work.title** the work's title
- **orcid.mapping.work.sub-title** the work's sub-title
- **orcid.mapping.work.short-description** the work's description
- **orcid.mapping.work.publication-date** the work's publication date
- **orcid.mapping.work.language** the work's language
- **orcid.mapping.work.language.converter** the name of a bean of the class org.dspace.util.**SimpleMapConverter** that allows to map the publication's language stored in DSpace-CRIS to the languages supported by ORCID
- **orcid.mapping.work.journal-title** the work's journal title
- **orcid.mapping.work.type** the work's type
- **orcid.mapping.work.type.converter** the name of a bean of the class org.dspace.util.**SimpleMapConverter** that allows to map the publication's type stored in DSpace-CRIS to the types supported by ORCID
- **orcid.mapping.work.citation.type** one of the citation type defined in the keys of the map defined for the attribute of the OrcidWorkFieldMapping named **citationCrosswalks**
- **orcid.mapping.work.contributors** the list of metadata associated with the contributors of the publications in the format <metadatafield>::<role>, where role must take on one of the roles allowed by ORCID ( as author, editor, co-inventor etc..)
- **orcid.mapping.work.external-ids** the list of metadata associated with the external identifiers of the publications in the format <metadatafield>::<type> or $simple-handle::<type>, where
    - type is one of the available external identifiers (click here for more details)

    - $simple-handle indicates to use the item handle
- **orcid.mapping.contributor.email** the work contributors email
- **orcid.mapping.contributor.orcid** the work contributors orcid
- **orcid.mapping.work.funding** the funding related to the work
- **orcid.mapping.work.funding.external-id.type** one of the available funding external identifier type
- **orcid.mapping.work.funding.external-id.value** the funding external identifier present in the work
- **orcid.mapping.work.funding.external-id.entity-value** the funding external identifier taken from the funding entity
- **orcid.mapping.work.funding.url** the funding url taken from the funding entity

Below is an example of a configuration:

```
orcid.mapping.work.title = dc.title
orcid.mapping.work.sub-title =
orcid.mapping.work.short-description = dc.description.abstract
orcid.mapping.work.publication-date = dc.date.issued
orcid.mapping.work.language = dc.language.iso
orcid.mapping.work.language.converter =
mapConverterDSpaceToOrcidLanguageCode
orcid.mapping.work.journal-title = dc.relation.ispartof
orcid.mapping.work.type = dc.type
orcid.mapping.work.type.converter =
mapConverterDSpaceToOrcidPublicationType
orcid.mapping.work.citation.type = bibtex
orcid.mapping.work.contributors = dc.contributor.author::author
orcid.mapping.work.contributors = dc.contributor.editor::editor
orcid.mapping.work.external-ids = dc.identifier.doi::doi
orcid.mapping.work.external-ids = dc.identifier.scopus::eid
orcid.mapping.work.external-ids = dc.identifier.pmid::pmid
orcid.mapping.work.external-ids = $simple-handle::handle
orcid.mapping.work.external-ids = dc.identifier.isi::wosuid
orcid.mapping.work.external-ids = dc.identifier.issn::issn
orcid.mapping.work.funding = dc.relation.funding
orcid.mapping.work.funding.external-id.type = grant_number
orcid.mapping.work.funding.external-id.value = dc.relation.grantno
orcid.mapping.work.funding.external-id.entity-value = oairecerif.
funding.identifier
orcid.mapping.work.funding.url = crisfund.award.url
orcid.mapping.contributor.email = person.email
orcid.mapping.contributor.orcid = person.identifier.orcid
```

#### Fundings synchronization

Fundings can be synchronized with the ORCID registry to create/update Funding entities. The conversion between the items representing the fundings and the fundings in xml format to be sent to the ORCID register is managed by the bean org.dspace.app.orcid.model.factory.impl.**OrcidFundingFactory**.
This bean use a dynamic configuration of the metadata to be read, implemented through the class org.dspace.app.orcid.model.**OrcidFundingFieldMapping**.
The mapping between the funding's metadata fields and the Funding attributes present in the ORCID registry can be configured through the following properties:

- **orcid.mapping.funding.title** the funding's title
- **orcid.mapping.funding.type** the funding's type
- **orcid.mapping.funding.type.converter** the name of a bean of the class **SimpleMapConverter** that allows to map the funding's type stored in DSpace-CRIS to the types supported by ORCID
- **orcid.mapping.funding.external-ids** the list of metadata associated with the external identifiers of the fundings in the format <metadatafield>::<type>, where type is one of the available external identifiers (click here for more details).
- **orcid.mapping.funding.description** the funding's description

- **orcid.mapping.funding.start-date** the funding's start date
- **orcid.mapping.funding.end-date** the funding's end date
- **orcid.mapping.funding.contributors** the list of metadata associated with the contributors of the fundings in the format <metadatafield>::<role>, where role must take on one of the roles allowed by ORCID ( as lead, co-lead etc..)
- **orcid.mapping.funding.organization** the funding's funder. It is necessary that the metadata field is authority controlled and linked to an orgUnit. For more information on how to configure the sending of organization data, refer to the "**Organizations mapping**" section.
- **orcid.mapping.funding.amount** the funding's amount
- **orcid.mapping.funding.amount.currency** the funding's amount currency
- **orcid.mapping.funding.amount.currency.converter** the name of the **SimpleMapConverter** bean that allows to map the amount currency stored in DSpace-CRIS to the currencies supported by ORCID
- **orcid.mapping.contributor.email** the funding contributors email
- **orcid.mapping.contributor.orcid** the funding contributors orcid

Below is an example of a configuration:

```
orcid.mapping.funding.title = dc.title
orcid.mapping.funding.type = dc.type
orcid.mapping.funding.type.converter =
mapConverterDSpaceToOrcidFundingType
orcid.mapping.funding.external-ids = oairecerif.internalid::other-id
orcid.mapping.funding.external-ids = crisfund.award.url::uri
orcid.mapping.funding.external-ids = oairecerif.funding.identifier::
grant_number
orcid.mapping.funding.description = dc.description
orcid.mapping.funding.start-date = oairecerif.funding.startDate
orcid.mapping.funding.end-date = oairecerif.funding.endDate
orcid.mapping.funding.contributors = crisfund.investigators::lead
orcid.mapping.funding.contributors = crisfund.coinvestigators::co-lead
orcid.mapping.funding.organization = oairecerif.funder
orcid.mapping.funding.amount = oairecerif.amount
orcid.mapping.funding.amount.currency = oairecerif.amount.currency
orcid.mapping.funding.amount.currency.converter =
mapConverterDSpaceToOrcidAmountCurrency
orcid.mapping.contributor.email = person.email
orcid.mapping.contributor.orcid = person.identifier.orcid
```

#### Profile synchronization

Unlike synchronization of publications and fundings, only certain information can be synchronized for the profile, based on the synchronization preferences you set. The profile's metadata to be synchronized based on the preferences expressed can be configured using the following properties:

| ORCID data | Property | Preference | Multi-values |
|---|---|---|---|
| Other names (Also known as) | orcid.mapping.other-names | BIOGRAPHICAL | ☑ |
| Keywords | orcid.mapping.keywords | BIOGRAPHICAL | ☑ |
| Country | orcid.mapping.country | BIOGRAPHICAL | ☑ |
| Other ids | orcid.mapping.person-external-ids | IDENTIFIERS | ☑ |
| Websites & Social Links | orcid.mapping.researcher-urls | IDENTIFIERS | ☑ |
| Employment name | orcid.mapping.affiliation.name | AFFILIATION | |
| Employment role | orcid.mapping.affiliation.role | AFFILIATION | |
| Employment start date | orcid.mapping.affiliation.start-date | AFFILIATION | |
| Employment end date | orcid.mapping.affiliation.end-date | AFFILIATION | |

| | | | |
|---|---|---|---|
| Qualification name | orcid.mapping.qualification.name | EDUCATION | |
| Qualification role | orcid.mapping.qualification.role | EDUCATION | |
| Qualification start date | orcid.mapping.qualification.start-date | EDUCATION | |
| Qualification end date | orcid.mapping.qualification.end-date | EDUCATION | |
| Education name | orcid.mapping.education.name | EDUCATION | |
| Education role | orcid.mapping.education.role | EDUCATION | |
| Education start date | orcid.mapping.education.start-date | EDUCATION | |
| Education end date | orcid.mapping.education.end-date | EDUCATION | |

Below is an example of a configuration:

```
### Affiliation mapping ###
orcid.mapping.affiliation.name = oairecerif.person.affiliation
orcid.mapping.affiliation.role = oairecerif.affiliation.role
orcid.mapping.affiliation.start-date = oairecerif.affiliation.startDate
orcid.mapping.affiliation.end-date = oairecerif.affiliation.endDate
### Qualification mapping ###
orcid.mapping.qualification.name = crisrp.qualification
orcid.mapping.qualification.role = crisrp.qualification.role
orcid.mapping.qualification.start-date = crisrp.qualification.start
orcid.mapping.qualification.end-date = crisrp.qualification.end
### Education mapping ###
orcid.mapping.education.name = crisrp.education
orcid.mapping.education.role = crisrp.education.role
orcid.mapping.education.start-date = crisrp.education.start
orcid.mapping.education.end-date = crisrp.education.end
### Other names mapping ###
orcid.mapping.other-names = crisrp.name.variant
orcid.mapping.other-names = crisrp.name.translated
### Keywords mapping ###
orcid.mapping.keywords = dc.subject
### Country mapping ###
orcid.mapping.country = crisrp.country
orcid.mapping.country.converter =
### Person External ids mapping ###
##orcid.mapping.person-external-ids syntax is <metadatafield>::<type>
orcid.mapping.person-external-ids = person.identifier.scopus-author-id::
SCOPUS
orcid.mapping.person-external-ids = person.identifier.rid::RID
### Researcher urls mapping ###
orcid.mapping.researcher-urls = oairecerif.identifier.url
```

For affiliation, education and qualification, the ORCID register also requires specific information related to the organizations associated with these activities. It is therefore necessary that the 3 metadata indicated for the name are authority controlled and linked to an orgUnit. For more information on how to configure the sending of organization data, refer to the "**Organizations mapping**" section.

**Organizations mapping**

Some information related to organizations, such as the funder of fundings or organizations related to the affiliation of a profile, require some details on the organizations themselves that in DSpace-CRIS must be obtained from the OrgUnit entities. The mapping between the data that make up an organization on the ORCID registry and the metadata of the organizations on the CRIS side is managed by the following configuration properties:

- **orcid.mapping.organization.country** the organization's country
- **orcid.mapping.organization.city** the organization's city
- **orcid.mapping.organization.identifiers** the organization's identifiers with the syntax <metadatafield>::<**source**>, where source must be one of the identifiers that ORCID supports, listed by the property **orcid.validation.organization.identifier-sources**

Below is an example of a configuration:

```
orcid.mapping.organization.country = organization.address.addressCountry
orcid.mapping.organization.city = organization.address.addressLocality
orcid.mapping.organization.identifiers = organization.identifier.
crossrefid::FUNDREF
orcid.mapping.organization.identifiers = organization.identifier.rin::
RINGGOLD
```

#### Metadata signature

When a record relating to a section of the profile is added to the orcid queue, the metadata column is populated by generating a signature of the metadata values that are associated with the particular data to be synchronized. The use of a signature, not linked to the id of the metadata values, allows not to interpret as new data to be synchronized those metadata that have the same value but a different id. The class that is used to generate the signature is org.dspace.app.orcid.service.impl.**PlainMetadataSignatureGeneratorImpl**, which generates signatures with the **<metadatafield>::<value>[::<authority]** format, where the authority section is set only if the metadata authority is not empty. If the signature to be generated relates to more than one metadata (such as the signature of the nested metadata that make up the affiliation), the signature described above is generated for each metadata and all the individual signatures are sorted based on the id of the metadata field and concatenated with the **§§** characters.

Examples:

- the signature of the metadata dc.title with value "Publication title" is "dc.title::Publciation title"
- the signature of the metadata dc.contributor.author with value "John Smith" and authority XXX and of the metadata oairecerif.author.affiliation with value "4Science" is "dc.contributor.author::John Smith::XXX§§oairecerif.author.affiliation::4Science"

#### Manual synchronization

To send an ORCID queue entry to the ORCID api and create a record into the ORCID history a reference of an ORCID queue record must be posted to the **ORCID history resource endpoint (/api/cris/orcidhistories)**. The ORCID queue record must be supplied as URI in the request body using the text/uri-list content-type.
This endpoint, once invoked, will then send the entity associated with the specified orcid_queue, will create a new record on the orcid_history with the result of the send and will return it to the caller.

An optional query param named **forceAddition** with value true or false could be provided to force the send of a new resource to the ORCID api without an update of an existing resource even if for the provided ORCID queue record there is a put-code. This parameter allows for example to force the insertion of a new object on ORCID even if it had already been sent and then be deleted on ORCID in the past.

In case of insertion or updating, the data to be sent to the ORCID registry are validated by the org.dspace.app.orcid.model.validator.impl.**OrcidValidatorImpl** class. In case of validation errors, such as the absence of a mandatory attribute, the endpoint returns a response with status 422 and body containing the error codes. This allows clearer error messages to be returned to the user.
It is possible to disable the validation of work, funding and affiliation (employment, education and qualification) through the following properties (all enabled by default):

- **orcid.validation.work.enabled**
- **orcid.validation.funding.enabled**
- **orcid.validation.affiliation.enabled**

If the synchronization was successful, the record of the orcid queue is deleted.

#### Synchronization Batch

The script called **orcid-bulk-push** and implemented by the org.dspace.app.orcid.script.**OrcidBulkPush** class allows to massively synchronize all the profiles that have configured their synchronization preferences to **BATCH.** The steps of the batch process are:

- identifies all the records of the ORCID queue to be synchronized by checking if the owner has configured the BATCH mode
- filters the record that exceed the maximum number of configured attempts (configured with the property **orcid.bulk-synchronization.max-attempts**)
- perform the synchronization with ORCID
- in case of error it increases the number of push attempts (in case of successful synchronization this is not necessary because the record is removed from the queue)

The script accept the following options:

- force (f)  force the synchronization ignoring maximum attempts

### ORCID Registry Lookup

For metadata controlled by an authority related to Person item, it is possible to configure the org.dspace.content.authority.**OrcidAuthority** which allows, in addition to searching for matches between persons stored in DSpace-CRIS, also to search for profiles on the ORCID register. The OrcidAuthority is therefore a class that extends the org.dspace.content.authority.**ItemAuthority** and appends the profiles identified on the ORCID register to the results identified by the ItemAuthority.

To set which metadata should be associated with this authority it is necessary to modify the file **authority.cfg**, as for the other authorities. By default, all the authorities linked to entities of type Person are configured with the OrcidAuthority.


Orcid expanded search

To identify the profiles on the ORCID register that match a specific string (the name of the person linked to the profile), the OrcidAuthority uses the **expanded search endpoint** made available by ORCID. For more information about the expanded search click here.

Given an input “XYZ”, the query sent to the search endpoint is the following:

```
given-names:XYZ+OR+family-name:XYZ+OR+other-names:XYZ
```

If the input name has spaces or commas then the query shown above is duplicated for each section of the name, concatenating them with AND. For example, a search for "Smith, John" produces the following query:

```
(given-names:Smith+OR+family-name:Smith+OR+other-names:Smith) AND
(given-names:John+OR+family-name:John+OR+other-names:John)
```

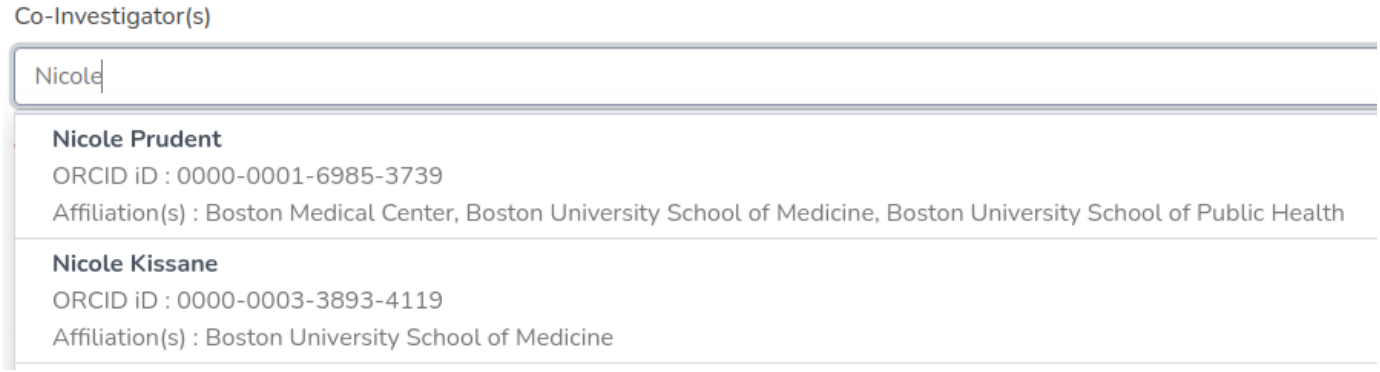
The request that is made is paged on the basis of the pagination set and taking into account the number of results resulting from the research of the ItemAuthority.

If the ORCID API keys are configured then the search is performed on the API endpoint, first obtaining a read public access-token; if instead the API keys are not configured then the public endpoint is contacted.

Authority and extra informations

For each profile identified by the search described above, a choice is constructed with the following attributes:

- **authority** the ORCID iD of the profile with the prefix configured through the xxx property.
  Example: will be referenced::ORCID::0000-0000-0123-4567
- **label** the given and family name of the profile
- **value** the given and family name of the profile
- **extras** the ORCID iD (data-person_identifier_orcid) and, if present, the list of all the institutions linked to the profile (institution-affiliation-name).



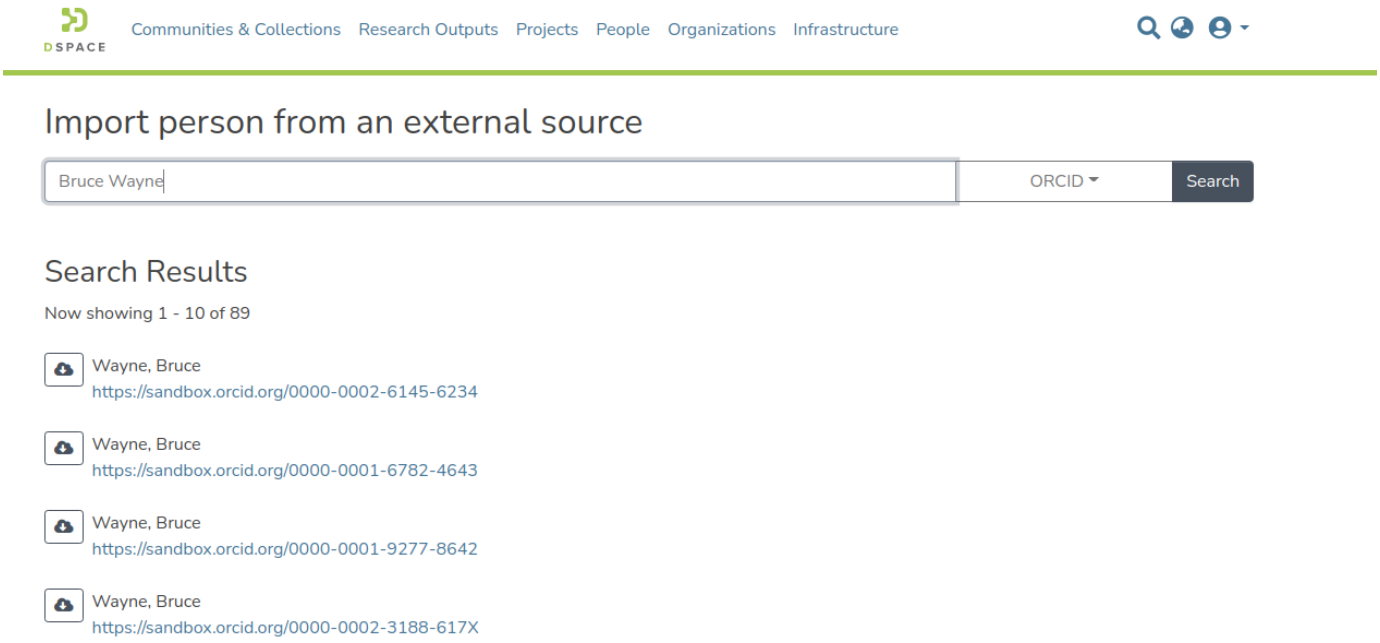

### ORCID Imports

Among the external data providers with which it is possible to import data from external sources, two providers are defined to import profiles and publications from the ORCID registry.

#### Author data provider

The import of the profiles from the ORCID registry is managed by the bean of the class org.dspace.external.provider.impl.**OrcidV3AuthorDataProvider:**

- to search by a query, the **/search** endpoint made available by the ORCID API is used
- to search by id, the **/person** endpoint is used looking for the provided ORCID iD




The configuration of the org.dspace.external.provider.impl.OrcidV3AuthorDataProvider bean present in the /config/spring/api/**external-services.xml** file is as follows:

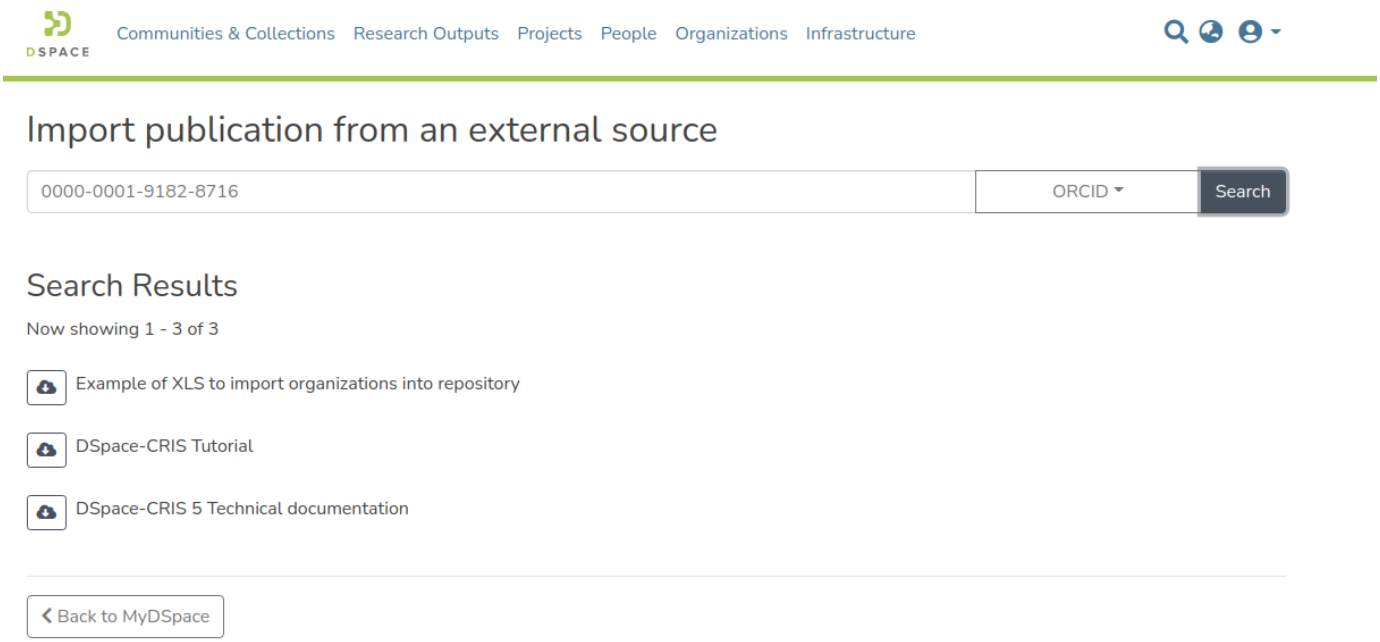
```
<bean class="org.dspace.external.provider.impl.
OrcidV3AuthorDataProvider" init-method="init">
  <property name="sourceIdentifier" value="orcid"/>
  <property name="orcidUrl" value="${orcid.domain-url}" />
  <property name="clientId" value="${orcid.application-client-id}" />
```

```
  <property name="clientSecret" value="${orcid.application-client-
secret}" />
  <property name="OAUTHUrl" value="${orcid.token-url}" />
  <property name="orcidRestConnector" ref="orcidRestConnector"/>
  <property name="supportedEntityTypes">
    <list>
        <value>Person</value>
    </list>
  </property>
</bean>
```

Publication data provider

The import of the publication of a specific profile from the ORCID registry is managed by the bean of the class org.dspace.external.provider.impl.**OrcidPublicationDataProvider:**

- to search for all the publications of a user given his ORCID id, the endpoint **/works** is used and, once all the publications of that profile have been obtained, those with the same source as DSpace-CRIS are discarded. To distinguish which works have DSpace-CRIS itself as source, it is used the client-id contained in the source attribute of the works obtained, comparing them with the configured client-id.
- to search by a single work, the **/work** endpoint is used searching by ORCID id and putCode




The configuration of the org.dspace.external.provider.impl.OrcidPublicationDataProvider bean present in the /config/spring/api/**external-services.xml** file is as follows:

```
<bean id="orcidPublicationDataProvider" class="org.dspace.external.
provider.impl.OrcidPublicationDataProvider">
  <property name="sourceIdentifier" value="orcidWorks"/>
  <property name="fieldMapping" ref="
orcidPublicationDataProviderFieldMapping"/>
  <property name="supportedEntityTypes">
    <list>
      <value>Publication</value>
```

```
    </list>
  </property>
</bean>
```

### ORCID Webhook

ORCID provides a notification webhook that allows premium members to stay up-to-date on new information, or even trigger events in their own systems based on an activity (for more details click here).

DSpace-CRIS allows to use this service by registering a callback for each new profile with ORCID iD set and, eventually, with an account connected to ORCID. Registration to the callback is done by specifying a specific REST endpoint, implemented by the **webhook** method of the org.dspace.app.rest.**OrcidRestController** class. Specifically, the url is **/api/cris/orcid/{orcid-id}/webhook/{registration-token}**, where:

- **orcid-id** is the ORCID iD of the profile related to the event
- **registration-token** is an UUID provided during the callback registration with the value of the configuration property **orcid.webhook.registration-token**. The use of this token allows, when receiving a call at this endpoint, to check if the token obtained is the same as that provided during registration and therefore to ignore other calls.

#### OrcidWebhookConsumer

The consumer implemented by the class org.dspace.app.orcid.webhook.**OrcidWebhookConsumer** takes care of registering the webhook for profiles that have configured an ORCID iD (person.identifier.orcid).

The **orcid.webhook.registration-mode** property defines under which conditions registration must be performed; the allowed values are:

- **disabled** the registration of the webhook is disabled
- **only_linked** the registration is done only for the profiles that has an ORCID iD and an access token
- **all** the registration is done for all the profiles with an ORCID id set

After registration, the metadata **cris.orcid.webhook** is added to the profile with a value equal to the date on which the registration was made.

The unregistration is instead carried out by the endpoint itself which is contacted upon receipt of a callback if no profiles are found for the ORCID iD provided or if the profile identified with that id no longer has a valid token and registration to the webhook is configured to be only_linked. The unregister is also done if the user disconnects its profile from ORCID. After the unregistration the metadata cris.orcid.webhook is deleted.

#### Webhook actions

Upon receipt of a callback associated with a specific ORCID iD, the endpoint specified in the registrations looks for the profiles associated with this ORCID and, if present, invokes on them all the actions registered in the context through the bean of the class org.dspace.app.orcid.webhook.**OrcidWebhookAction.** Current implementations of this interface are:

- org.dspace.app.orcid.webhook.**CheckOrcidAuthorization** verify that any access token associated with the profile is still valid, using it to obtain information about the person from the ORCID registry. In case of authentication errors, the token is deleted.
- org.dspace.app.orcid.webhook.**RetrieveOrcidPublicationAction** using the Solr suggestion provider implemented by org.dspace.app.suggestion.orcid.**OrcidPublicationLoader** takes care of obtaining all publications associated with the profile that do not have DSpace-CRIS as a source and, for each of them, creates a new suggestion. This loader to get the publications uses the same external data provider described in the ORCID imports section and creates the suggestions in the specific Sorl core.

#### OrcidBulkPull script

To simulates the Webhook flow for each profile present in the system with an ORCID iD it is possible to use the script named **orcid-bulk-pull** and implemented by org.dspace.app.orcid.script.**OrcidBulkPull.**

The script has the following options:

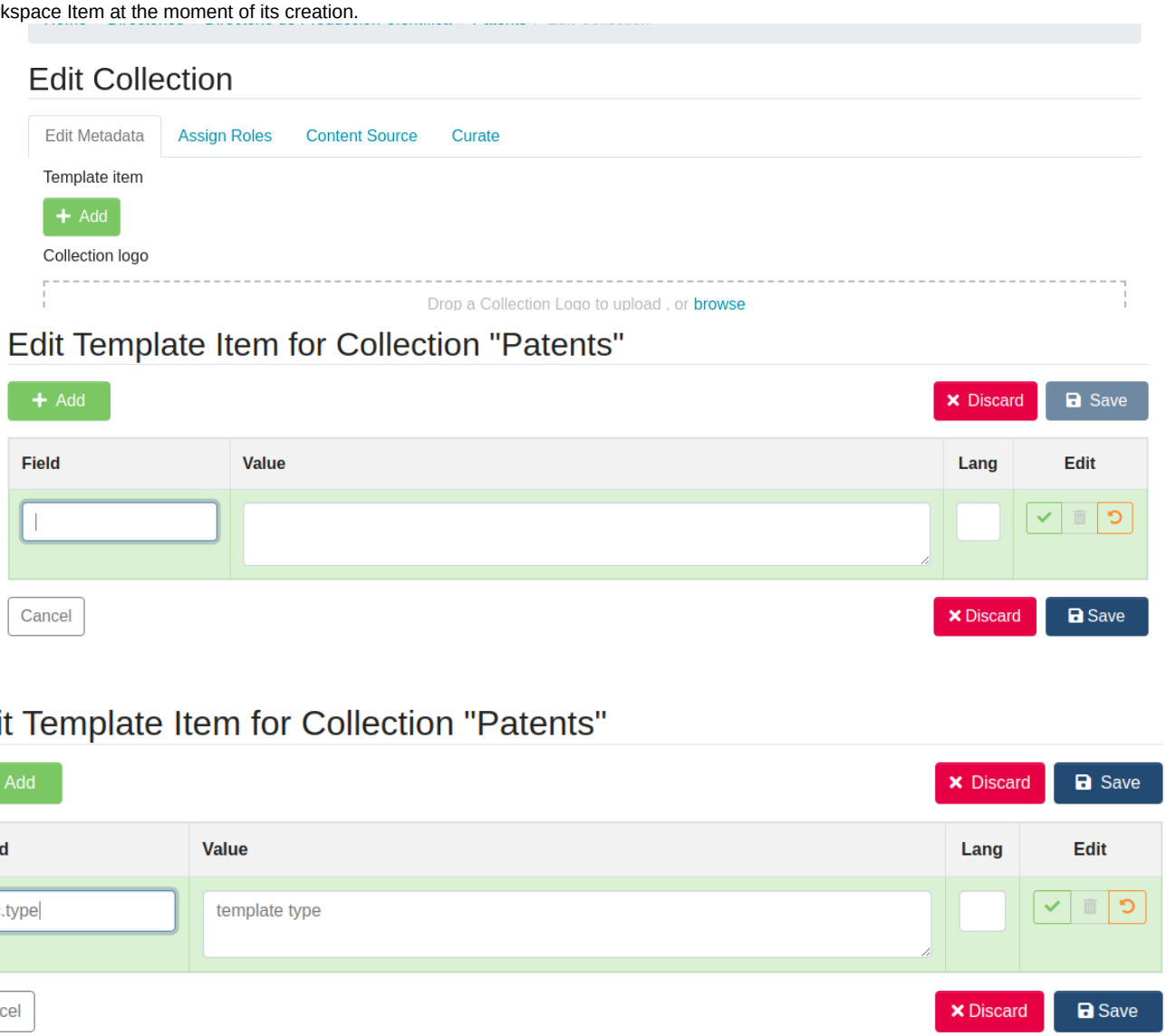
- linked (l)  to search only for profiles linked to ORCID (with ORCID iD and access-token)

Once the profiles have been identified with an ORCID id set, the script performs all the webhook actions defined on each of them, as is done by the endpoint that receives the callback from the ORCID registry.

## Create / import content

### Item Template

From the collection edit page, it is possible to define an Item Template. An Item template is a set of metadata that are automatically generated on the Workspace Item at the moment of its creation.







The inserted value can be of two kinds:

- Static value: it is a value that is copied into the new item metadata set as it is
- Dynamic value: a value to be generated at item creation moment by a specific generator. Generator is applied based on item template value syntax

#### Dynamic values

This is the list of current dynamic generators:

### *Date Generator*

If template item value is expressed as `###DATE.<date-pattern>###` , the metadata field will be set with item creation date expressed with the pattern indicated after the . (dot) character. For example a syntax '`###DATE.yyyy-MM-dd###`' used for a Workspace item created on 17th of December of 2020 will set a metadata field on created item with value '2020-12-17'

**Current user generator**

Generate metadata value with the name of the user that is creating a new workspace item. The template syntax is `###CURRENTUSER###` . If the metadata configured with this type of template is authority controlled, the authority will be set with the ePerson id associated with the user.

**Submitter's metadata generator**

Similar to the previous one allow to configure which metadata (i.e. email, fullname, phone), in the submitter eperson record is used to populate the value in the created item. The template syntax is `###SUBMITTER.<metadata-to-use>###`

## Bulk Import

The bulk import script allows to **add, update or delete multiple items** by uploading an **excel** file with a specific structure. All the items involved in a single import run must refer to a single **collection**: the updated and deleted objects must therefore be contained in that collection and the added items will be placed within the same collection.

The import can be started only by an admin or by one of the admins of the specified collection.

#### Script options

Bulk import is configured as a standard script and can be started either via CLI or via Rest. The options that can be provided when scheduling the script are the following:

- **collection** (c): the own collection uuid of the imported items (mandatory)
- **file** (f): the path of the excel source file from which to read the items to import (mandatory)
- **concludeOnError** (e): to conclude the import at the first error (default is false)

#### Excel structure

The excel to be imported must have a first sheet that represents the list of items to add/update/delete and many other sheets that contain groups of metadata that refer to the items contained in the first page.
The first page of the excel file must be compliant with the following rules:

- the sheet can have any name
- the first row represents the page header
- the first column, which must be named **ID**, represents the identifier of the item on which the operation should be performed. The syntax for specifying the id is as follows:
  - it can be the **uuid** of the item to be modified / deleted
  - it can have a value as **<type>::<value>**, where <type> represent the type of the id and <value> its value. Allowed values for <type> are the values configured in the **ItemSearcherMapper** described here.
    For example, the id ORCID::0000-0002-1825-0097 identifies the item that has the value 0000-0002-1825-0097 for the metadata person.identifier.orcid. The item identified in this way must be unique: if more items are identified for the same id, an error occurs.
- the second column, which must be named **ACTION**, represents the operation to be performed. The allowed values are the following:
  - **ADD**: create a workspace item and, if valid, start the workflow for it; in this case the ID column must be empty
  - **ADD_WORKSPACE**: create an workspace item without try to start the workflow for it; in this case the ID column must be empty
  - **ADD_ARCHIVE:** create an workspace item and archive it directly; only the admin users can use this action. In this case the ID column must be empty
  - **UPDATE**: update the item; in this case the ID column must not be empty
  - **UPDATE_WORKFLOW**: update the item and, if it is still in workspace and it is valid, start the workflow for it; in this case the ID column must not be empty
  - **UPDATE_ARCHIVE**: update the item and, if it is not already archived, archive it; only the admin users can use this action. In this case the ID column must not be empty
  - **DELETE**: update the item; in this case the ID column must not be empty
  - **no value set**: in this case if the ID is set and an item exists for that identifier that item is updated, otherwise a new item is created
- the remaining columns represent the **metadata** to be set for the item; the headers of these columns must be the metadata fields and must be conform to the **submission's configuration** related to the collection in which you are importing ( **submission-forms.xml** ). The metadata associated with groups should not be listed on the first page but on the following pages, as will be explained later.

        - You can specify the **language** of the metadata with the syntax **<metadata-field>[<language-code>]**.
          For example, to specify both value without language and the english value for the dc.title metadata the excel must contains the columns **dc.title** and **dc.title[en].**
    - The values of the metadata columns must be the metadata values to be set for the item being created/updated. In case of update, all the metadata present are replaced with the specified values.
        - It is possible specify **multiple values** for a single metadata field by concatenating the values with **||**
        - it is possible to specify an **authority** and the **confidence** related to a metadata value with the syntax **<value>$$<authority>$$<confidence>.** If the confidence is not provided, 600 is set.

Example of main sheet:

| ID | ACTION | dc.title | dc.title[en] | dc.date.issued |
|---|---|---|---|---|
| | ADD | title \|\| title 2 | title en | 01/01/2020 |
| 0b3339a4-da1c-467a-a52e-979bf78eb0ae | UPDATE | title3$$auth | | 02/01/2020 |
| DOI::10.1000/182 | DELETE | | | |

Importing an excel with the previous content would produce the following operations (if the listed metadata are valid for the specified collection):

- adding a new item with two dc.title “title” and “title2” without language, with one dc.title “title en” with “en” language and a dc.date.issued with value “01/01/2020”
- updating the item with UUID equals to “0b3339a4-da1c-467a-a52e-979bf78eb0ae” replacing the dc.title without language with a metadata with value “title3” and authority “auth”, the dc.title for the language “en” with an empty value and the dc.date.issued with “02/01/2020”
- deleting the item with the DOI metadata equals to “10.1000/182”

The sheets following the first are used to specify additional sets of metadata to add to those specified on the first page. Any number of this type of sheet can be present in the excel to be imported; all these sheets must comply with the following rules:

- their name must match the name of a metadata that represents a group of metadata configured for the submission ( **submission-forms.xml** ).
- the first row represents the page header
- the first column, which must be named **PARENT-ID**, represents the identifier of the item to add the metadata group to. This ID must therefore correspond to that of an item on the first page, using the same syntax explained above. In addition, the reference to a specific item on the first page can also be indicated with the syntax **ROW-ID::<item-row>,** where <item-row> represents the row’s index of the item to be referenced (the row count starts from 1 and includes the header).
- the remaining columns represent the **metadata** to be set for the item; the headers of these columns must be the metadata fields and must be conform to the **submission’s configuration** related to the collection in which you are importing ( **submission-forms.xml** )
- The values of the metadata columns must be the metadata group values to be set for the item being created/updated. In case of update, all the metadata present are replaced with the specified values.
    - It is not possible specify multiple values in a single cell; to provided more values for the same metadata it is necessary to insert more rows referring them through the PARENT-ID to the same item present in the first sheet.
    - it is possible to specify an **authority** and the **confidence** related to a metadata value with the syntax **<value>$$<authority>$$<confidence>.** If the confidence is not provided, 600 is set.
- rows that do not refer to any row on the main page will be ignored. Therefore, even if you want to add an item with metadata groups only, it is still necessary to add a row in the first sheet and then refer to this row in the following sheets.

#### Reference objects by their business identifiers

Using the authority it is possible to refer to other items to create a link between them. To do this, the authority of a metadata can be set, for example, with the uuid of the item to which this metadata is related. Furthermore the authority can have the format **to be generated::<type>::<value>** or **to be referenced::<type>::<value>**, where the pair <type>::<value> follows the same rules of values insertable in the ID column:

- **type** can have one of the keys defined in the ItemSearcherMapper map
- **value** represents the string to search for

Through the use of these syntax it is possible to instruct the CrisConsumer to relate items correctly. The difference between this two prefixes is that with “will be generated” a new item will be created if not found, while with “will be referenced” no. The behaviour, on the other hand, is identical in the case in which the related item is found.

Therefore, to summarize the different syntaxes by which, for example, a publication can be linked with a person:

**Federico Garcia$$a469026b-af3e-4f53-8781-54242a709e48** --> will link the publication with the person having uuid a469026b-af3e-4f53-8781-54242a709e48, if it already exists; otherwise it won’t create any link

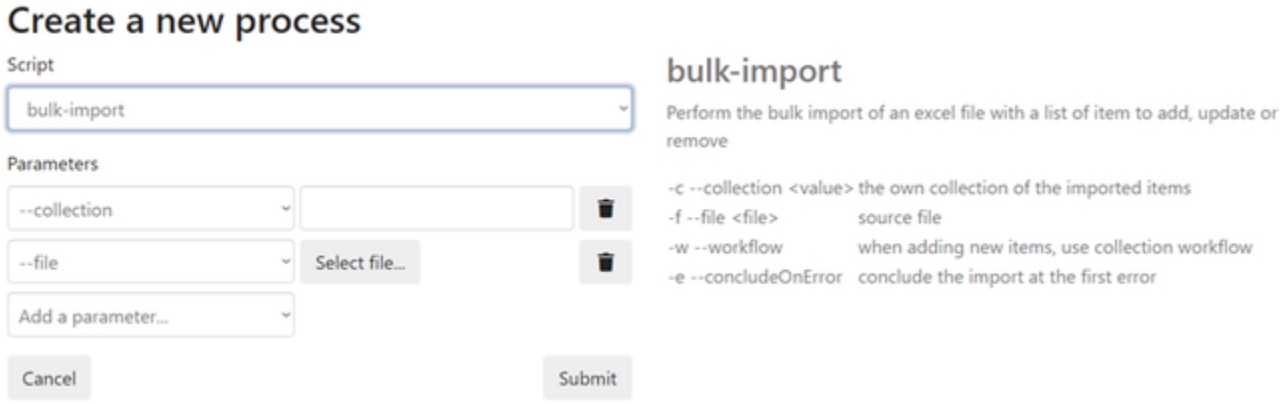
**Fernando Garcia$$will be referenced::ORCID::0000-0002-1825-0097** --> will link the publication with the person having ORCID 0000-0002-1825-0097 if it already exists, or will create the link once that person will be created;

**Fernando Garcia$$will be generated::ORCID::0000-0002-1825-0097 -->** will link the publication with the person having ORCID 0000-0002-1825-0097 if it already exists,; otherwise it will create the person with those ORCID and link it with the publication.

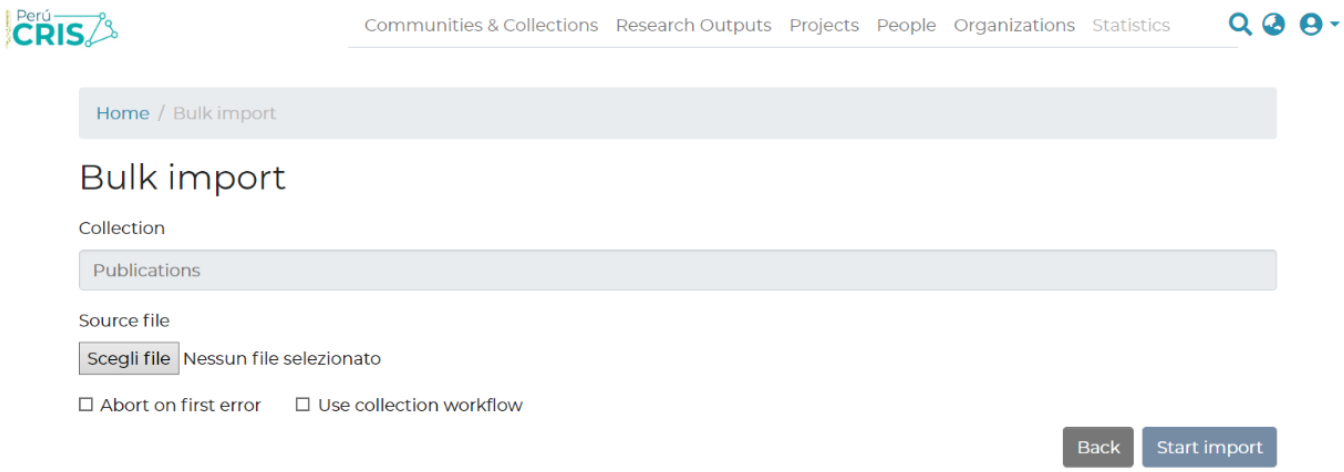
#### Start the script from GUI

The script can be started from Angular in two ways:

- from the processes page from which all allowed scripts can be started




- from the page dedicated to this import under **/bulk-import/<collection-id>.** This page can be accessed via a new button located at the top right of the collection page. This button is visible only to the admin or to the admin of the collection shown on the page. These users are therefore the only ones who can access the page to start the script. If other users try to access the /bulk-import/<collection-id> page directly they get an unauthorized error




#### Collection's items export

To facilitate the updating of the items of a collection via bulk import, there is also an export mode for a specific collection that allows to download the list of items in an xls file having the same format as required by the bulk import (the only difference is the absence of the ACTION column on the main page).

The process that allow to download the items of a single collection in xls format is called **collection-export** and requires the uuid of the collection as option named **-c**. Like the bulk import, this process can also be started either from the processes page or through a specific button on the collection page. Only the admin of the collection can start the download of the collection in the format required by the bulk import.

#### Reference between items

To create references between items through their metadata, a value with the prefix will be referenced can be set as authority. For more information refer to Item reference resolution

#### Pre transformation

It is possible to handle and modify the value contained in xls file used for the bulk import and the actual metadata value and authority. This functionality can be used, for example, on data coming from controlled vocabularies or lists in order to allow the submitter to report into the xls file the key instead of the full value.

To reach this goal, an interface is exposed, `org.dspace.app.bulkedit.BulkImportValueTransformer`, and must be implemented to define a custom value transformation.

Once implemented, should be configured in spring configuration `dspace/config/spring/api/bulk-import-value-transformer-service.xml` file, within constructor argument list of `BulkImportTransformerService` and associated to metadata which will transform during the bulk import process. With this interface implementations it is possible to drive logic that sets a metadata value and its authority.

```
<bean class="org.dspace.app.bulkedit.BulkImportTransformerService">
        <constructor-arg>
                        <map>
                        <!-- <entry  key="" value-ref= ""> </entry> --
>
                        </map>
                </constructor-arg>
</bean>
```

In following example, a metadata called “dc.example” will be transformed during bulk import by logic contained into ExampleValueTransformer implementation. Please note, following is just an example, at the moment no default implementations of `BulkImportTransformerService` are part of DSpace-CRIS7 source code.

```
<bean class="org.dspace.app.bulkedit.BulkImportTransformerService">
        <constructor-arg>
                        <map>
                         <entry  key="dc.example" value-ref=
"exampleTransformer"> </entry>
                        </map>
                </constructor-arg>
        </bean>

    <bean id="exampleTransformer" class="org.dspace.app.bulkedit.
ExampleValueTransformer"/>
```

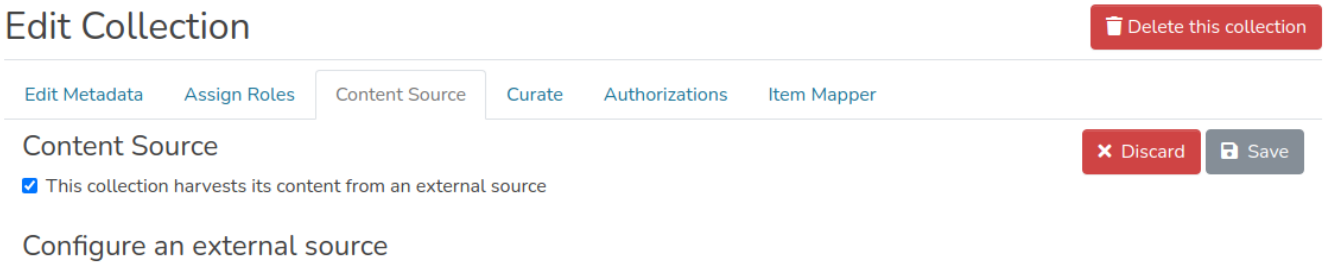
Import via OAI-PMH

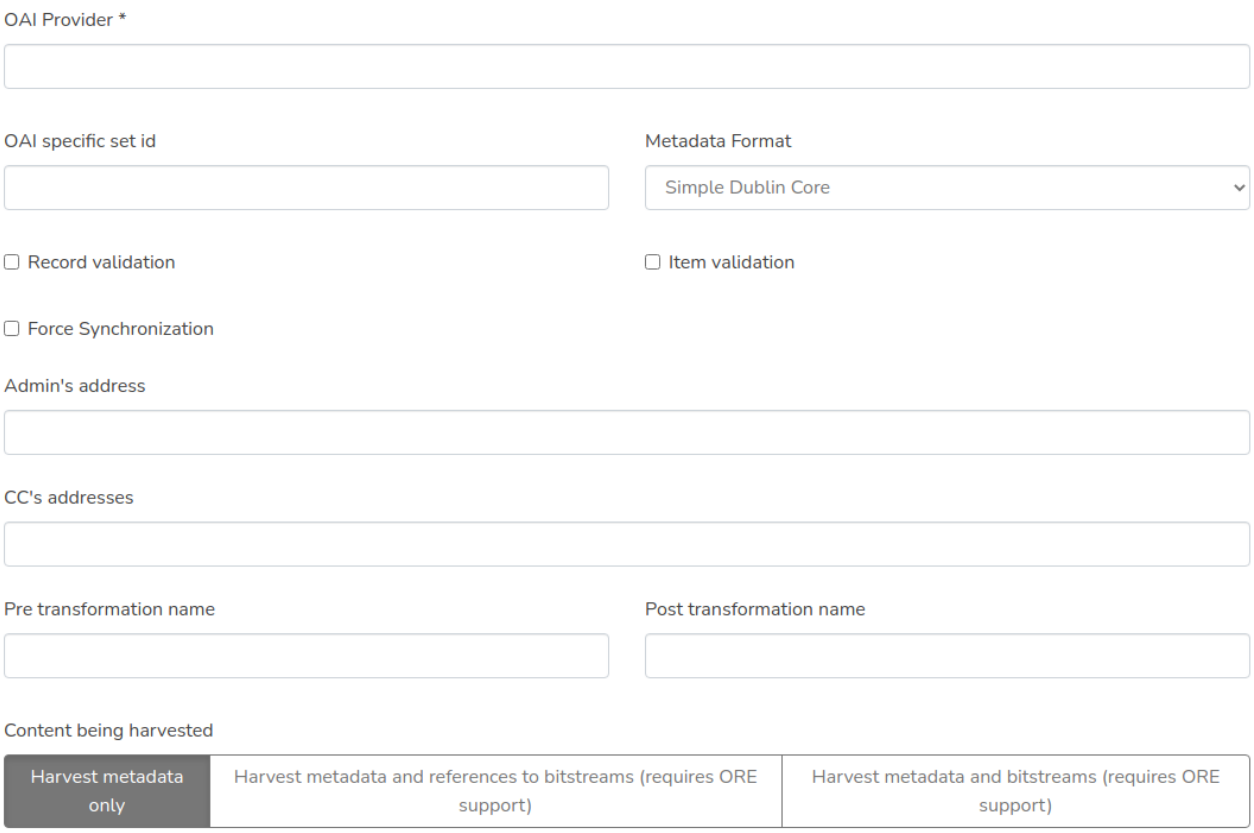
The **Open Archives Initiative Protocol for Metadata Harvesting** (**OAI-PMH**) is a protocol developed for harvesting metadata descriptions of records in an archive so that services can be built using metadata from many archives. OAI-PMH is based on a client–server architecture, in which "harvesters" request information on updated records from "repositories". Requests for data can be based on a datestamp range, and can be restricted to named sets defined by the provider. Data providers are required to provide XML metadata in Dublin Core format, and may also provide it in other XML formats.

It is possible to configure which collections should be populated through imports from external repositories via the OAI-PMH protocol.

### Collection configuration

To configure a collection so that its items are harvested from an external repository there is a specific section in **Edit Collection** named **Content Source** that allows to specify that the collection being edited harvests its content from an external source.






The external source configuration contains:

- **OAI Provider**: the base url of the external source
- **OAI specific set id:** identifies the id of a set with which repositories can organize items (optional)
- **Metadat format:** the metadata format with which to import resources from the repository. The available values are configurable through the properties **oai.harvester.metadataformats.{name} = {namespace}, {optional display name}**, where the {name} value must correspond to a declared ingestion crosswalk (see the Ingestion Crosswalk section). Currently the formats available are:
  - Simple Dublin Core (namespace http://www.openarchives.org/OAI/2.0/oai_dc/)
  - Qualified Dublin Core (namespace http://purl.org/dc/terms/)
  - DSpace Intermediate Metadata (namespace http://www.dspace.org/xmlns/dspace/dim\)
  - OpenAIRE CERIF (namespace https://www.openaire.eu/cerif-profile/1.1/)
- **Admin's email:** the email to which to send notifications if a not recoverable error occurs during harvesting. If set to IDENTIFY, the address to which the emails will be sent will be obtained using the Identify method from the repository, by reading the adminEmail element contained in the response.
- **Pre transformation and post transformation names:** the names of the xsl files with which to perform the transformations of the xml records before and after the transformation between the incoming format and the internal format DIM. These files must be present in the directory whose path corresponds to the **oai.harvester.tranformation-dir** property. Currently, only the harvest in CERIF format supports these two transformations.
- **Content being harvested:** you can choose if harvest metadata only, harvest metadata and references to bitstream or harvest metadata and bitstream. The last two options need the repository to support **ORE**, a metadata format which, however, also contains data relating to any bitstreams.

Once the configuration has been saved, a record will be created in the **harvested_collection** table with the reference to the collection just edited. For more details on this table see "**Harvested Collection and Harvested Item tables**" section.

The data entered in relation to the admin email and the pre and post transformation names will instead be added as collection metadata (**cris.harvesting.email**, **cris.harvesting.preTransform** e **cris.harvesting.postTransform**).

## Perform the harvesting

To manage the collections harvested via OAI-PMH you can start the script named **harvest** ( implemented by the class **org.dspace.app.harvest.Harvest**) in several modes: from command line ({dspace-installation-dir}/bin/dspace harvest), or from processes dashboard, for Administrator users. The script can accept the following options:

| Name | Long name | Description | Parameters |
|---|---|---|---|
| p | purge | Delete all items in the collection | e: eperson email<br>c: collection uuid or handle |

| | | | |
|---|---|---|---|
| r | run | Run the standard harvest procedure | e: eperson email<br>c: collection uuid or handle<br>f: if present force synchronization (harvest all record and ignore already updated items)<br>w: if present the workflow for created items is started instead of directly submit them<br>rv: if present enable incoming records validation<br>iv: if present enable item validation before installing or submitting it |
| g | ping | Test the OAI server and set | a: the OAI source base URL<br>i: the OAI set id<br>m: the metadata format prefix |
| s | setup | Set the collection up for harvesting | c: collection uuid or handle<br>a: the OAI source base URL<br>i: the OAI set id<br>m: the metadata format prefix<br>t: the harvesting type * |
| R | reset | Reset harvest status on all collections | |
| P | purge | Purge all harvestable collections | e: eperson email |

* 0 none, 1 to harvest only metadata, 2 to harvest metadata and references to bitstreams, 3 to harvest metadata and bitstreams

## Harvested Collection and Harvested Item tables

For each collection configured to be populated by harvest, a single record into the **harvested_collection** table is present with the following information:

- the collection uuid
- the harvest type
- the OAI source base url
- the OAI set id
- the metadata prefix name
- the harvest status ( the possible values are 0 for collection ready to be process, 1 for in progress, 2 for queued for scheduling, 3 for OAI error, 4 for harvest to be retried and -1 for unknown error)
- the start time of the last attempt to harvest
- the start time of the last harvest completed without failures
- a message related to the last harvest event

Furthermore, for each item inserted in the collection via harvest, a record is inserted in the **harvested_item** table with:

- the item uuid
- the start time of the last harvest in which the related item was created or updated
- the id of the related item into the external source (the OAI id)

## Harvesting run details

Given a specific collection, the harvest is started if:

- the harvesting type is not 0 (NONE)
- the OAI source and OAI set id are set
- the harvested collection status is not -1 or 1 (ERROR or BUSY)

If the chosen collection is harvestable, its status is set to 1 (BUSY) and the harvesting process starts.
The harvest for a specific collection is done by calling the configured repository (OAI source) with the following query parameters:

- **verb** set to ListRecords to harvest records from the repository
- **from** specifies a lower bound for timestamp-based selective harvesting. It is set equal to the last successful harvesting timestamp for that collection. If no harvesting has previously been completed without fail, this parameter is not set. The date format is calculated with an **Identify** request to the repository by reading the granularity element
- **until** specifies a upper bound for timestamp-based selective harvesting. It is set with the current timestamp. The date format is calculated with an Identify request to the repository by reading the granularity element.
- **set** specifies set criteria for selective harvesting. If configured It is set with the OAI set id value
- **metadataPrefix** specifies the metadata prefix of the format that should be included in the metadata part of the returned records. The metadata prefix is retrieved from the repository searching with the **ListMetadataFormats** request a prefix related to the metadata forma namespace configured by the user for the collection harvesting.

For each record harvested, it is checked whether it is already associated with an item in DSpace through two searches:

- searching for an harvested_item record related to the OAI id of the given record
- searching for an item with a **cris.sourceId** equals to **<repository-name>::<resource-id>**, where repository name is the repository identifier retrievable with an Identify request and the resource id is the id specified in the root metadata element of the given record

If an item is identified with these searches and the incoming record has not deleted status, then the item itself is updated if the timestamp of the last modification associated with the record is more recent than the last timestamp in which the item was harvested. If, on the other hand, an item is not found with the search described previously then it is created. If the imported record has the status deleted then if in the past an item was already created for the same record, this is deleted.

If an item is created for a harvested record, it is validated if validation is enabled (option -V) and, if successful, it is either archived or started in the workflow based on the presence of the -W option.

The update of an item is done by removing the metadata of the item and then repopulating them based on the content of the harvested record. The preliminary deletion of the metadata excludes those configured through the **oai.harvester.update.metadata-to-keep** property.

The harvesting is paged using a **resumption token** and therefore completed the iteration on the current records if in the repository response there is a resumption token then a new ListRecords request is performed passing that token, otherwise the harvesting is concluded.

If errors occur during the import of some records, the process is not blocked but continues in the attempt to import the subsequent records. If, on the other hand, an unrecoverable error occurs (such as the inability to contact the repository) then the process is blocked and an email notification is sent to the admin at the address specified by the **cris.harvesting.email** metadata. In both cases the status of the harvested collection record is set to 4 (**RETRY**). If, on the other hand, no error occurs during the import of the records, the status is set to 0 (**READY**) and the last harvesting date is set to process start timestamp.

The population of the item starting from the harvested record is managed by a particular instance of a class that implements the **IngestionCrosswalk** interface, identified through the plugin framework starting from the configuration of the desired metadata format.

### CERIFIngestionCrosswalk

The CERIFIngestionCrosswalk class fill the newly created item or to update from the xml in CERIF format obtained from the repository. To do this, an intermediate XSLT transformation is used to transform the xml in CERIF format into an xml in the internal DIM format. The DIM xml will then be used to add metadata to the item, through a further implementation of the IngestionCrosswalk interface (DIMIngestionCrosswalk).
The xsl file used for this transformation varies according to the type of the entity to be filled and is identified by name by concatenating the relationship.type of the item to the prefix **oai_cerif_to_dim_**. These files must be placed in the folder **crosswalks/oai/metadataFormats/** in the config dir.

In the CERIF to DIM transformation it is possible to add a prefix of the type **<converterName>@@<value>** to the metadata values to perform a conversion of the <value> through the MapConverter named <converterName>.

For example the snippet of xslt

```
<dim:field mdschema="dc" element="type" >
  <xsl:value-of select="concat
('coarToPublicationTypes',$converterSeparator,pt:Type)" />
</dim:field>
```

will produce the following element in the DIM xml

```
<dim:field mdschema="dc" element="type" >
  Article
</dim:field>
```

If the pt:Type element in the incoming CERIF xml has a value that the converter coarToPublicationTypes associates to "Article".

Furthermore, the CERIFIngestionCrosswalk provides the possibility to specify the path of two xsl files to carry out two intermediate transformations before the CERIF to DIM transformation and after. In this way it is possible to manipulate the CERIF xml before converting it to DIM and also to manipulate the same DIM produced.

### Incoming record and created item validation

It is possible to perform two different validations on the records that are imported:

- record validation via xsd
- item validation using the DSpace validation framework

To specify the directory in which there are the xsd to be used for the validation of the record, it is necessary to value the property `oai.harvester.validation-dir`. Furthermore, it is also necessary to specify the name of the xsd by evaluating the property `oai.harvester.validation.<metadataConfig>.xsd` , where **metadataConfig** is the metadata format of the incoming resources. Configuration example:

```
oai.harvester.validation-dir = ${dspace.dir}/config/crosswalks/oai
/validation
oai.harvester.validation.cerif.xsd = openaire/openaire-cerif-profile.xsd
oai.harvester.validation.perucris-cerif.xsd = perucris/perucris-cerif-
profile.xsd
```

The class that performs the validation given the xsd is org.dspace.harvest.**OAIHarvesterValidatorImpl**.

It is also possible to enable the validation of the item created from the imported record using the validation framework, the same used to validate the items being submitted. For more details see the item validation page.

To establish whether or not to perform one of the two validations, or both, the parameters **rv** (record validation) and **ri** (item validation) can be used. It is also possible to establish the default for each collection to be harvested by configuring the metadata **cris.harvesting.itemValidationEnabled** and **cris.harvesting.recordValidationEnabled**. From the graphic interface, these metadata can be configured by ticking the appropriate checkboxes on the harvest configuration page.

A workspace item associated with the imported resource is also created in the event of record or item validation errors. If there are no validation errors of the item then the workspace item is submitted by starting the workflow or archiving it.

## Import report

Once an import has been completed, if there have been validation errors or generic errors, an xls report is built and sent to the configured emails. The file is composed by a single sheet with the following columns:

- **Record identifier** the record identifier
- **Record link** the absolute url of the record
- **Error** a message with the error's detail
- **Action** the action performed (created, updated, deleted, none)

Example:

| Record identifier | Record link | Error | Action |
|---|---|---|---|
| oai:test:Publications/123 | https://test/oai?verb=GetRecord&identifier=oai:test:Publications/123&metadataPrefix=oai_cerif_openaire | cvc-complex-type.2.3: Element 'oai_cerif:Publishers' cannot have character [children], because the type's content type is element-only. | created |
| oai:test:Publications/456 | https://test/oai?verb=GetRecord&identifier=oai:test:Publications/456&metadataPrefix=oai_cerif_openaire | It was not possible to produce an xml in DIM format | none |

This xls is attached to an email with the content specified by the **harvesting_completed_with_errors** template and is sent to an email address that depends on the **cris.harvesting.email** metadata value of the collection, as described in the "**Collection configuration**" section. In addition, the addresses configured with the metadata **cris.harvesting.ccAddress** are put as cc.

If, on the other hand, an unexpected non-recoverable error has occurred that blocked the import, an email is sent to the same addresses indicated above but without attachments and with the content specified in the **harvesting_error** template.

Live Import Framework

**General Framework**

## Introduction

This documentation explains the features and the usage of the importer framework.
Implementation specific or additional configuration can be found in their related documentation, if any.
Please refer to subdivisions of this documentation for specific implementations of the framework.

## Features

- Lookup publications from remote sources
- Import from files
- Support for multiple implementations

## Abstraction of input format

The importer framework does not enforce a specific input format. Each importer implementation defines which input format it expects from a remote source or file. The import framework uses generics to achieve this. Each importer implementation will have a type set of the record type it receives from the remote source's response. This type set will also be used by the framework to use the correct MetadataFieldMapping for a certain implementation.

## Transformation to DSpace item

The framework produces an 'ImportRecord' that is completely decoupled from DSPace. It contains a set of metadata DTO's that contain the notion of schema,element and qualifier. The specific implementation is responsible for populating this set. It is then very simple to create a DSPace item from this list.

#### Implementation of an import source for External Sources

Each external source importer implementation must at least implements `org.dspace.importer.external.service.components.QuerySource`, which provides the query method used by the framework to retrieve data from the remote source (e.g. Pubmed, ArXiv, etc).

Each external source importer must implements, according to the provider APIs, the declared methods.

An useful abstract for remote sources is `org.dspace.importer.external.service.components.AbstractRemoteMetadataSource`. This class contains functionality to handle request timeout and to retry requests. Using this abstract, the query method must implements `java.util.concurrent.Callable`.

#### Implementation of an import source for files

Each file importer implementation must at least implements `org.dspace.importer.external.service.components.FileSource`, which provides the basic methods used by the framework to parse and load data from the file (e.g. CSV, Endnote, etc).

Each importer must implements the method:

```
public List<ImportRecord> getRecords(InputStream inputStream)
    throws FileSourceException;
```

This method is responsible to transform the input data into an ImportRecord list, which will then managed by the top layer of the framework.

The conversion from raw data to an ImportRecord could be done using the framework too, using the metadata mapping structure (see also).

File sources needs to know which file extensions they have to supports. This is done by the default method `isValidSourceForFile` in `FileSource`, and is controlled by the entries in the list returned by declared method `public List<String> getSupportedExtensions();`

An useful abstract for file source is `org.dspace.importer.external.service.components.AbstractPlainMetadataSource`. It should be used whenever it is possible to model the data in the file as a list of key-value lists (e.g. for CSV files, any row is a key value list).

#### Mapping raw data to Metadata

The framework core is a mid-layer component which allow the conversion of raw data into metadata (ImportRecord) using xml configurable spring beans.

The core of this approach is `org.dspace.importer.externa.service.AbstractImportMetadataSourceService`. Any service that wants to generate metadata from raw data should go through this abstract.

Our service then should extends AbstractImportMetadataSourceService, and use `transformSourceRecords` to transform raw data into ImportRecords.

The most relevant concept in the framework is `private MetadataFieldMapping<RecordType, MetadataContributor<RecordType>> metadataFieldMapping`. This is the place where the framework take the mapping between row data and the associated metadatum. This map must be injected in the service, and will be used by `transformSourceRecords` to convert the data.

`RecordType` is a generic type, which rapresent a single entry of the list of data, and will be mapped to a single ImportRecord. Any metadatum will be mapped to a specific field in the RecordType using a Contributor as described in Metadata mapping.

## Inherited methods

Method getImportSource() should return a unique identifier. Importer implementations should not be called directly, but class *org.dspace.importer.external.service.ImportService* should be called instead. This class contains the same methods as the importer implementations, but with an extra parameter 'url'. This url parameter should contain the same identifier that is returned by the getImportSource() method of the importer implementation you want to use.

The other inherited methods are used to query the remote source.

#### Spring configuration for External Sources

In order to make the live import providers available, them must be mapped as spring beans into dspace-api/src/main/resources/spring/spring-dspace-addon-import-services.xml.

This is an example of a provider which allow to import both files and remote source.

```
    <bean id="PubmedImportService"
          class="org.dspace.importer.external.pubmed.service.
PubmedImportMetadataSourceServiceImpl" scope="singleton">
        <property name="metadataFieldMapping" ref="
PubmedMetadataFieldMapping"/>
        <property name="supportedExtensions">
            <list>
                 <value>xml</value>
            </list>
        </property>
        ...
    </bean>
```

Here is defined the service responsible to fetch and transform the data `PubmedImportMetadataSourceServiceImpl`, which is an extension of`AbstractImportMetadataSourceService` as described above.

The field `metadataFieldMapping` is an instance of `Map<MetadataFieldConfig,MetadataContributor>` and contains the effective mapping.

`supportedExtensions` is the file extension this provider supports.

To expose this provider as Live Import provider, we need to construct a bean of type `org.dspace.external.provider.impl.LiveImportDataProvider` in the following way

```
    <bean id="pubmedLiveImportDataProvider" class="org.dspace.external.
provider.impl.LiveImportDataProvider">
        <property name="metadataSource" ref="PubmedImportService"/>
        <property name="sourceIdentifier" value="pubmed"/>
        <property name="recordIdMetadata" value="dc.identifier.other"/>
    </bean>
```

where `metadataSource` is the bean referencing to live import service as described in “Metadata mapping”, `sourceIdentifier` the name of the provider in the live import framework and `recordIdMetadata` the metadatum used as id of the ImportRecord.

## Metadata mapping

When using an implementation of AbstractImportSourceService, a mapping of remote record fields to DSpace metadata fields can be created.

first create an implementation of class AbstractMetadataFieldMapping with the same type set used for the importer implementation.

Then create a spring configuration file in `[dspace.dir]/config/spring/api`.

Each DSpace metadata field that will be used for the mapping must first be configured as a spring bean of class*org.dspace.importer.external.metadatamapping.MetadataFieldConfig*.

```
    <bean id="dc.title" class="org.dspace.importer.external.
metadatamapping.MetadataFieldConfig">
        <constructor-arg value="dc.title"/>
    </bean>
```

Now this metadata field can be used to create a mapping. To add a mapping for the "dc.title" field declared above, a new spring bean configuration of a class class *org.dspace.importer.external.metadatamapping.contributor.MetadataContributor* needs to be added. This interface contains a type argument. The type needs to match the type used in the implementation of AbstractImportSourceService. The responsibility of each MetadataContributor implementation is to generate a set of metadata from the retrieved document. How it does that is completely opaque to the AbstractImportSourceService but it is assumed that only one entity (i.e. item) is fed to the metadatum contributor.

For example `java SimpleXpathMetadatumContributor implements MetadataContributor<OMElement>` can parse a fragment of xml and generate one or more metadata values.

This bean expects 2 property values:

- field: A reference to the configured spring bean of the DSpace metadata field. e.g. the "dc.title" bean declared above.
- query: The xpath expression used to select the record value returned by the remote source.

```
    <bean id="titleContrib" class="org.dspace.importer.external.
metadatamapping.contributor.SimpleXpathMetadatumContributor">
        <property name="field" ref="dc.title"/>
        <property name="query" value="dc:title"/>
    </bean>
```

Multiple record fields can also be combined into one value. To implement a combined mapping first create a *SimpleXpathMetadatumContributor* a s explained above for each part of the field.

```
    <bean id="lastNameContrib" class="org.dspace.importer.external.
metadatamapping.contributor.SimpleXpathMetadatumContributor">
        <property name="field" ref="dc.contributor.author"/>
        <property name="query" value="x:authors/x:author/x:surname"/>
    </bean>
    <bean id="firstNameContrib" class="org.dspace.importer.external.
metadatamapping.contributor.SimpleXpathMetadatumContributor">
        <property name="field" ref="dc.contributor.author"/>
        <property name="query" value="x:authors/x:author/x:given-name"/>
    </bean>
```

Note that namespace prefixes used in the xpath queries are configured in bean "FullprefixMapping" in the same spring file.

```
    <util:map id="FullprefixMapping" key-type="java.lang.String" value-
type="java.lang.String">
        <description>Defines the namespace mappin for the
SimpleXpathMetadatum contributors</description>
        <entry key="http://purl.org/dc/elements/1.1/" value="dc"/>
        <entry key="http://www.w3.org/2005/Atom" value="x"/>
    </util:map>
```

Then create a new list in the spring configuration containing references to all *SimpleXpathMetadatumContributor* beans that need to be combined.

```
   <util:list id="combinedauthorList" value-type="org.dspace.importer.
external.metadatamapping.contributor.MetadataContributor" list-class="
java.util.LinkedList">
        <ref bean="lastNameContrib"/>
        <ref bean="firstNameContrib"/>
    </util:list>
```

Finally create a spring bean configuration of class*org.dspace.importer.external.metadatamapping.contributor.CombinedMetadatumContributor*. This bean expects 3 values:

- field: A reference to the configured spring bean of the DSpace metadata field. e.g. the "dc.title" bean declared above.
- metadatumContributors: A reference to the list containing all the single record field mappings that need to be combined.
- separator: These characters will be added between each record field value when they are combined into one field.

```
    <bean id="authorContrib" class="org.dspace.importer.external.
metadatamapping.contributor.CombinedMetadatumContributor">
        <property name="separator" value=", "/>
        <property name="metadatumContributors" ref="combinedauthorList"
/>
        <property name="field" ref="dc.contributor.author"/>
    </bean>
```

Each contributor must also be added to the "MetadataFieldMap" used by the *MetadataFieldMapping* implementation. Each entry of this map maps a metadata field bean to a contributor. For the contributors created above this results in the following configuration:

```
    <util:map id="org.dspace.importer.external.metadatamapping.
MetadataFieldConfig"
              value-type="org.dspace.importer.external.metadatamapping.
contributor.MetadataContributor">
        <entry key-ref="dc.title" value-ref="titleContrib"/>
        <entry key-ref="dc.contributor.author" value-ref="authorContrib"
/>
    </util:map>
```

Note that the single field mappings used for the combined author mapping are not added to this list.

**MappedMetadataContributor**

It is a particular contributor that uses another contributor to extract metadata value from source, and transforms its output according to an internal mapping and a default value.

For example in this snippet of a custom mapping applied for the Peruvian Alicia Database (implemented with VuFind):

```
<bean id="aliciaTypeVersion" class="org.dspace.importer.external.
metadatamapping.contributor.MappedMetadataContributor">
        <constructor-arg name="innerContributor" ref="
aliciaInnerTypeVersion"/>
        <constructor-arg name="mapConverter" ref="
aliciaTypeVersionMapConverter"/>
</bean>
```

```
<bean name="aliciaTypeVersionMapConverter" class="org.dspace.util.
SimpleMapConverter" init-method="init">
        <constructor-arg value="aliciaCoarTypes" />
        <property name="converterNameFile" value="mapConverter-
aliciaCoarTypes.properties" />
        <property name="configurationService" ref="org.dspace.services.
ConfigurationService" />
        <property name="defaultValue" value="NA"/>
</bean>
```

```
<bean id="aliciaInnerTypeVersion" class="org.dspace.importer.external.
metadatamapping.contributor.SimpleJsonPathMetadataContributor">
        <property name="field" ref="alicia.dc.type.version"/>
        <property name="query" value="$.dcTypeVersion[*]"/>
    </bean>
    <bean id="alicia.dc.type.version" class="org.dspace.importer.
external.metadatamapping.MetadataFieldConfig">
        <constructor-arg value="dc.type.version"/>
    </bean>
```

a MappedMetadataContributor is defined, that takes outputs returned by "`aliciaInnerTypeVersion`" , in this case values belonging to a controlled vocabulary, and transforms it according to a `SimpleMapConverter` that map values according to a given property file:

```
info\:eu-repo/semantics/draft = AO
info\:eu-repo/semantics/submittedVersion = SMUR
info\:eu-repo/semantics/acceptedVersion = AM
info\:eu-repo/semantics/publishedVersion = VoR
info\:eu-repo/semantics/updatedVersion = EVoR
```

### Available metadata contributor

| Class | Description |
|---|---|
| `SimpleXpathMetadatumContributor` | Use an XPath expression to map the XPath result to a metadatum |
| `SimpleXpathDateFormatMetadataContributor` | Use an XPath expression to map the XPath result to a metadatum. The resulting value will be used as date and reformatted using DateFormat. |
| `CombinedMetadatumContributor` | Use a LinkedList of MetadataContributor to combine into the value the resulting value for each contributor. |
| `ReplaceCharacterXPathMetadataContributor` | Use an XPath expression to extract the value. In the resulting value, all characters `characterToBeReplaced` will be replaced with `characterToReplaceWith` |
| `SimpleMetadataContributor` | This contributor is used in plain metadata as exposed above. Mapping is easy because it is based on the key used in the DTO. |
| `SimpleJsonPathMetadataContributor` | Use a JSon path expression to map json values to a metadatum |
| `SimpleJsonPathKeyMetadataContributor` | Map all subkeys (fields name) of a specific json nodes to a metadatum. The JSon path expression must extract to the parent key of the keys to map. |
| `MappedMetadataContributor` | Returns value(s) extracted by another Contributor with value(s) transformed according to an internal map and a default value. |
| `WosIdentifierRidContributor` | Custom contributor to extract Rid from Web Of Science response |
| `WosAttribute2ValueContributor` | Custom contributor to extract a value from a given structure inside Web Of Science response |
| `WosIdentifierContributor` | Custom contributor to extract an identifier from Web Of Science response |

#### Available external sources

### Pubmed

Pubmed integration use as Metadata Field Map the map defined in dspace/config/spring/api/pubmed-integration.xml:

| Pubmed field (XPath) | Metadatum | Contributor |
|---|---|---|
| descendant::Language | dc.language.iso | SimpleXpathMetadatumContributor |
| descendant::AbstractText | dc.description.abstract | SimpleXpathMetadatumContributor |
| descendant::ArticleTitle | dc.title | SimpleXpathMetadatumContributor |
| descendant::MedlineCitation/PMID | dc.identifier.pmid | SimpleXpathMetadatumContributor |
| descendant::Author/LastName + descendant::Author/ForeName | dc.contributor.author | CombinedMetadatumContributor |
| descendant::PubDate/Year + descendant::PubDate/Month + descendant::PubDate/Day | dc.date.issued | PubmedDateMetadatumContribut |
| descendant::Language | dc.language.iso | SimpleXpathMetadatumContributor |
| descendant::Keyword | dc.subject | SimpleXpathMetadatumContributor |
| descendant::Journal/Title | dc.relation.ispartof | SimpleXpathMetadatumContributor |
| descendant::Journal/ISSN | dc.relation.issn | SimpleXpathMetadatumContributor |
| descendant::Volume | oaire.citation.volume | SimpleXpathMetadatumContributor |
| descendant::JournalIssue/Issue | oaire.citation.issue | SimpleXpathMetadatumContributor |
| | dc.identifier.doi | SimpleXpathMetadatumContributor |

| | | |
|---|---|---|
| descendant::ELocationID[@EIdType='doi' and @ValidYN='Y'] | | |
| descendant::PublicationTypeList /PublicationType | dc.type | SimpleXpathMetadatumContributor |

### ArXiv Integration

ArXiv integration use as Metadata Field Map the map defined in dspace/config/spring/api/arxiv-integration.xml:

| ArXiv field (XPath) | Metadatum | Contributor |
|---|---|---|
| ns:id | dc.identifier.other | ArXivIdMetadataContributor |
| ns:title | dc.title | SimpleXpathMetadatumContributor |
| ns:summary | dc.description.abstract | SimpleXpathMetadatumContributor |
| ns:published | dc.date.issued | SimpleXpathMetadatumContributor |
| arxiv:doi | dc.identifier | SimpleXpathMetadatumContributor |
| arxiv:journal_ref | dc.source | SimpleXpathMetadatumContributor |
| ns:category/@term | dc.subject | SimpleXpathMetadatumContributor |
| ns:author/name | dc.contributor.author | SimpleXpathMetadatumContributor |

with the following namespace setting:

```
    <util:map id="arxivBasePrefixToNamespaceMapping" map-class="java.
util.HashMap"
        key-type="java.lang.String" value-type="java.lang.String">
            <entry key="http://www.w3.org/2005/Atom" value="ns" />
    </util:map>


    <util:map id="arxivArxivPrefixToNamespaceMapping" map-class="java.
util.HashMap"
        key-type="java.lang.String" value-type="java.lang.String">
            <entry key="http://arxiv.org/schemas/atom" value="arxiv" />
    </util:map>
```

### CrossRef Integration

CrossRef integration use as Metadata Field Map the map defined in dspace/config/spring/api/crossref-integration.xml:

| CrossRef field (JsonPath) | Metadatum | Contributor |
|---|---|---|
| `$.title` | `dc.title` | `SimpleJsonPathMetadataContributor` |
| $.author[*].given + $.author[*].family | dc.contributor.author | SimpleJsonPathMetadataContributor |
| $.ISBN | dc.identifier.isbn | SimpleJsonPathMetadataContributor |
| $.issued.date-parts[0][0] | dc.date.issued | SimpleJsonPathMetadataContributor |
| $.editor[*].given + $.editor[*].family | dc.contributor.editor | SimpleJsonPathMetadataContributor |
| $.type | dc.type | SimpleJsonPathMetadataContributor |
| $.DOI | dc.identifier.doi | SimpleJsonPathMetadataContributor |

| | | |
|---|---|---|
| $.container-title | dc.relation.ispartof | SimpleJsonPathMetadataContributor |
| $.DOI | dc.identifier.other | SimpleJsonPathMetadataContributor |
| $.ISSN | dc.relation.issn | SimpleJsonPathMetadataContributor |
| $.volume | oaire.citation.volume | SimpleJsonPathMetadataContributor |
| $.journal-issue.issue | oaire.citation.issue | SimpleJsonPathMetadataContributor |
| $.abstract | dc.description.abstract | SimpleJsonPathMetadataContributor |

## Scopus Integration

Scopus integration use as Metadata Field Map the map defined in dspace/config/spring/api/scopus-integration.xml:

| Scopus field (XPath) | Metadatum | Contributor |
|---|---|---|
| prism:doi | dc.identifier.doi | SimpleXpathMetadatumContributor |
| dc:title | dc.title | SimpleXpathMetadatumContributor |
| prism:aggregationType | dc.type | SimpleXpathMetadatumContributor |
| prism:isbn | dc.identifier.isbn | SimpleXpathMetadatumContributor |
| prism:issn | dc.identifier.issn | SimpleXpathMetadatumContributor |
| prism:coverDate | dc.date.issued | SimpleXpathMetadatumContributor |
| dc:description | dc.description.abstract | SimpleXpathMetadatumContributor |
| ns:pubmed-id | dc.identifier.pmid | SimpleXpathMetadatumContributor |
| prism:volume | oaire.citation.volume | SimpleXpathMetadatumContributor |
| prism:issueIdentifier | oaire.citation.issue | SimpleXpathMetadatumContributor |
| ns:eid | dc.identifier.scopus | SimpleXpathMetadatumContributor |
| ns:fund-sponsor | dc.relation.funding | SimpleXpathMetadatumContributor |
| ns:fund-no | dc.relation.grantno | SimpleXpathMetadatumContributor |
| ns:authkeywords | ns:authkeywords | SimpleXpathMetadatumContributor |
| ns:article-number | oairecerif.citation.number | SimpleXpathMetadatumContributor |
| dc:creator | dc.contributor.author | SimpleXpathMetadatumContributor |
| ns:openaccessFlag | dc.rights | ReplaceFieldXPathMetadataContributor |
| prism:pageRange | oaire.citation.pages | PageRangeXPathMetadataContributor |
| prism:publicationName | prism:publicationName | SimpleXpathMetadatumContributor |
| prism:publicationName | dc.relation.ispartofseries | SimpleXpathMetadatumContributor |

## Web of Science Integration

Web of Science integration use as Metadata Field Map the map defined in dspace/config/spring/api/wos-integration.xml:

| Wos field (Xpath) | Metadatum | Contributor |
|---|---|---|
| static_data/summary/EWUID/edition/@value | oaire.citation.edition | SimpleXpathMetadatumContributor |
| static_data/summary/pub_info/@pubyear | dc.date.issued | SimpleXpathMetadatumContributor |
| static_data/summary/pub_info/page/@begin | oaire.citation.startPage | SimpleXpathMetadatumContributor |
| static_data/summary/pub_info/page/@end | oaire.citation.endPage | SimpleXpathMetadatumContributor |

| | | |
|---|---|---|
| static_data/summary/pub_info/@pubtype | dc.type | SimpleXpathMetadatumContributor |
| static_data/summary/pub_info/@vol | oaire.citation.volume | SimpleXpathMetadatumContributor |
| static_data/summary/titles/title | dc.relation.ispartof | WosAttribute2ValueContributor |
| static_data/summary/titles/title | dc.relation.ispartofseries | WosAttribute2ValueContributor |
| static_data/summary/titles/title | dc.title | WosAttribute2ValueContributor |
| dynamic_data/cluster_related/identifiers /identifier | dc.identifier.doi, dc.identifier.issn, dc. identifier.isbn | WosIdentifierContributor |
| static_data/summary/pub_info/@issue | oaire.citation.issue | SimpleXpathMetadatumContributor |
| static_data/fullrecord_metadata/abstracts /abstract/abstract_text/p | dc.description.abstract | SimpleConcatContributor |
| static_data/fullrecord_metadata /normalized_languages/@count | dc.language.iso | SimpleXpathMetadatumContributor |
| static_data/summary/names/name/full_name | dc.contributor.author | SimpleXpathMetadatumContributor |
| static_data/item/keywords_plus/keyword, static_data/fullrecord_metadata/keywords /keyword,static_data/fullrecord_metadata /category_info/headings/heading, static_data /fullrecord_metadata/category_info /subheadings/subheading | dc.subject | SimpleMultiplePathContributor |
| static_data/summary/names/name[orcid_id] | person.identifier.orcid | SimpleXpathMetadatumAndAttributeContributor |
| static_data/summary/names/name[role] /full_name | dc.contributor.editor | WosAttribute2ValueContributor |
| UID | dc.identifier.isi | SimpleXpathMetadatumContributor |
| static_data/contributors/contributor | person.identifier.rid | WosIdentifierRidContributor |

### SciELO Integration

SciELO integration uses as Metadata Field Map the map defined in dspace/config/spring/api/scielo-integration.xml:

| SciELO field (RIS) | Metadatum | Contributor |
|---|---|---|
| DO | dc.identifier.doi | SimpleRisToMetadataContributor |
| AU | dc.contributor.author | SimpleRisToMetadataContributor |
| TI | dc.title | SimpleRisToMetadataContributor |
| SN | dc.identifier.issn | SimpleRisToMetadataContributor |
| VL | oaire.citation.volume | SimpleRisToMetadataContributor |
| IS | oaire.citation.issue | SimpleRisToMetadataContributor |
| ID | dc.identifier.other | SimpleRisToMetadataContributor |
| SP | oaire.citation.startPage | SimpleRisToMetadataContributor |
| EP | oaire.citation.endPage | SimpleRisToMetadataContributor |
| KW | dc.subject | SimpleRisToMetadataContributor |
| TY | dc.type | SimpleRisToMetadataContributor |
| PY | dc.date.issued | SimpleRisToMetadataContributor |
| JO | dc.relation.ispartof | SimpleRisToMetadataContributor |

#### VuFind Integration

VuFind integration uses as Metadata Field Map the map defined in dspace/config/spring/api/vufind-integration.xml:

| VuFind field (Json path) | Metadatum | Contributor |
|---|---|---|
| $.dcIdentifierIsbn[*] | dc.identifier.isbn | SimpleJsonPathMetadataContributor |
| $.dcIdentifierDoi[*] | dc.identifier.doi | SimpleJsonPathMetadataContributor |
| $.dcIdentifierHandle[*] | dc.identifier.uri | SimpleJsonPathMetadataContributor |
| $.dcDescriptionSponsorship[*] | dc.description.sponsorship | SimpleJsonPathMetadataContributor |
| $.dcDateIssued[*] | dc.date.issued | SimpleJsonPathMetadataContributor |
| $.dcRelationURI[*] | dc.relation.uri | SimpleJsonPathMetadataContributor |
| $.dcContributorEditor[*] | dc.contributor.editor | SimpleJsonPathMetadataContributor |
| $.dcTitleAlternative[*] | dc.title.alternative | SimpleJsonPathMetadataContributor |
| $.dcTableOfContents[*] | dc.description.tableOfContents | SimpleJsonPathMetadataContributor |
| $.dcDescriptionAbstract[*] | dc.description.abstract | SimpleJsonPathMetadataContributor |
| $.id | dc.identifier.other | SimpleJsonPathMetadataContributor |
| $.languages[*] | dc.language.iso | SimpleJsonPathMetadataContributor |
| $.dcPublisher[*] | dc.publisher | SimpleJsonPathMetadataContributor |
| $.dcTitle[*] | dc.title | SimpleJsonPathMetadataContributor |
| $.urls[*].url | dc.identifier | SimpleJsonPathMetadataContributor |
| $.subjects[*][*] | dc.subject | SimpleJsonPathMetadataContributor |
| $.dcContributorAuthor[*] | dc.contributor.author | SimpleJsonPathKeyMetadataContributor |

#### OpenAIRE Publication Integration

OpenAIRE Project integration uses as Metadata Field Map the map `openaireMetadataFieldMap` defined in dspace/config/spring/api/openaire-integration.xml:

| OpenAIRE field (xpath) | Metadatum | Contributor |
|---|---|---|
| descendant::oaf:result/title | dc.title | SimpleXpathMetadatumContributor |
| descendant::dri:objIdentifier | dc.identifier.other | SimpleXpathMetadatumContributor |
| descendant::oaf:result/pid | dc.identifier | SimpleXpathMetadatumContributor |
| descendant::oaf:result/relevantdate | dc.date.issued | SimpleXpathDateFormatMetadataContributor |
| descendant::oaf:result/subject | dc.subject | SimpleXpathMetadatumContributor |
| descendant::oaf:result/creator | dc.contributor.author | SimpleXpathMetadatumContributor |
| descendant::oaf:result/description | dc.description.abstract | SimpleXpathMetadatumContributor |

#### OpenAIRE Project (Funding) Integration

OpenAIRE Project integration uses as Metadata Field Map the map `openaireProjectsMetadataFieldMap` defined in dspace/config/spring/api/openaire-integration.xml:

| OpenAIRE field (xpath) | Metadatum | Contributor |
|---|---|---|
| .//title | dc.title | SimpleXpathMetadatumContributor |
| .//acronym | oairecerif.acronym | SimpleXpathMetadatumContributor |
| .//code | oairecerif.funding.identifier | SimpleXpathMetadatumContributor |

| | | |
|---|---|---|
| .//startdate | oairecerif.funding.startDate | SimpleXpathDateFormatMetadataContributor |
| .//enddate | oairecerif.funding.endDate | SimpleXpathDateFormatMetadataContributor |
| .//fundingtree/funder/name | oairecerif.funder | SimpleXpathMetadatumContributor |
| .//callidentifier | oairecerif.fundingParent | SimpleXpathMetadatumContributor |
| .//oamandatepublications | oairecerif.oamandate | SimpleXpathMetadatumContributor |
| .//summary | dc.description | SimpleXpathMetadatumContributor |

#### European Patent Office (EPO) Integration

EPO integration uses as Metadata Field Map the map defined in dspace/config/spring/api/epo-integration.xml:

| OpenAIRE field (xpath) | Metadatum | Contributor |
|---|---|---|
| //ns:invention-title | dc.title | SimpleXpathMetadatumContributor |
| //ns:inventor-name/ns:name | dc.contributor.author | SimpleXpathMetadatumContributor |
| //ns:applicant-name/ns:name | dc.contributor.applicant | SimpleXpathMetadatumContributor |
| //ns:publication-reference | dc.identifier.patentno | SimpleXpathMetadatumContributor |
| //ns:application-reference/document-id [@document-id-type="original"] /doc-number | dc.identifier.applicationnumber | SimpleXpathMetadatumContributor |
| //ns:publication-reference/ns:document-id [@document-id-type="epodoc"] /ns:date | dc.date.issued | SimpleXpathDateFormatMetadataContributor |
| //ns:application-reference/ns:document-id/ns: date | dcterms.dateSubmitted | SimpleXpathDateFormatMetadataContributor |
| //ns:abstract | dc.description.abstract | SimpleXpathMetadatumContributor |

### Available file sources

#### CSV/TSV format

CSV and TSV files are position based. That means, based on the following configuration, a CSV files have to format `dc.title, dc.contributor.author, dc.date.issued,…`

| Position | Metadata | Contributor |
|---|---|---|
| 0 | dc.title | SimpleMetadataContributor |
| 1 | dc.contributor.author | EnhancedSimpleMetadataContributor |
| 2 | dc.date.issued | SimpleMetadataContributor |
| 3 | dc.source | SimpleMetadataContributor |
| 4 | dc.description.abstract | SimpleMetadataContributor |
| 5 | dc.identifier.issn | SimpleMetadataContributor |
| 6 | dc.type | SimpleMetadataContributor |

Note that `EnhancedSimpleMetadataContributor` could process an inner CSV/TSV file. This contributor is useful, for example, if the authors list is as the follow:

```
“\”Surname1, Name\”, \“Surname2, Name2\”,\”…\””
```

### Bibtex format

| | | |
|---|---|---|

| Key | Metadatum | Contributor |
|---|---|---|
| ISSN | dc.identifier.issn | SimpleMetadataContributor |
| title | dc.title | SimpleMetadataContributor |
| year | dc.date.issued | SimpleMetadataContributor |
| journal | dc.source | SimpleMetadataContributor |
| author | dc.contributor.author | SimpleMetadataContributor |

### Endnote format

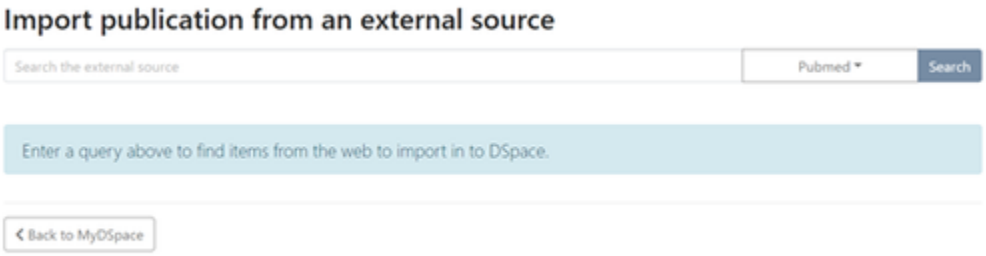
| Key | Metadatum | Contributor |
|---|---|---|
| TI | dc.identifier.issn | SimpleMetadataContributor |
| AU | dc.title | SimpleMetadataContributor |
| PY | dc.date.issued | SimpleMetadataContributor |
| AB | dc.description.abstract | SimpleMetadataContributor |
| SO | dc.contributor.author | SimpleMetadataContributor |

### RIS format

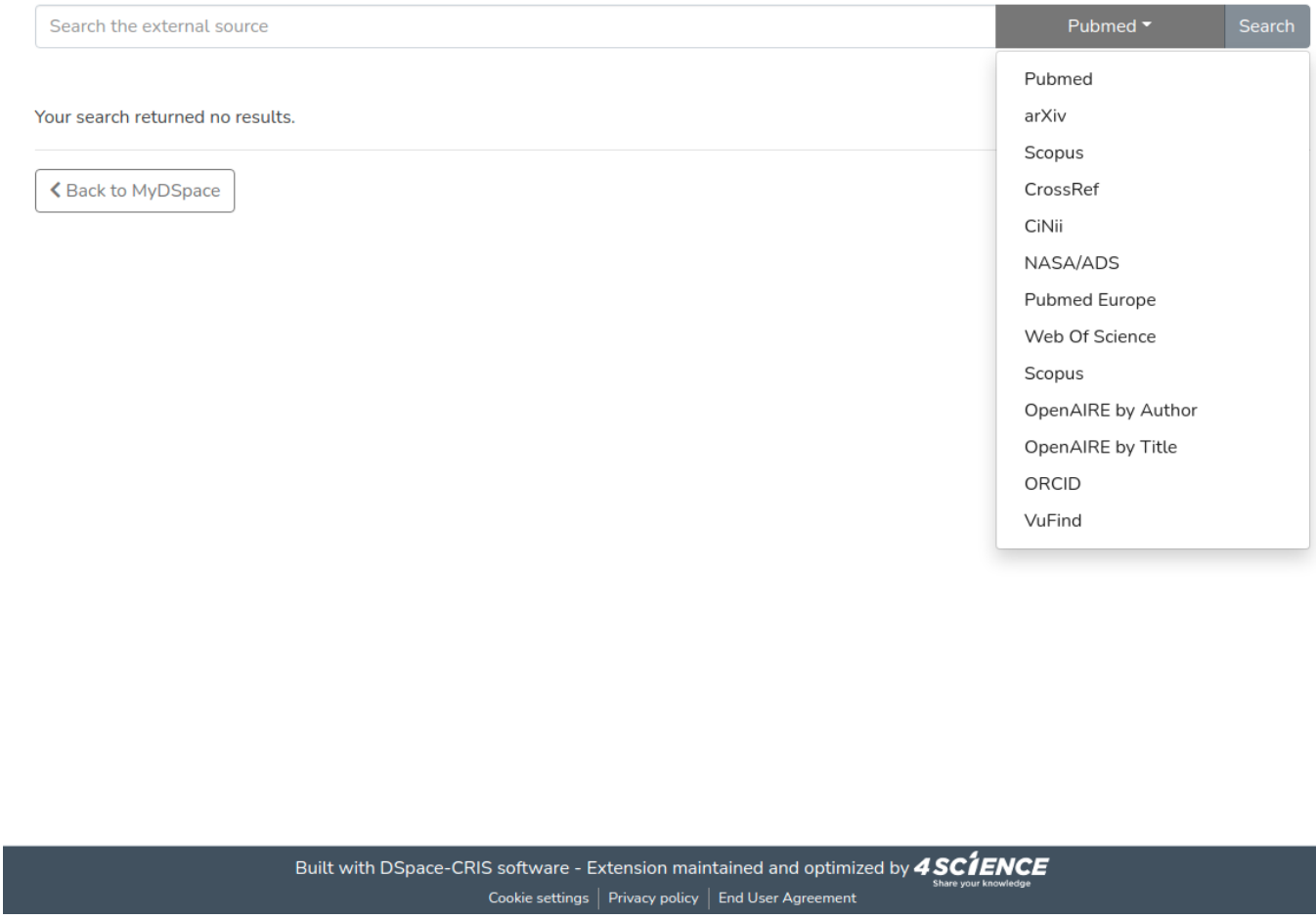
| Key | Metadatum | Contributor |
|---|---|---|
| SN | dc.identifier.issn | SimpleMetadataContributor |
| TI AND T1 | dc.title | MultipleMetadataContributor |
| PY | dc.date.issued | SimpleMetadataContributor |
| AB | dc.description.abstract | SimpleMetadataContributor |
| AU | dc.contributor.author | SimpleMetadataContributor |
| PT | dc.type | SimpleMetadataContributor |
| PY | dc.date.issued | SimpleMetadataContributor |

**Submitting starting from external sources**

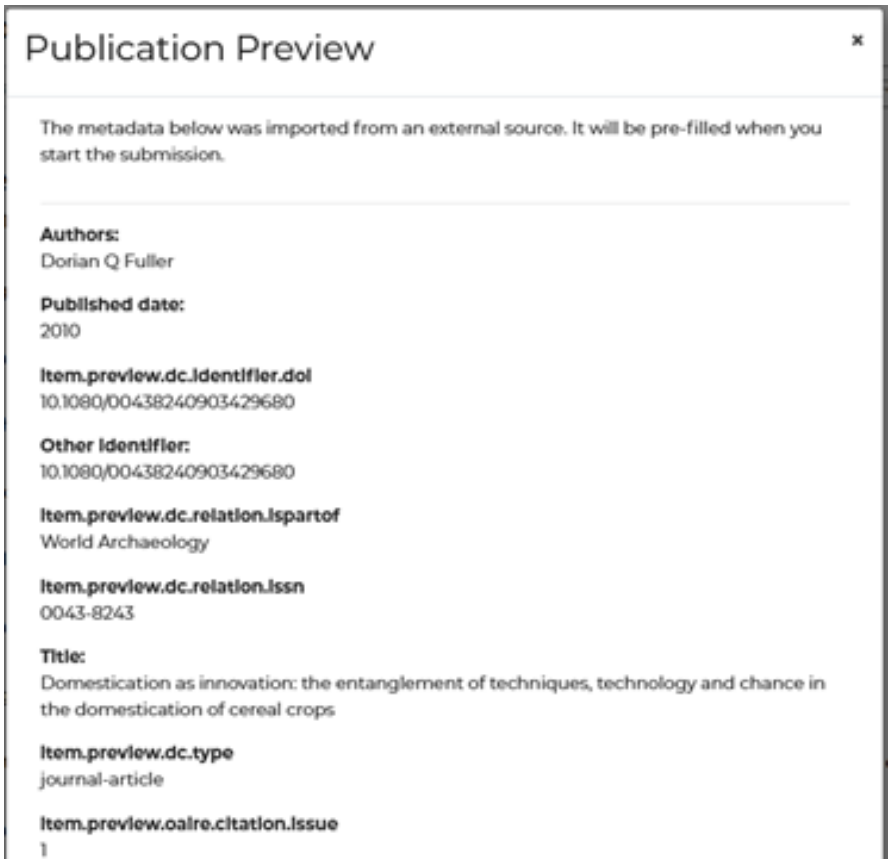
From the myDSpace page a new submission can be started not only using the submission form but also automatically populating metadata, importing them from several online services (up to now Alicia,  PubMed and CrossRef have been implemented).




After choosing the external source to import from and inserting a term in search bar, the system will show the list of matching results.




When selecting an item, the system will display the metadata to be imported, according to the configured mapping.




Clicking on “Start submission” the system will display the submission forms filled with the imported metadata.

**Massive publications import from external services**

It is possible to import publications from Scopus and Web Of Science systems by mean of ‘update-publications’ script.

Script can be run via both process section and CLI, with following syntax

**import-publications <:service>**

where service may assume one of ‘wos’ (for Web Of Science) and ‘scopus’ values, depending on service from where publications should be collected. A single script shall be triggered for every service, it is not possible to run the same script instance to collect publications from many services.

This script will extract all entities in DSpace-CRIS of type ‘Person’ having following metadata set:

- **person.identifier.scopus-author-id** if selected service is ‘scopus’
- **person.identifier.orcid** or **person.identifier.rid** if selected service is ‘wos’

For each person having such metadata set, a query with such metadata value is performed towards the selected service, results will be mapped in DSpace Publications using the mapping defined for LiveImport framework (Live Import Framework (Update) ) . Resulting Publications will be stored in collections as follows:

- If `scopus.importworkspaceitem.collection-id` or `wos.importworkspaceitem.collection-id` properties are defined in d `space.cfg` , with a valid collection uuid, collections identified by those uuids will collect publications imported from Scopus (uuid set in the first key) and Web Of Science (uuid set in the second key.
- If above mentioned keys are not set, system will look up for a valid collection by searching one with entity type set as ‘Publication’, and will add imported publication to this collection.

DBMS Import framework

DSpace-CRIS provides a lot of way to import, update and manipulate both native dspace objects than CRIS objects in bulk. Other than the ones offered by a basic DSpace it is possible to use (also from the UI) excel files (CRIS Objects) or adhoc simplified database tables (currently only DSpace items) to perform operation over the data.

The following database tables have been introduced:

- **imp_record**: contains information about the operations to perform. Each row represent a specific operation on a single item
- **imp_metadatavalue**: contains all the metadata associated with an item that need to be created or updated (optional)
- **imp_bitstream**: contains all the information related to bitstreams to attach / replace in the item (optional)

To elaborate the imp_* tables you need to run the following script:

**org.dspace.app.cris.batch.ItemImportMainOA**

**-p** Send the email for the in archive event to the authors, coauthors, etc. - the workflow email are EVER disabled
**-E** BatchJob User email
**-x** Indexing disabled (improve performance)
**-n** Summary EMail disabled (improve performance)
**-b** Delete bitstream related to the item in the update phase (you need to provide details about the new bitstream or the bitstream to keep in the imp_bitstreams table)
**-m** List of metadata that are cleanup before to perform the operation. By default all metadata are delete except the cris.sourceId metadata, specifying only the dc.title it will obtain an append on the other metadata. Use this option many times on the single metadata e.g. -m dc.title -m dc. contributor.*
**-s** Invert the logic for the -m option, using the option -s only the metadata list with the option -m are saved (ad es. -m dc.description.provenance) the other will be delete except the cris.sourceId metadata
**-S** muted logs
**-t** Threads numbers (default 0, if omitted read by configuration). Very experimental.

## imp_record

Table used by the DBMS Import feature which includes the items to be imported in DSpace-CRIS using the DBMS import framework:

- **imp_id:** the unique ID used to link the operation with the additional data in the other imp_* tables

- imp_record_id: an unique ID for the record in the external source system. This is used togheter with the imp_sourceref to guarantee that subsequent operation over the "same" source record will be performed always on the same DSpace object without forcing the external system to know about DSpace-CRIS
- imp_sourceref: an unique acronym for the system that have provided the data
- imp_eperson_id: the id of the eperson to use to perform the action
- imp_collection_id: the collection where create the item if relevant
- status: can be one of the following values:
  - p = workspace
  - w = workflow step 1
  - y = workflow step 2
  - x = workflow step 3
  - z = in archive
  - g = withdrawn
- operation: can be one of update or delete. Update is used also for record creation
- integra: not used, to be revisited to manage versioning
- last_modified: must be empty. It will be populated when the record is used
- handle: only for creation of new item is it possible to force a specific handle , otherwise the system will assign a new one in the usual way

### imp_metadatavalue

- imp_metadatavalue_id: an unique id sequence generated
- **imp_id: link to the imp_record main table**
- imp_schema: the shortname of the schema (dc, dcterms, etc.)
- imp_element: the element
- imp_qualifier: the qualifier
- imp_value: the textual value of the metadata
- imp_authority: the authority key if any for this value. Since 40eeb989c4354731c0ee3fce6e80d6df64b80c94 the authority and confidence values are used by default as is forcing the metadata creation to skip the getBestMatch method of the authority framework. To guess a potential match it is possible to use the value, case insensitive, **[GUESS]**, to force the use of the authority framework getBestMatch method.
- imp_confidence: the confidence of the authority if any (600 mean accepted match)
- imp_share: not used, for future use
- metadata_order: used to sort the metadata values within the same schema.element.qualifier to insert/update
- text_lang: the lang for the metadata value (en, it, etc.)

### imp_bitstream

- imp_bitstream_id: an unique id sequence generated
- **imp_id: link to the imp_record main table**
- filepath
- description
- bundle: the name of the Bundle where put the bitstream (ORIGINAL, TEXT, etc.)
- bitstream_order: to sort the processing of the rows
- primary_bitstream: flag to mark the bitstream as primary
- assetstore
- name
- imp_blob: the content of the bitstream (alternative to filepath)
- embargo_policy: can be one of:
  - 0 --> mean open access

    - 1 --> embargo (need to use also the embargo_start_date column)
    - 2 --> assign a READ policy to epersongroup ID 2 (you need to create a epersongroup with such ID for "authorized users")
    - 3 --> assign a READ policy only to the administrators group
- embargo_start_date: to use as start date of Anonymous READ  policy when embargo_policy = 1

#### Item tracking

To track the result of creation action the **cris.sourceId** metadata is used, so that subsequent operation over the same origin record will result in update instead of duplication of entries. Each item created through this import will have valued the cris.sourceId metadata with the value **imp_sourceref::imp_record_id**. So at the beginning of the record import, a search for an item is made for a cris.sourceId metadata equals to the same value stored in the way descripted previously.

### Metadata enrichment from authority

DSpace-CRIS provide the opportunity to configure an authority in order to automatically encrich specific metadata during a submission when an authority entry is selected.

#### Configuration

To enable the enrichment you need to edit the configuration file cris-authority-metadatagenerator.xml.

Here a sample configuration :

```
<bean class="org.dspace.content.authority.
ItemSimpleAuthorityMetadataGenerator">
        <property name="authorityName" value="AuthorAuthority"/>
        <property name="relatedInputformMetadata" value="
oairecerif_author_affiliation"/>
        <property name="schema" value="person"/>
        <property name="element" value="affiliation"/>
        <property name="qualifier" value="name"/>
        <property name="singleResultOnAggregate" value="false"/>
</bean>
```

- `authorityName` : is the name of the authority that will provide the metadata enrichment
- `relatedInputformMetadata` : is the name of the metadata belonging to the authority entry which is exposed and used for the enrichment
- `schema` : the schema of the metadata that is enriched in the submission with the value of the metadata condigured in `relatedInputformMetadata`
- `element` : the element of the metadata that is enriched in the submission with the value of the metadata condigured in `relatedInputformMetadata`
- `qualifier` : the qualifier of the metadata that is enriched in the submission with the value of the metadata condigured in `relatedInputformMetadata`

**NB**

Every metadata defined in the `relatedInputformMetadata` property should be present even in the property `discovery.index.projection` of the discovery.cfg file

#### Enrichemnt logic

In the submission form the item metadata are enriched by following those rules :

- if in the submission form the target metadata is not a repeatable field the value is added if empty or replaced if already exists
- if in the submission form the target metadata is a repeatable field a new value is automatically added to the existing ones

## System configuration

In this section we will explain how to configure some general aspects of the system

### Layout and data security configuration tool

To speedup the configuration of the layout and data security aspects of DSpace-CRIS administrators can use the script named “cris-layout-tool”.

That script requires the following parameters:

- f (file)  the source file with the full cris layout configuration

```
dspace-install/bin# ./dspace cris-layout-tool -f ../etc/conftool/cris-
layout-configuration.xls
```

In the folder `dspace/etc/conftool` it is included the excel file `cris-layout-configuration.xls` representing the default configuration of DSpace-CRIS 7. The file has several tabs

- **tab**. It contains the details about all the tabs that need to be created.
  - *Entity*: it is the label of the Entity Type to which the tab belong. A tab's shortname must be unique for a specific entity
  - *Shortname*: it is an unique name assigned to the tab, used to refer to it in the other sheets and configuration
  - *Label*: it is the human readable name of the tab or the i18n key
  - *Priority*: the tabs are sorted by priority in ascending order
  - *Leading*: It can be y (yes) or n (no). If set to yes the tab is shown on the top of the item's page and remains there even if the user browse the other tabs
  - *Security*: it defines who has access to the tab
- **box**. It contains the details about all the boxes that need to be created. **The order in which they are listed will be respected** for each entity type in the UI.
  - *Entity*: it is the label of the Entity Type to which the box belong. A box's shortname must be unique for a specific entity
  - *Collapsed*. It can be y (yes) or n (no). When collapsed is y the box is shown as a closed panel in the default theme
  - *Container*. It can be y (yes) or n (no). When container is y the outline of the box is shown, including the section at the top with the name.
  - *Type*. It is the box's type available options are listed in the *utilsdata* sheet
  - *Shortname*: it is an unique name assigned to the box, used to refer to it in the other sheets and configuration
  - *Label*: it is the human readable name of the box or the i18n key
  - *Minor*. It can be y (yes) or n (no). Minor box are not used to determine if a tab actually has content to show to the user.
  - *Security*: it defines who has access to the box
  - *Style*. It defines extra information that can be used by the UI to personalize the view. In the default theme the style is added as extra CSS classes to the box panel and can be used to bind the box to the bootstrap grid (i.e. col-md-6 to size the box to half page)
- **tab2box**. It links box to a specific tab.
  - *Entity* and *Tab* colums are used to lookup in the tab sheet (Tab  Shortname).
  - *Entity* and *Boxes* colums are used to lookup in the box sheet (Box  Shortname). The *Boxes* column is a comma separated list of boxes name.
  - *Row*: The row of the tab where place the given boxes. If the same row is specified for the same tab, the boxes are placed in a new cell with the given cell style
  - *Row_style*: The row's style where the boxes are placed. Same row must have the same style (or at least empty)
  - *Cell_style*: The cell's style where the boxes are placed in the given row.
- **box2metadata**. It provides extra configuration details for metadata box such as the list of metadata (fields) included.
  - *Entity* and *Box* colums are used to lookup in the box sheet (Box  Shortname).
  - *Row*. The number of the row where the field will be added.
  - *Cell*. The number of the cell where the field will be added.
  - *Row_style*. The style of the row where the field will be added.
  - *Cell_style*. The style of the cell where the field will be added.
  - *FieldType*. The field's type such as METADATA or BITSTREAM or METADATAGROUP
  - *Metadata*. The metadata key of the field
  - *Value*. For bitstream field only, limit the bitstream to list to the one where the metadata has such value
  - *Bundle*. Required for bitstream field only, limit the bitstream to list to the one that belong to a bundle with the specified name
  - *Label*: it is the human readable name of the field or the i18n key
  - *Rendering*. A custom display strategy to apply to the field
  - *Style*. It defines extra information that can be used by the UI to personalize the view. In the default theme the style is added as extra CSS classes to the field container.
  - *Style_label:* It defines extra information that can be used by the UI to personalize the view. In the default theme the style is added as extra CSS classes to the label.
  - *Style_value:* It defines extra information that can be used by the UI to personalize the view. In the default theme the style is added as extra CSS classes to the value.
  - *Label_as_heading*: It can be y (yes) or n (no). If yes the label is shown above the value rather than to the left
  - *Values_inline*: It can be y (yes) or n (no). If yes the values of the same metadata field are shown inline, if no are shown in column
- **metadatagroups** It provides extra configuration details for nested metadata that should be visualized as a group.
  - *Entity* : it is the label of the Entity Type to which the neted metdata belong.
  - *Parent*: The name of main metadata which all the nested metdata are related to. This metadata must be present also in the **box2 metadata** tab with type METADATAGROUP
  - *FieldType*. The field's type must be always METADATA.
  - *Metadata*. The metadata key of the field

    - *Value*. For bitstream field only, limit the bitstream to list to the one where the metadata has such value
    - *Bundle*. Required for bitstream field only, limit the bitstream to list to the one that belong to a bundle with the specified name
    - *Label*: it is the human readable name of the field or the i18n key
    - *Rendering*. A custom display strategy to apply to the field
    - *Style_label:* It defines extra information that can be used by the UI to personalize the view. In the default theme the style is added as extra CSS classes to the label.
    - *Style_value:* It defines extra information that can be used by the UI to personalize the view. In the default theme the style is added as extra CSS classes to the value.
- **box2metrics** It provides configuration details for box displaying metrics.
    - *Entity* : it is the label of the Entity Type to which the neted metdata belong.
    - *Box*: name of the box, as defined in “Box” sheet.
    - *Metric_type*. List of metrics to be displayed in the box. Currently supported metrics are: scopus-author-h-index, scopus-author-coauthor-count, scopus-author-cited-count, scopus-author-citation-count, scopus-author-document-count, wosPersonCitation, view, google-scholar, altmetrics (only for publications), dimensions (only for publications), download (only publications)
- **tabpolicy** and **boxpolicy**. They provide extra information for tab and box configured to have custom security policy (*security* = `CUSTOM DATA`)
    - *Entity* and *Shortname* colums are used to lookup in the tab and box sheet
    - *Metadata*. It contains the metadata key in the format *schema*.*element*[.*qualifier*] (i.e. cris.policy.group) that holds the information about which groups or users can access the linked tab or box
- **utilsdata**. The sheet is not really used by the tool. It is intended to help with the filling out of the excel
- **tab_i18n, box_i18n, metadata_i18n, metadatagroup_i18n**: see last paragraph

The tool performs the following actions in subsequent order stopping in case of failure

1. **validate** the excel configuration file reporting any identified inconsistency such as, undefined entity types, metadata, missing tabs or boxes referenced in the association sheets, etc.
2. completely **wipe** the current layout configuration
3. **load** the configuration

#### Available rendering strategies

Current out of the box rendering strategies are:

heading
text
longtext
link
link.label
date
identifier
crisref
thumbnail
attachment
tag

#### I18n key conventions

When file reports i18n keys references to be used as labels, following conventions are followed

| Type of label | Prefix | Example (to be added to translation files) |
|---|---|---|
| Tab | `layout.tab.header.tabShortName` | `"layout.tab.header.details": "Details",` |
| Box | `layout.box.header.boxShortName` | `"layout.box.header.altmetrics": "Alternative Metrics",` |
| Metadata / Metadatagroups | `layout.field.label.EntitType.Metadata` | `"layout.field.label.Equipment.dc.type": "Cost Type",` |

Utility sheets to generate keys to be added to internationalization files (en.json etc...):

`tab_i18n`, `box_i18n`, `metadata_i18n`, `metadatagroup_i18n`

are available on `cris-layout-configuration.xls` file

whith following formulas used (from B2 cell):

| Sheet | Formula |
|---|---|
| `tab_i18n` | ="""""&$A$2&$tab.$B2&""""": """"&$tab.$C2&"""","  |
| `box_i18n` | ="""""&$A$2&$box.$D2&""""": """"&$box.$E2&"""","  |

| | |
|---|---|
| `metadata_i18n` | ="""""&$A$2&$box2metadata.$A2&"."&$box2metadata.$F2&""""": """"&$box2metadata.$I2&"""","" |
| `metadatagroup_i18n` | ="""""&$A$2&$metadatagroups.$A2&"."&$metadatagroups.$D2&""""": """"&$metadatagroups.$G2&"""","" |

### Metadata security configuration

It is possible to make Item submitter or item editor able to define, for a given set of metadata, the accessibility level of its value(s).

When a security level is set, a metadata value will be available on both front end (entity layout) and back end (REST) if and only if requiring user matches with defined security.

#### Configuration

Security levels are represented by integer values, metadata security is based on content of "`metadata-security.cfg`" , where key value pairs define possible security levels for each metadata. Security levels are represented as arrays.

```
metadatavalue.visibility.settings = [0 1 2]
metadatavalue.visibility.Person.settings = [0 1]
metadatavalue.visibility.Person.dc.date.available.settings = [0 1]
metadatavalue.visibility.Person.dc.description.provenance.settings =
[0 1 2]
metadatavalue.visibility.Person.creativework.datePublished.settings =
[1 2]
metadatavalue.visibility.Person.creativework.publisher.settings = [0 1]
metadatavalue.visibility.Person.cris.author.scopus-author-id.settings
=  [1 2]
metadatavalue.visibility.Person.cris.identifier.gscholar.settings = [0
1 2]
metadatavalue.visibility.Person.cris.workflow.name.settings =  [1]
```

Keys template is as follows

`metadatavalue.visibility.<EntityType>.<MetadataValue>.settings`

Configuration lookup follows a fallback logic, if for a given metadata, no value is defined, default metadata security settings for the entity (`metadatavalue.visibility.<EntityType>.settings`) applies. If neither at entity level metadata security is defined, default `metadatavalue.visibility.settings` field value is used.

A `null` value means that for a metadata, or its fallback, no security rules must be proposed to the submitter or to the editor.

In the above example, [0 1 2] means that for a metadata, or its fallbacks, three security levels can be defined, level 0, level 1 and level 2, [1 2] means only levels 1 and 2 are available, etc.

If only one value greater than 0 is defined, the choice proposed for this metadata security will be between level 0 and the level defined in configuration.

#### Security evaluation

Metadata security check is performed, after tabs and boxes layout security checks, this means that a metadata value, to be included into a response has to be part of a tab and a box in DSpace-CRIS 7 layout definition. Once part of layout, its security, if any, is evaluated after the one of the tab and box such metadata belongs to. In case of layout security and metadata security of different levels, the most restricted security criteria is applied. For example, a "Public" metadata value is returned only if its containing tab and box have a PUBLIC security, if layout security is not PUBLIC, this layout security will be applied:

| Tab security | Box Security | Metadata security | Metadata visible? |
|---|---|---|---|
| PUBLIC | PUBLIC | Public | YES |
| PUBLIC | PUBLIC | level > Public | Depending on metadata security level |
| level > PUBLIC | PUBLIC | Public | Only to users allowed to see tab content |
| PUBLIC | level > PUBLIC | Public | |

| | | | |
|---|---|---|---|
| | | | Only to users allowed to see box content |
| level > PUBLIC | level > PUBLIC | Public | Only users having the most restricted access criteria between tab and box security rules |
| level > PUBLIC | level > PUBLIC | level > Public | Most restricted rule is applied |

Once a security level is set, security of a metadata is evaluated by implementations of the `org.dspace.content.service.MetadataSecurityEvaluation` interface. Mapping is defined in `dspace/config/spring/api/spring-dspace-security-metadata.xml` spring configuration file. In this file, each level number is mapped with an implementation of `org.dspace.content.service.MetadataSecurityEvaluation` interface

#### Default implementation

Out of the box, DSpace-CRIS7 has 3 different security levels:

- **level 0 (Public)**: metadata value is available to all users, even Anonymous in case entity is available to them
- **level 1 (Trusted)**: metadata value is available only to logged in users members of a defined group, named “Trusted”, as a prerequisite, this group must be available in DSpace-CRIS 7 installation
- **level 2 (Admin and Owner)**: metadata value is available only to users belonging to the “Administrators” group or to the owner of the DSpace-CRIS 7 entity.

Default Spring configuration file content, driving above described security is here reported, for level 1 it is defined the name of the egroup which members are considered users trusted to see metadata protected by a level 1 security.

```
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:util="http://www.springframework.org/schema/util"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-2.5.
xsd
           http://www.springframework.org/schema/util
           http://www.springframework.org/schema/util/spring-util.xsd">
    <util:map id="securityLevelsMap">
        <entry key="0" value-ref="level0Security"/>
        <entry key="1" value-ref="level1Security"/>
        <entry key="2" value-ref="level2Security"/>
    </util:map>
    <bean id ="level0Security" name="level0Security" class="org.dspace.
content.MetadataPublicAccess"/>
    <bean id ="level1Security" name="level1Security" class="org.dspace.
content.MetadataGroupBasedAccess">
        <property name="egroup" value="Trusted"/>
    </bean>
    <bean  id ="level2Security" name="level2Security" class="org.dspace.
content.MetadataAdministratorAndOwnerAccess"/>
</beans>
```

#### Custom implementation

Existing logic can be extended, and many custom security levels, with their evaluation logic, can be defined.

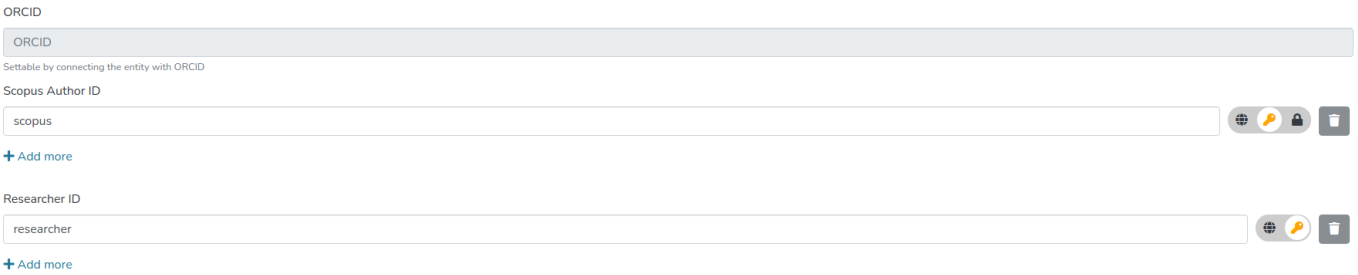
Needed steps to reach this goal are:

- Implement `org.dspace.content.service.MetadataSecurityEvaluation` interface
- Map above implementation with its level in `dspace/config/spring/api/spring-dspace-security-metadata.xml` file

#### Front end

From the submitter and editor perspective, metadata security configuration appears as a toggle placed beside the input field used to collect metadata value in submission form, edit in submission mode, edit in admin mode sections.

Security can be defined while editing or submitting:




or while editing item metadata in admin mode:6


Settings needed to define for which metadata field(s) (if any) security choice should be proposed to submitter or editor, REST endpoint `{dspace.rest.url}/api/core/securitysettings/` is used. Contract of this endpoint defined at https://github.com/4Science/Rest7Contract/blob/dspace-cris-7/securitysettings-endpoint.md .

Security levels are represented by icons, icons can be customized (we suggest to use fontawesome icons), as well as the background color to be used when they are selected, by mean of the `security` section in `environment.ts` file

```
security: {
  levels: [
    {
      value: 0,
      icon: 'fa fa-globe',
      color: 'green'
    },
    {
      value: 1,
      icon: 'fa fa-key',
      color: 'orange'
    },
    {
      value: 2,
      icon: 'fa fa-lock',
      color: 'red'
    }
  ]
}
```

Above mapping defines for every default DSpace-CRIS 7 metadata security level, which icon, based on fontawesome, will represent this level, and will have to be selected by the submitter or by the editor, and which background color will this icon have once selected.

Level 0 is represented by the globe, a green globe means that metadata to which is referred to has level 0 security policy, Level 1 is represented by a key (orange when selected), and level 2 is represented by a lock (red when selected).

## Schedule periodic execution of scripts

DSpace-API contains scripts JAVA that perform maintenance operation or batch manipulation of the platform content.

They can be executed from the installation folder `/bin` using the special `dspace` command

The `dspace` command accepts as parameter the name of the script to execute as configured in the `config/launcher.xml` file.

The following features are based on script that should be run periodically via CRONTAB

- Bibliometrics, see
  - Scopus Metrics
  - H-Index Metrics
  - WOS (Web of Science) Metrics
- Usage data as metrics, see Usage statistics data generators
- Automatic retrieval of Scopus Publication, see Scanning Scopus for additional publications in profiles
- Automatic retrieval of Web of Science Publication, see Scanning WOS (Web of Science) for additional publications in profiles
- ORCID Push, see ORCID Synchronization

**Scopus Metrics**

Scopus metrics collection for publications and researchers is handled by the process hereafter described. To have connection with Scopus properly working, following configuration properties must be set

```
metrics.scopus.citation-count.url = <scopus query url>
metrics.scopus.citation-count.apiKey = <scopus api key>
metrics.scopus.citation-count.instToken = <scopus instToken>
```

If last two properties are the same used to import publications from scopus, its value can be “inherited” from ${scopus.apiKey} and ${scopus.instToken} properties already set, in this way

```
metrics.scopus.citation-count.apiKey = ${scopus.apiKey}
metrics.scopus.citation-count.instToken = ${scopus.instToken}
```

The updating of publication and researchers metrics from WOS (Web of Science) service is controlled by two processes, which can be run from both processes section and Command Line Interface:

| | |
|---|---|
| ***update-metrics scopus*** | *updates metrics of type entityType=Publication* |
| **update-metrics scopus-person** | *updates metrics of type entityType=Person* |

where:
**update-metrics** - is the name of the script
**scopus** or **scopus-person** - is the name of the service, in case “-person” suffix is present, metrics regarding researchers will be collected, if no suffix is added, process will collect research output metrics

The script applies the following steps to perform the update:

1. performs a global search to retrieve all entities of type *Publication* and have one of the following metadata set: dc.identifier.doi or dc.identifier.pmid values.
2. taking one item at a time - extracts the metadata values such as : (dc.identifier.doi , dc.identifier.pmid and dc.identifier.scopus) with these values it constructs the query to be sent to the external Scopus service which in turn returns the document containing the metric.

a generic answer from scopus can be:

Fai clic qui per espandere...

*<?xml version="1.0" encoding="UTF-8"?>*

<search-results xmlns="http://www.w3.org/2005/Atom" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/" xmlns:prism="http://prismstandard.org/namespaces/basic/2.0/" xmlns:atom="http://www.w3.org/2005/Atom">

<opensearch:totalResults>1</opensearch:totalResults>

<opensearch:startIndex>0</opensearch:startIndex>

<opensearch:itemsPerPage>1</opensearch:itemsPerPage>

<opensearch:Query role="request" searchTerms="DOI(10.1016/j.gene.2009.04.019)" startPage="0"/>

<link ref="self" href="https://api.elsevier.com/content/search/scopus?start=0&count=25&query=DOI%2810.1016%2Fj.gene.2009.04.019%29" type="application/xml"/>

<link ref="first" href="https://api.elsevier.com/content/search/scopus?start=0&count=25&query=DOI%2810.1016%2Fj.gene.2009.04.019%29" type="application/xml"/>

<entry>

<link ref="self" href="https://api.elsevier.com/content/abstract/scopus_id/67349162500"/>

<link ref="author-affiliation" href="https://api.elsevier.com/content/abstract/scopus_id/67349162500?field=author,affiliation"/>

<link ref="scopus" href="https://www.scopus.com/inward/record.uri?partnerID=HzOxMe3b&scp=67349162500&origin=inward" />

<link ref="scopus-citedby" href="https://www.scopus.com/inward/citedby.uri?partnerID=HzOxMe3b&scp=67349162500&origin=inward"/>

<link ref="full-text" href="https://api.elsevier.com/content/article/eid/1-s2.0-S0378111909001929"/>

<prism:url>https://api.elsevier.com/content/abstract/scopus_id/67349162500</prism:url>

<dc:identifier>SCOPUS_ID:67349162500</dc:identifier>

<eid>2-s2.0-67349162500</eid>

<dc:title>Transcriptomic response of Argopecten purpuratus post-larvae to copper exposure under experimental conditions</dc:title>

<dc:creator>Zapata M.</dc:creator>

<prism:publicationName>Gene</prism:publicationName>

<prism:issn>03781119</prism:issn>

<prism:volume>442</prism:volume>

<prism:issueIdentifier>1-2</prism:issueIdentifier>

<prism:pageRange>37-46</prism:pageRange>

<prism:coverDate>2009-08-01</prism:coverDate>

<prism:coverDisplayDate>1 August 2009</prism:coverDisplayDate>

<prism:doi>10.1016/j.gene.2009.04.019</prism:doi>

<pii>S0378111909001929</pii>

<citedby-count>44</citedby-count>

<affiliation>

<affilname>Institut Universitaire Européen de la Mer (IUEM)</affilname>

<affiliation-city>Plouzane</affiliation-city>

<affiliation-country>France</affiliation-country>

</affiliation>

<affiliation>

<affilname>Universidad de Antofagasta</affilname>

<affiliation-city>Antofagasta</affiliation-city>

<affiliation-country>Chile</affiliation-country>

</affiliation>

<pubmed-id>19406218</pubmed-id>

<prism:aggregationType>Journal</prism:aggregationType>

<subtype>ar</subtype>

<subtypeDescription>Article</subtypeDescription>

<source-id>15636</source-id>

<openaccess>0</openaccess>

<openaccessFlag>false</openaccessFlag>

</entry>

</search-results>

---

3. In the next step, a new metric is created with the data retrieved from scopus, such as:

| | |
|---|---|
| MetricType | ***scopusCitation*** |
| Last | ***true*** |
| MetricCount | ***44*** (value contained in the tag `<citedby-count>`) |
| AcquisitionDate | **date** on which the metric was recorded |
| Remark | is a more complex field, which contains 4 values if there are any in the response<br>***identifier*** : 2-s2.0-67349162500 (value contained in the tag *<eid>*)<br>**link** : https://www.scopus.com/inward/citedby.uri?.. (value contained in the tag *<link ref="scopus-citedby"* )<br>**pmid** : 19406218 (value contained in the tag *<pubmed-id>*)<br>**doi** : 10.1016/j.gene.2009.04.019 (value contained in the tag *<prism:doi>*) |

In case a metric for this Publication was already on db, it is not overwritten, but its 'Last' flag is set to 'false'.

**H-Index Metrics**

The update of the h-index metrics from Scopus is controlled by a script that must be executed using the following command: ***update-metrics hindex <p>***

where:
**update-metrics** - is the name of the script
**hindex** - is the name of the service

**<p>** - is the type of metric, which can take the following values :

| |
|---|
| *scopus-author-h-index* |
| *scopus-author-cited-count* |
| *scopus-author-document-count* |
| *scopus-author-citation-count* |
| *scopus-author-coauthor-count* |

The script applies the following steps to perform the update:

1. performs a global search to retrieve all entities of type *Person* and have the following metadata set: person.identifier.scopus-author-id
2. taking one item at a time - extracts the metadata values such as : (person.identifier.scopus-author-id) with these values it constructs the query to be sent to the external Scopus service which in turn returns the document containing the metrics.

a generic answer from scopus can be:

Fai clic qui per espandere...


```
{
"author-retrieval-response": [
{
"@status": "found",
"@_fa": "true",
"coredata": {
"prism:url": "http://api.elsevier.com/content/author/author_id/7406754790",
"dc:identifier": "AUTHOR_ID:7406754790",
"superseding-identifier": [
{
"@_fa": "true",
"$": "author_id:35354080500"
}
],
"historical-identifier": [
{
"@_fa": "true",
"$": "author_id:16163784900"
},
{
"@_fa": "true",
"$": "author_id:16164338100"
},
{
"@_fa": "true",
"$": "author_id:16182403800"
},
{
"@_fa": "true",
"$": "author_id:19236027800"
},
{
"@_fa": "true",
"$": "author_id:48161681000"
},
{
"@_fa": "true",
"$": "author_id:7406748482"
},
{
"@_fa": "true",
"$": "author_id:7406751469"
},
{
"@_fa": "true",
"$": "author_id:7406756670"
},
{
"@_fa": "true",
"$": "author_id:7406758558"
},
{
"@_fa": "true",
"$": "author_id:7406759079"
},
{
"@_fa": "true",
"$": "author_id:7406761472"
},
{
"@_fa": "true",
"$": "author_id:7406756821"
},
{
"@_fa": "true",
"$": "author_id:7406758337"
},
{
"@_fa": "true",
"$": "author_id:7406757377"
},
```

```
{
"@_fa": "true",
"$": "author_id:24430291100"
},
{
"@_fa": "true",
"$": "author_id:7406760222"
},
{
"@_fa": "true",
"$": "author_id:7406751117"
},
{
"@_fa": "true",
"$": "author_id:7406754546"
},
{
"@_fa": "true",
"$": "author_id:7406758522"
},
{
"@_fa": "true",
"$": "author_id:7406757840"
},
{
"@_fa": "true",
"$": "author_id:7406760968"
},
{
"@_fa": "true",
"$": "author_id:7406761819"
},
{
"@_fa": "true",
"$": "author_id:24792320900"
},
{
"@_fa": "true",
"$": "author_id:7406754585"
},
{
"@_fa": "true",
"$": "author_id:7406761420"
},
{
"@_fa": "true",
"$": "author_id:7406760476"
},
{
"@_fa": "true",
"$": "author_id:7406760420"
},
{
"@_fa": "true",
"$": "author_id:7406748135"
},
{
"@_fa": "true",
"$": "author_id:7406748141"
},
{
"@_fa": "true",
"$": "author_id:7406756467"
},
{
"@_fa": "true",
"$": "author_id:24792414100"
},
{
"@_fa": "true",
"$": "author_id:7406755698"
},
```

```
{
"@_fa": "true",
"$": "author_id:7406757200"
},
{
"@_fa": "true",
"$": "author_id:7406752451"
},
{
"@_fa": "true",
"$": "author_id:35343787700"
},
{
"@_fa": "true",
"$": "author_id:7406755983"
},
{
"@_fa": "true",
"$": "author_id:7406747309"
},
{
"@_fa": "true",
"$": "author_id:7406746824"
},
{
"@_fa": "true",
"$": "author_id:35288841700"
},
{
"@_fa": "true",
"$": "author_id:8886334200"
},
{
"@_fa": "true",
"$": "author_id:7406748683"
},
{
"@_fa": "true",
"$": "author_id:7406752882"
},
{
"@_fa": "true",
"$": "author_id:16186817800"
},
{
"@_fa": "true",
"$": "author_id:7406760300"
},
{
"@_fa": "true",
"$": "author_id:7406751796"
},
{
"@_fa": "true",
"$": "author_id:7406751610"
},
{
"@_fa": "true",
"$": "author_id:7406758250"
},
{
"@_fa": "true",
"$": "author_id:7406756361"
},
{
"@_fa": "true",
"$": "author_id:7406758368"
},
{
"@_fa": "true",
"$": "author_id:7406754722"
},
```

```
{
"@_fa": "true",
"$": "author_id:7406761233"
},
{
"@_fa": "true",
"$": "author_id:7406758186"
},
{
"@_fa": "true",
"$": "author_id:55223149100"
},
{
"@_fa": "true",
"$": "author_id:7406751745"
},
{
"@_fa": "true",
"$": "author_id:7406757599"
},
{
"@_fa": "true",
"$": "author_id:7406750482"
},
{
"@_fa": "true",
"$": "author_id:8084650900"
},
{
"@_fa": "true",
"$": "author_id:36512400300"
},
{
"@_fa": "true",
"$": "author_id:7406756266"
},
{
"@_fa": "true",
"$": "author_id:35481944600"
},
{
"@_fa": "true",
"$": "author_id:55740287500"
},
{
"@_fa": "true",
"$": "author_id:49962180600"
},
{
"@_fa": "true",
"$": "author_id:55498586000"
},
{
"@_fa": "true",
"$": "author_id:55567933800"
},
{
"@_fa": "true",
"$": "author_id:55740335800"
},
{
"@_fa": "true",
"$": "author_id:56718787100"
},
{
"@_fa": "true",
"$": "author_id:56125651000"
},
{
"@_fa": "true",
"$": "author_id:56757379200"
},
```

```
{
"@_fa": "true",
"$": "author_id:56128206000"
},
{
"@_fa": "true",
"$": "author_id:56751326100"
},
{
"@_fa": "true",
"$": "author_id:7406749093"
},
{
"@_fa": "true",
"$": "author_id:55740275200"
},
{
"@_fa": "true",
"$": "author_id:57202829776"
},
{
"@_fa": "true",
"$": "author_id:57207896310"
},
{
"@_fa": "true",
"$": "author_id:55865075800"
},
{
"@_fa": "true",
"$": "author_id:57207907090"
},
{
"@_fa": "true",
"$": "author_id:22977424200"
},
{
"@_fa": "true",
"$": "author_id:6701331881"
},
{
"@_fa": "true",
"$": "author_id:57202749922"
},
{
"@_fa": "true",
"$": "author_id:7409563964"
},
{
"@_fa": "true",
"$": "author_id:57208122810"
},
{
"@_fa": "true",
"$": "author_id:57215806262"
}
],
"eid": "9-s2.0-7406754790",
"orcid": "0000-0002-4568-3015",
"document-count": "249",
"cited-by-count": "16677",
"citation-count": "23324",
"link": [
{
"@href": "https://www.scopus.com/authid/detail.uri?partnerID=HzOxMe3b&authorId=7406754790&origin=inward",
"@rel": "scopus-author",
"@_fa": "true"
},
{
"@href": "http://api.elsevier.com/content/author/author_id/7406754790",
"@rel": "self",
"@_fa": "true"
```

```
},
{
"@href": "http://api.elsevier.com/content/search/scopus?query=au-id(7406754790)",
"@rel": "search",
"@_fa": "true"
},
{
"@href": "http://api.elsevier.com/content/search/author?co-author=7406754790",
"@rel": "coauthor-search",
"@_fa": "true"
}
]
},
"h-index": "68",
"coauthor-count": "15468",
"affiliation-current": {
"@id": "60017592",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60017592"
},
"affiliation-history": {
"affiliation": [
{
"@_fa": "true",
"@id": "60000221",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60000221"
},
{
"@_fa": "true",
"@id": "60121115",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60121115"
},
{
"@_fa": "true",
"@id": "60014439",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60014439"
},
{
"@_fa": "true",
"@id": "60032897",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60032897"
},
{
"@_fa": "true",
"@id": "60020304",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60020304"
},
{
"@_fa": "true",
"@id": "60019778",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60019778"
},
{
"@_fa": "true",
"@id": "60007776",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60007776"
},
{
"@_fa": "true",
"@id": "60026175",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60026175"
},
{
"@_fa": "true",
"@id": "60005248",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60005248"
},
{
"@_fa": "true",
"@id": "60010537",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60010537"
},
{
```

"@_fa": "true",
"@id": "60020623",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60020623"
},
{
"@_fa": "true",
"@id": "60020661",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60020661"
},
{
"@_fa": "true",
"@id": "60011418",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60011418"
},
{
"@_fa": "true",
"@id": "60013791",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60013791"
},
{
"@_fa": "true",
"@id": "60025778",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60025778"
},
{
"@_fa": "true",
"@id": "60031581",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60031581"
},
{
"@_fa": "true",
"@id": "60021121",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60021121"
},
{
"@_fa": "true",
"@id": "60024941",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60024941"
},
{
"@_fa": "true",
"@id": "60025590",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60025590"
},
{
"@_fa": "true",
"@id": "60009982",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60009982"
},
{
"@_fa": "true",
"@id": "60016340",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60016340"
},
{
"@_fa": "true",
"@id": "60003269",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60003269"
},
{
"@_fa": "true",
"@id": "60025038",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60025038"
},
{
"@_fa": "true",
"@id": "60000745",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60000745"
},
{
"@_fa": "true",
"@id": "60030612",

"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60030612"
},
{
"@_fa": "true",
"@id": "60030635",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60030635"
},
{
"@_fa": "true",
"@id": "60075450",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/60075450"
},
{
"@_fa": "true",
"@id": "105951559",
"@href": "http://api.elsevier.com/content/affiliation/affiliation_id/105951559"
}
]
},
"subject-areas": {
"subject-area": [
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3100",
"$": "Physics and Astronomy (all)"
},
{
"@_fa": "true",
"@abbrev": "ENGI",
"@code": "2208",
"$": "Electrical and Electronic Engineering"
},
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3101",
"$": "Physics and Astronomy (miscellaneous)"
},
{
"@_fa": "true",
"@abbrev": "MULT",
"@code": "1000",
"$": "Multidisciplinary"
},
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3107",
"$": "Atomic and Molecular Physics, and Optics"
},
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3105",
"$": "Instrumentation"
},
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3106",
"$": "Nuclear and High Energy Physics"
},
{
"@_fa": "true",
"@abbrev": "ENGI",
"@code": "2201",
"$": "Engineering (miscellaneous)"
},
{
"@_fa": "true",
"@abbrev": "MATH",

```
"@code": "2610",
"$": "Mathematical Physics"
},
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3103",
"$": "Astronomy and Astrophysics"
},
{
"@_fa": "true",
"@abbrev": "ENER",
"@code": "2104",
"$": "Nuclear Energy and Engineering"
},
{
"@_fa": "true",
"@abbrev": "BIOC",
"@code": "1300",
"$": "Biochemistry, Genetics and Molecular Biology (all)"
},
{
"@_fa": "true",
"@abbrev": "ARTS",
"@code": "1207",
"$": "History and Philosophy of Science"
},
{
"@_fa": "true",
"@abbrev": "EART",
"@code": "1912",
"$": "Space and Planetary Science"
},
{
"@_fa": "true",
"@abbrev": "PHYS",
"@code": "3104",
"$": "Condensed Matter Physics"
}
]
},
"author-profile": {
"status": "update",
"date-created": {
"@day": "02",
"@month": "12",
"@year": "2005"
},
"alias": {
"@current-status": "moved-from",
"aliased-id": [
{
"@status": "moved-from",
"@timestamp": "2020-12-07T07:36:25.000025-05:00",
"$": "57215806262"
},
{
"@status": "moved-into",
"@timestamp": "2020-12-07T07:36:25.000025-05:00",
"$": "35354080500"
}
]
},
"preferred-name": {
"@date-locked": "2020-02-12T09:34:24.476-05:00",
"@source": "corrapi-external",
"initials": "A.J.",
"indexed-name": "Smith A.",
"surname": "Smith",
"given-name": "A. J.Stewart"
},
"name-variant": [
```

```
{
"@doc-count": "1",
"@source": "internal-ani",
"initials": "A.",
"indexed-name": "Smith A.",
"surname": "Smith",
"given-name": "A."
},
{
"@doc-count": "7",
"@source": "internal-ani",
"initials": "A.M.",
"indexed-name": "Smith A.",
"surname": "Smith",
"given-name": "A. M."
},
{
"@doc-count": "5",
"@source": "internal-ani",
"initials": "A.J.",
"indexed-name": "Smith A.",
"surname": "Smith",
"given-name": "A. J."
},
{
"@doc-count": "145",
"@source": "internal-ani",
"initials": "J.",
"indexed-name": "Smith J.",
"surname": "Smith",
"given-name": "J."
},
{
"@doc-count": "1",
"@source": "internal-ani",
"initials": "A.J.S.",
"indexed-name": "SMITH A.",
"surname": "SMITH",
"given-name": "A. J.S."
},
{
"@doc-count": "209",
"@source": "internal-ani",
"initials": "J.G.",
"indexed-name": "Smith J.",
"surname": "Smith",
"given-name": "J. G."
},
{
"@doc-count": "46",
"@source": "internal-ani",
"initials": "A.J.S.",
"indexed-name": "Smith A.",
"surname": "Smith",
"given-name": "A. J.S."
}
],
"classificationgroup": {
"classifications": {
"@type": "ASJC",
"classification": [
{
"@frequency": "93",
"$": "3100"
},
{
"@frequency": "3",
"$": "2208"
},
{
"@frequency": "37",
"$": "3101"
```

```
},
{
"@frequency": "2",
"$": "1000"
},
{
"@frequency": "2",
"$": "3107"
},
{
"@frequency": "13",
"$": "3105"
},
{
"@frequency": "104",
"$": "3106"
},
{
"@frequency": "8",
"$": "2201"
},
{
"@frequency": "6",
"$": "2610"
},
{
"@frequency": "2",
"$": "3103"
},
{
"@frequency": "2",
"$": "2104"
},
{
"@frequency": "1",
"$": "1300"
},
{
"@frequency": "1",
"$": "1207"
},
{
"@frequency": "1",
"$": "1912"
},
{
"@frequency": "1",
"$": "3104"
}
]
}
},
"publication-range": {
"@end": "2019",
"@start": "1957"
},
"affiliation-current": {
"affiliation": {
"@affiliation-id": "60017592",
"@source": "corrapi-external",
"ip-doc": {
"@id": "60017592",
"@type": "parent",
"@relationship": "author",
"afnameid": "Carleton University#60017592",
"afdispname": "Carleton University",
"manual-curation": {
"@curated": "true",
"date-curation": {
"@day": "01",
"@month": "12",
"@timestamp": "2016-12-01T12:07:38.373-05:00",
```

```
"@year": "2016"
},
"curation-source": "QABOAPI",
"curation-type": "QABO-2657529"
},
"preferred-name": {
"@source": "internal-ani",
"$": "Carleton University"
},
"sort-name": "Carleton University",
"address": {
"@country": "can",
"address-part": "1125 Colonel By Drive",
"city": "Ottawa",
"state": "ON",
"postal-code": "K1S 5B6",
"country": "Canada"
},
"org-domain": "carleton.ca",
"org-URL": "http://www.carleton.ca"
}
}
},
"affiliation-history": {
"affiliation": [
{
"@affiliation-id": "60000221",
"@source": "internal-ani",
"ip-doc": {
"@id": "60000221",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of Colorado Boulder#60000221",
"afdispname": "University of Colorado Boulder",
"preferred-name": {
"@source": "internal-ani",
"$": "University of Colorado Boulder"
},
"sort-name": "Colorado Boulder, University of",
"address": {
"@country": "usa",
"address-part": "20 UCB",
"city": "Boulder",
"state": "CO",
"postal-code": "80309-0001",
"country": "United States"
},
"org-domain": "colorado.edu",
"org-URL": "http://www.colorado.edu/"
}
},
{
"@affiliation-id": "60121115",
"@source": "internal-ani",
"ip-doc": {
"@id": "60121115",
"@type": "parent",
"@relationship": "author",
"afnameid": "Department of Physics#60121115",
"afdispname": "Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"sort-name": "Department of Physics",
"address": {
"@country": "gbr",
"address-part": "19 J J Thomson Avenue",
"city": "Cambridge",
"state": "Cambridgeshire",
"postal-code": "CB3 0HE",
"country": "United Kingdom"
```

},
"org-domain": "phy.cam.ac.uk",
"org-URL": "https://www.phy.cam.ac.uk/"
}
},
{
"@affiliation-id": "60014439",
"@source": "internal-ani",
"ip-doc": {
"@id": "60014439",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of California, Davis#60014439",
"afdispname": "University of California, Davis",
"manual-curation": {
"@curated": "true",
"date-curation": {
"@day": "15",
"@month": "03",
"@timestamp": "2018-03-15T17:17:12.672-04:00",
"@year": "2018"
},
"curation-source": "QABOAPI",
"curation-type": "QABO-3777397"
},
"preferred-name": {
"@source": "internal-ani",
"$": "University of California, Davis"
},
"sort-name": "California, Davis, University of",
"address": {
"@country": "usa",
"address-part": "One Shields Avenue",
"city": "Davis",
"state": "CA",
"postal-code": "95616-5270",
"country": "United States"
},
"org-domain": "ucdavis.edu",
"org-URL": "https://www.ucdavis.edu/"
}
},
{
"@affiliation-id": "60032897",
"@source": "internal-ani",
"ip-doc": {
"@id": "60032897",
"@type": "parent",
"@relationship": "author",
"afnameid": "Universidad de Sonora#60032897",
"afdispname": "Universidad de Sonora",
"preferred-name": {
"@source": "internal-ani",
"$": "Universidad de Sonora"
},
"sort-name": "Sonora, University of",
"address": {
"@country": "mex",
"address-part": "Calle Rosales y Boulevard Luis Encinas",
"city": "Hermosillo",
"state": "SON",
"postal-code": "83000",
"country": "Mexico"
},
"org-domain": "uson.mx",
"org-URL": "http://www.uson.mx"
}
},
{
"@affiliation-id": "60020304",
"@source": "internal-ani",
"ip-doc": {

"@id": "60020304",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of Maryland#60020304",
"afdispname": "University of Maryland",
"preferred-name": {
"@source": "internal-ani",
"$": "University of Maryland"
},
"sort-name": "Maryland, University of",
"address": {
"@country": "usa",
"city": "College Park",
"state": "MD",
"postal-code": "20742",
"country": "United States"
},
"org-domain": "umd.edu",
"org-URL": "https://umd.edu/"
}
},
{
"@affiliation-id": "112856084",
"@parent": "60020304",
"@source": "internal-ani",
"ip-doc": {
"@id": "112856084",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of Maryland, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Maryland"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"city": "College Park",
"state": "MD",
"postal-code": "20742",
"country": "United States"
},
"org-domain": "umd.edu",
"org-URL": "https://umd.edu/"
}
},
{
"@affiliation-id": "60019778",
"@source": "internal-ani",
"ip-doc": {
"@id": "60019778",
"@type": "parent",
"@relationship": "author",
"afnameid": "European Organization for Nuclear Research#60019778",
"afdispname": "European Organization for Nuclear Research",
"preferred-name": {
"@source": "internal-ani",
"$": "European Organization for Nuclear Research"
},
"sort-name": "European Organization for Nuclear Research",
"address": {
"@country": "che",
"city": "Geneva",
"state": "GE",
"postal-code": "1211",
"country": "Switzerland"
},
"org-domain": "public.web.cern.ch",

"org-URL": "http://public.web.cern.ch/Public/Welcome.html"
}
},
{
"@affiliation-id": "60007776",
"@source": "internal-ani",
"ip-doc": {
"@id": "60007776",
"@type": "parent",
"@relationship": "author",
"afnameid": "Cornell University#60007776",
"afdispname": "Cornell University",
"manual-curation": {
"@curated": "true",
"date-curation": {
"@day": "23",
"@month": "12",
"@timestamp": "2017-12-23T06:00:33.307-05:00",
"@year": "2017"
},
"curation-source": "Elsevier",
"curation-type": "SCOPUS-47"
},
"preferred-name": {
"@date-locked": "2017-12-23T18:26:14.976-05:00",
"@source": "corrapi-external",
"$": "Cornell University"
},
"sort-name": "Cornell University",
"address": {
"@country": "usa",
"address-part": "Hungerford Hill Rd",
"city": "Ithaca",
"state": "NY",
"postal-code": "14850-2488",
"country": "United States"
},
"org-domain": "cornell.edu",
"org-URL": "http://www.cornell.edu/"
}
},
{
"@affiliation-id": "60026175",
"@source": "internal-ani",
"ip-doc": {
"@id": "60026175",
"@type": "parent",
"@relationship": "author",
"afnameid": "Lawrence Livermore National Laboratory#60026175",
"afdispname": "Lawrence Livermore National Laboratory",
"preferred-name": {
"@source": "internal-ani",
"$": "Lawrence Livermore National Laboratory"
},
"sort-name": "Lawrence Livermore National Laboratory",
"address": {
"@country": "usa",
"address-part": "7000 East Ave.",
"city": "Livermore",
"state": "CA",
"postal-code": "94550-9234",
"country": "United States"
},
"org-domain": "llnl.gov",
"org-URL": "https://www.llnl.gov/%22
}
},
{
"@affiliation-id": "60005248",
"@source": "internal-ani",
"ip-doc": {
"@id": "60005248",

"@type": "parent",
"@relationship": "author",
"afnameid": "Johns Hopkins University#60005248",
"afdispname": "Johns Hopkins University",
"preferred-name": {
"@source": "internal-ani",
"$": "Johns Hopkins University"
},
"sort-name": "Johns Hopkins University",
"address": {
"@country": "usa",
"address-part": "3400 N. Charles Street",
"city": "Baltimore",
"state": "MD",
"postal-code": "21218-2680",
"country": "United States"
},
"org-domain": "jhu.edu",
"org-URL": "https://www.jhu.edu/"
}
},
{
"@affiliation-id": "60010537",
"@source": "internal-ani",
"ip-doc": {
"@id": "60010537",
"@type": "parent",
"@relationship": "author",
"afnameid": "Fairfield University#60010537",
"afdispname": "Fairfield University",
"preferred-name": {
"@source": "internal-ani",
"$": "Fairfield University"
},
"sort-name": "Fairfield University",
"address": {
"@country": "usa",
"address-part": "1073 North Benson Road",
"city": "Fairfield",
"state": "CT",
"postal-code": "06824",
"country": "United States"
},
"org-domain": "fairfield.edu",
"org-URL": "http://www.fairfield.edu/"
}
},
{
"@affiliation-id": "60020623",
"@source": "internal-ani",
"ip-doc": {
"@id": "60020623",
"@type": "parent",
"@relationship": "author",
"afnameid": "Brunel University London#60020623",
"afdispname": "Brunel University London",
"manual-curation": {
"@curated": "true",
"date-curation": {
"@day": "24",
"@month": "07",
"@timestamp": "2015-07-24T13:25:13.971-04:00",
"@year": "2015"
},
"curation-source": "QABOAPI",
"curation-type": "QABO-1542254"
},
"preferred-name": {
"@date-locked": "2015-07-24T13:25:13.971-04:00",
"@source": "corrapi-external",
"$": "Brunel University London"
},

"sort-name": "Brunel University London",
"address": {
"@country": "gbr",
"address-part": "Kingston Lane",
"city": "Uxbridge",
"state": "Middlesex",
"postal-code": "UB8 3PH",
"country": "United Kingdom"
},
"org-domain": "brunel.ac.uk",
"org-URL": "https://www.brunel.ac.uk/"
}
},
{
"@affiliation-id": "60020661",
"@source": "internal-ani",
"ip-doc": {
"@id": "60020661",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of Liverpool#60020661",
"afdispname": "University of Liverpool",
"preferred-name": {
"@source": "internal-ani",
"$": "University of Liverpool"
},
"sort-name": "Liverpool, University of",
"address": {
"@country": "gbr",
"address-part": "Foundation Building, Brownlow Hill",
"city": "Liverpool",
"state": "Merseyside",
"postal-code": "L69 7ZX",
"country": "United Kingdom"
},
"org-domain": "liverpool.ac.uk",
"org-URL": "https://www.liverpool.ac.uk/"
}
},
{
"@affiliation-id": "60011418",
"@source": "internal-ani",
"ip-doc": {
"@id": "60011418",
"@type": "parent",
"@relationship": "author",
"afnameid": "Kobe University#60011418",
"afdispname": "Kobe University",
"preferred-name": {
"@source": "internal-ani",
"$": "Kobe University"
},
"sort-name": "Kobe University",
"address": {
"@country": "jpn",
"address-part": "1-1, Rokkodai-cho",
"city": "Kobe",
"state": "Hyogo",
"postal-code": "657-8501",
"country": "Japan"
},
"org-domain": "kobe-u.ac.jp",
"org-URL": "https://www.kobe-u.ac.jp/en/index.html"
}
},
{
"@affiliation-id": "105181560",
"@parent": "60025272",
"@source": "internal-ani",
"ip-doc": {
"@id": "105181560",
"@type": "dept",

"@relationship": "author",
"afdispname": "The University of Tokyo, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "The University of Tokyo"
},
"sort-name": "Department of Physics",
"address": {
"@country": "jpn",
"address-part": "7-3-1 Hongo, Bunkyo-ku",
"city": "Tokyo",
"postal-code": "113-8654",
"country": "Japan"
},
"org-domain": "u-tokyo.ac.jp",
"org-URL": "https://www.u-tokyo.ac.jp/en/index.html"
}
},
{
"@affiliation-id": "60013791",
"@source": "internal-ani",
"ip-doc": {
"@id": "60013791",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of Hawaii at Mnoa#60013791",
"afdispname": "University of Hawaii at Mnoa",
"preferred-name": {
"@source": "internal-ani",
"$": "University of Hawaii at Mnoa"
},
"sort-name": "Hawaii at Mnoa, University of",
"address": {
"@country": "usa",
"address-part": "2500 Campus Road",
"city": "Honolulu",
"state": "HI",
"postal-code": "96822-2217",
"country": "United States"
},
"org-domain": "manoa.hawaii.edu",
"org-URL": "https://manoa.hawaii.edu/"
}
},
{
"@affiliation-id": "60025778",
"@source": "internal-ani",
"ip-doc": {
"@id": "60025778",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of Michigan, Ann Arbor#60025778",
"afdispname": "University of Michigan, Ann Arbor",
"preferred-name": {
"@source": "internal-ani",
"$": "University of Michigan, Ann Arbor"
},
"sort-name": "Michigan, Ann Arbor, University of",
"address": {
"@country": "usa",
"address-part": "500 S. State Street",
"city": "Ann Arbor",
"state": "MI",
"postal-code": "48109-1382",
"country": "United States"
},
"org-domain": "umich.edu",
"org-URL": "https://umich.edu/"

```
}
},
{
"@affiliation-id": "60031581",
"@source": "internal-ani",
"ip-doc": {
"@id": "60031581",
"@type": "parent",
"@relationship": "author",
"afnameid": "California Institute of Technology#60031581",
"afdispname": "California Institute of Technology",
"preferred-name": {
"@source": "internal-ani",
"$": "California Institute of Technology"
},
"sort-name": "California Institute of Technology",
"address": {
"@country": "usa",
"address-part": "1200 East California Boulevard",
"city": "Pasadena",
"state": "CA",
"postal-code": "91125-0001",
"country": "United States"
},
"org-domain": "caltech.edu",
"org-URL": "https://www.caltech.edu/"
}
},
{
"@affiliation-id": "60021121",
"@source": "internal-ani",
"ip-doc": {
"@id": "60021121",
"@type": "parent",
"@relationship": "author",
"afnameid": "Indiana University Bloomington#60021121",
"afdispname": "Indiana University Bloomington",
"preferred-name": {
"@source": "internal-ani",
"$": "Indiana University Bloomington"
},
"sort-name": "Indiana University Bloomington",
"address": {
"@country": "usa",
"address-part": "107 S. Indiana Avenue",
"city": "Bloomington",
"state": "IN",
"postal-code": "47405-7000",
"country": "United States"
},
"org-domain": "indiana.edu",
"org-URL": "https://www.indiana.edu/"
}
},
{
"@affiliation-id": "60024941",
"@source": "internal-ani",
"ip-doc": {
"@id": "60024941",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of California, Santa Cruz#60024941",
"afdispname": "University of California, Santa Cruz",
"preferred-name": {
"@source": "internal-ani",
"$": "University of California, Santa Cruz"
},
"sort-name": "California, Santa Cruz, University of",
"address": {
"@country": "usa",
"address-part": "1156 High Street",
"city": "Santa Cruz",
```

"state": "CA",
"postal-code": "95064-1099",
"country": "United States"
},
"org-domain": "ucsc.edu",
"org-URL": "https://www.ucsc.edu/"
}
},
{
"@affiliation-id": "112920047",
"@parent": "60025038",
"@source": "internal-ani",
"ip-doc": {
"@id": "112920047",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of California, Berkeley, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of California, Berkeley"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "2000 Carleton Street",
"city": "Berkeley",
"state": "CA",
"postal-code": "94720-2284",
"country": "United States"
},
"org-domain": "berkeley.edu",
"org-URL": "https://www.berkeley.edu/"
}
},
{
"@affiliation-id": "60025590",
"@source": "internal-ani",
"ip-doc": {
"@id": "60025590",
"@type": "parent",
"@relationship": "author",
"afnameid": "SLAC National Accelerator Laboratory#60025590",
"afdispname": "SLAC National Accelerator Laboratory",
"preferred-name": {
"@source": "internal-ani",
"$": "SLAC National Accelerator Laboratory"
},
"sort-name": "SLAC National Accelerator Laboratory",
"address": {
"@country": "usa",
"address-part": "2575 Sand Hill Road",
"city": "Menlo Park",
"state": "CA",
"postal-code": "94025-7015",
"country": "United States"
},
"org-domain": "slac.stanford.edu",
"org-URL": "https://www6.slac.stanford.edu"
}
},
{
"@affiliation-id": "104262262",
"@parent": "60025038",
"@source": "internal-ani",
"ip-doc": {
"@id": "104262262",
"@type": "dept",
"@relationship": "author",

"afdispname": "University of California, Berkeley, Department of Chemistry and Lawrence",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Chemistry and Lawrence"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of California, Berkeley"
},
"sort-name": "Department of Chemistry and Lawrence",
"address": {
"@country": "usa",
"address-part": "2000 Carleton Street",
"city": "Berkeley",
"state": "CA",
"postal-code": "94720-2284",
"country": "United States"
},
"org-domain": "berkeley.edu",
"org-URL": "https://www.berkeley.edu/"
}
},
{
"@affiliation-id": "60009982",
"@source": "internal-ani",
"ip-doc": {
"@id": "60009982",
"@type": "parent",
"@relationship": "author",
"afnameid": "Harvard University#60009982",
"afdispname": "Harvard University",
"preferred-name": {
"@source": "internal-ani",
"$": "Harvard University"
},
"sort-name": "Harvard University",
"address": {
"@country": "usa",
"address-part": "Massachusetts Hall",
"city": "Cambridge",
"state": "MA",
"postal-code": "02138-3800",
"country": "United States"
},
"org-domain": "harvard.edu",
"org-URL": "https://www.harvard.edu/"
}
},
{
"@affiliation-id": "103216411",
"@parent": "60003269",
"@source": "internal-ani",
"ip-doc": {
"@id": "103216411",
"@type": "dept",
"@relationship": "author",
"afdispname": "Princeton University, Joseph Henry Laboratories",
"preferred-name": {
"@source": "internal-ani",
"$": "Joseph Henry Laboratories"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "Princeton University"
},
"sort-name": "Joseph Henry Laboratories",
"address": {
"@country": "usa",
"address-part": "110 Morrison Hall",
"city": "Princeton",
"state": "NJ",
"postal-code": "08544-0001",

"country": "United States"
},
"org-domain": "princeton.edu",
"org-URL": "https://www.princeton.edu/"
}
},
{
"@affiliation-id": "108250966",
"@parent": "60032179",
"@source": "internal-ani",
"ip-doc": {
"@id": "108250966",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of Wisconsin-Madison, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Wisconsin-Madison"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "702 West Johnson Street, Suite 1101",
"city": "Madison",
"state": "WI",
"postal-code": "53715-1007",
"country": "United States"
},
"org-domain": "wisc.edu",
"org-URL": "http://www.wisc.edu"
}
},
{
"@affiliation-id": "100272888",
"@parent": "60025488",
"@source": "internal-ani",
"ip-doc": {
"@id": "100272888",
"@type": "dept",
"@relationship": "author",
"afdispname": "The University of Utah, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "The University of Utah"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "201 Presidents Circle",
"city": "Salt Lake City",
"state": "UT",
"postal-code": "84112-9049",
"country": "United States"
},
"org-domain": "utah.edu",
"org-URL": "https://www.utah.edu/"
}
},
{
"@affiliation-id": "105188410",
"@parent": "60012708",
"@source": "internal-ani",
"ip-doc": {
"@id": "105188410",

"@type": "dept",
"@relationship": "author",
"afdispname": "Stanford University, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@date-locked": "2016-12-30T12:15:04.293-05:00",
"@source": "corrapi-external",
"$": "Stanford University"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "450 Serra Mall, Stanford",
"city": "Palo Alto",
"state": "CA",
"postal-code": "94305",
"country": "United States"
},
"org-domain": "stanford.edu",
"org-URL": "http://www.stanford.edu/"
}
},
{
"@affiliation-id": "104448638",
"@parent": "60028628",
"@source": "internal-ani",
"ip-doc": {
"@id": "104448638",
"@type": "dept",
"@relationship": "author",
"afdispname": "Northeastern University, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "Northeastern University"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "360 Huntington Ave.",
"city": "Boston",
"state": "MA",
"postal-code": "02115-5000",
"country": "United States"
},
"org-domain": "northeastern.edu",
"org-URL": "https://www.northeastern.edu/"
}
},
{
"@affiliation-id": "100257077",
"@parent": "60005837",
"@source": "internal-ani",
"ip-doc": {
"@id": "100257077",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of Houston, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Houston"
},

"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "4800 Calhoun Rd.",
"city": "Houston",
"state": "TX",
"postal-code": "77204-2693",
"country": "United States"
},
"org-domain": "uh.edu",
"org-URL": "https://www.uh.edu/"
}
},
{
"@affiliation-id": "60016340",
"@source": "internal-ani",
"ip-doc": {
"@id": "60016340",
"@type": "parent",
"@relationship": "author",
"afnameid": "INFN, Laboratori Nazionali Di Frascati#60016340",
"afdispname": "INFN, Laboratori Nazionali Di Frascati",
"preferred-name": {
"@source": "internal-ani",
"$": "INFN, Laboratori Nazionali Di Frascati"
},
"sort-name": "INFN, National Laboratory of Frascati",
"address": {
"@country": "ita",
"address-part": "Via Enrico Fermi 40",
"city": "Frascati",
"postal-code": "00044",
"country": "Italy"
},
"org-domain": "w3.lnf.infn.it",
"org-URL": "http://w3.lnf.infn.it/?lang=en"
}
},
{
"@affiliation-id": "105177061",
"@parent": "60000221",
"@source": "internal-ani",
"ip-doc": {
"@id": "105177061",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of Colorado Boulder, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Colorado Boulder"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "20 UCB",
"city": "Boulder",
"state": "CO",
"postal-code": "80309-0001",
"country": "United States"
},
"org-domain": "colorado.edu",
"org-URL": "http://www.colorado.edu/"
}
},
{
"@affiliation-id": "105282824",
"@parent": "60003269",
"@source": "internal-ani",

"ip-doc": {
"@id": "105282824",
"@type": "dept",
"@relationship": "author",
"afdispname": "Princeton University, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "Princeton University"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "110 Morrison Hall",
"city": "Princeton",
"state": "NJ",
"postal-code": "08544-0001",
"country": "United States"
},
"org-domain": "princeton.edu",
"org-URL": "https://www.princeton.edu/"
}
},
{
"@affiliation-id": "105378094",
"@parent": "60025590",
"@source": "internal-ani",
"ip-doc": {
"@id": "105378094",
"@type": "dept",
"@relationship": "author",
"afdispname": "SLAC National Accelerator Laboratory, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "SLAC National Accelerator Laboratory"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "2575 Sand Hill Road",
"city": "Menlo Park",
"state": "CA",
"postal-code": "94025-7015",
"country": "United States"
},
"org-domain": "slac.stanford.edu",
"org-URL": "https://www6.slac.stanford.edu"
}
},
{
"@affiliation-id": "60003269",
"@source": "internal-ani",
"ip-doc": {
"@id": "60003269",
"@type": "parent",
"@relationship": "author",
"afnameid": "Princeton University#60003269",
"afdispname": "Princeton University",
"preferred-name": {
"@source": "internal-ani",
"$": "Princeton University"
},
"sort-name": "Princeton University",
"address": {
"@country": "usa",

"address-part": "110 Morrison Hall",
"city": "Princeton",
"state": "NJ",
"postal-code": "08544-0001",
"country": "United States"
},
"org-domain": "princeton.edu",
"org-URL": "https://www.princeton.edu/"
}
},
{
"@affiliation-id": "60025038",
"@source": "internal-ani",
"ip-doc": {
"@id": "60025038",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of California, Berkeley#60025038",
"afdispname": "University of California, Berkeley",
"manual-curation": {
"@curated": "true",
"date-curation": {
"@day": "23",
"@month": "02",
"@timestamp": "2018-02-23T06:27:24.094-05:00",
"@year": "2018"
},
"curation-source": "QABOAPI",
"curation-type": "QABO-3729332"
},
"preferred-name": {
"@source": "internal-ani",
"$": "University of California, Berkeley"
},
"sort-name": "California, Berkeley, University of",
"address": {
"@country": "usa",
"address-part": "2000 Carleton Street",
"city": "Berkeley",
"state": "CA",
"postal-code": "94720-2284",
"country": "United States"
},
"org-domain": "berkeley.edu",
"org-URL": "https://www.berkeley.edu/"
}
},
{
"@affiliation-id": "103216339",
"@parent": "60003269",
"@source": "internal-ani",
"ip-doc": {
"@id": "103216339",
"@type": "dept",
"@relationship": "author",
"afdispname": "Princeton University, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "Princeton University"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "110 Morrison Hall",
"city": "Princeton",
"state": "NJ",
"postal-code": "08544-0001",
"country": "United States"

},
"org-domain": "princeton.edu",
"org-URL": "https://www.princeton.edu/"
}
},
{
"@affiliation-id": "122463837",
"@parent": "60072925",
"@source": "internal-ani",
"ip-doc": {
"@id": "122463837",
"@type": "dept",
"@relationship": "author",
"afdispname": "Njala University, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "Njala University"
},
"sort-name": "Department of Physics",
"address": {
"@country": "sle",
"address-part": "Private Mail Bag",
"city": "Freetown",
"country": "Sierra Leone"
},
"org-domain": "nu-online.com",
"org-URL": "http://www.nu-online.com/"
}
},
{
"@affiliation-id": "104297594",
"@parent": "60025590",
"@source": "internal-ani",
"ip-doc": {
"@id": "104297594",
"@type": "dept",
"@relationship": "author",
"afdispname": "SLAC National Accelerator Laboratory, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "SLAC National Accelerator Laboratory"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "2575 Sand Hill Road",
"city": "Menlo Park",
"state": "CA",
"postal-code": "94025-7015",
"country": "United States"
},
"org-domain": "slac.stanford.edu",
"org-URL": "https://www6.slac.stanford.edu"
}
},
{
"@affiliation-id": "109524692",
"@parent": "60032179",
"@source": "internal-ani",
"ip-doc": {
"@id": "109524692",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of Wisconsin-Madison, Department of Physics",

"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Wisconsin-Madison"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "702 West Johnson Street, Suite 1101",
"city": "Madison",
"state": "WI",
"postal-code": "53715-1007",
"country": "United States"
},
"org-domain": "wisc.edu",
"org-URL": "http://www.wisc.edu"
}
},
{
"@affiliation-id": "105165943",
"@parent": "60030612",
"@source": "internal-ani",
"ip-doc": {
"@id": "105165943",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of California, San Diego, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of California, San Diego"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "9500 Gilman Drive",
"city": "San Diego",
"state": "CA",
"postal-code": "92093-0021",
"country": "United States"
},
"org-domain": "ucsd.edu",
"org-URL": "https://www.ucsd.edu/"
}
},
{
"@affiliation-id": "60000745",
"@source": "internal-ani",
"ip-doc": {
"@id": "60000745",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of Illinois at Urbana-Champaign#60000745",
"afdispname": "University of Illinois at Urbana-Champaign",
"preferred-name": {
"@source": "internal-ani",
"$": "University of Illinois at Urbana-Champaign"
},
"sort-name": "Illinois at Urbana-Champaign, University of",
"address": {
"@country": "usa",
"address-part": "901 West Illinois Street",
"city": "Urbana",
"state": "IL",
"postal-code": "61801-3444",
"country": "United States"

},
"org-domain": "illinois.edu",
"org-URL": "https://illinois.edu/"
}
},
{
"@affiliation-id": "104479385",
"@parent": "60025597",
"@source": "internal-ani",
"ip-doc": {
"@id": "104479385",
"@type": "dept",
"@relationship": "author",
"afdispname": "The University of Chicago, Enrico Fermi Institute",
"preferred-name": {
"@source": "internal-ani",
"$": "Enrico Fermi Institute"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "The University of Chicago"
},
"sort-name": "Enrico Fermi Institute",
"address": {
"@country": "usa",
"address-part": "Edward H. Levi Hall, 5801 South Ellis Avenue",
"city": "Chicago",
"state": "IL",
"postal-code": "60637-5418",
"country": "United States"
},
"org-domain": "uchicago.edu",
"org-URL": "https://www.uchicago.edu/"
}
},
{
"@affiliation-id": "100247007",
"@parent": "60006297",
"@source": "internal-ani",
"ip-doc": {
"@id": "100247007",
"@type": "dept",
"@relationship": "author",
"afdispname": "University of Pennsylvania, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Pennsylvania"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "1 College Hall, Room 100",
"city": "Philadelphia",
"state": "PA",
"postal-code": "19104-6303",
"country": "United States"
},
"org-domain": "home.www.upenn.edu",
"org-URL": "https://home.www.upenn.edu/"
}
},
{
"@affiliation-id": "105769371",
"@parent": "60000221",
"@source": "internal-ani",
"ip-doc": {
"@id": "105769371",
"@type": "dept",

"@relationship": "author",
"afdispname": "University of Colorado Boulder, Department of Physics & Astrophysics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics & Astrophysics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "University of Colorado Boulder"
},
"sort-name": "Department of Physics & Astrophysics",
"address": {
"@country": "usa",
"address-part": "20 UCB",
"city": "Boulder",
"state": "CO",
"postal-code": "80309-0001",
"country": "United States"
},
"org-domain": "colorado.edu",
"org-URL": "http://www.colorado.edu/"
}
},
{
"@affiliation-id": "60030612",
"@source": "internal-ani",
"ip-doc": {
"@id": "60030612",
"@type": "parent",
"@relationship": "author",
"afnameid": "University of California, San Diego#60030612",
"afdispname": "University of California, San Diego",
"manual-curation": {
"@curated": "true",
"date-curation": {
"@day": "20",
"@month": "02",
"@timestamp": "2018-02-20T19:13:26.482-05:00",
"@year": "2018"
},
"curation-source": "QABOAPI",
"curation-type": "QABO-3724732"
},
"preferred-name": {
"@source": "internal-ani",
"$": "University of California, San Diego"
},
"sort-name": "California, San Diego, University of",
"address": {
"@country": "usa",
"address-part": "9500 Gilman Drive",
"city": "San Diego",
"state": "CA",
"postal-code": "92093-0021",
"country": "United States"
},
"org-domain": "ucsd.edu",
"org-URL": "https://www.ucsd.edu/"
}
},
{
"@affiliation-id": "60030635",
"@source": "internal-ani",
"ip-doc": {
"@id": "60030635",
"@type": "parent",
"@relationship": "author",
"afnameid": "Deutsches Elektronen-Synchrotron (DESY)#60030635",
"afdispname": "Deutsches Elektronen-Synchrotron (DESY)",
"manual-curation": {
"@curated": "true",
"date-curation": {

"@day": "12",
"@month": "07",
"@timestamp": "2019-07-12T01:51:16.790-04:00",
"@year": "2019"
},
"curation-source": "PARITY",
"curation-type": "PARITY-11072019225116421"
},
"preferred-name": {
"@date-locked": "2019-06-18T11:42:17.399-04:00",
"@source": "corrapi-external",
"$": "Deutsches Elektronen-Synchrotron (DESY)"
},
"sort-name": "Deutsches Elektronen-Synchrotron",
"address": {
"@country": "deu",
"address-part": "Notkestr. 85 city:Hamburg",
"city": "Hamburg",
"state": "Brandenburg",
"postal-code": "22603",
"country": "Germany"
},
"org-domain": "desy.de",
"org-URL": "http://www.desy.de/"
}
},
{
"@affiliation-id": "105179505",
"@parent": "60022195",
"@source": "internal-ani",
"ip-doc": {
"@id": "105179505",
"@type": "dept",
"@relationship": "author",
"afdispname": "Massachusetts Institute of Technology, Department of Physics",
"preferred-name": {
"@source": "internal-ani",
"$": "Department of Physics"
},
"parent-preferred-name": {
"@source": "internal-ani",
"$": "Massachusetts Institute of Technology"
},
"sort-name": "Department of Physics",
"address": {
"@country": "usa",
"address-part": "77 Massachusetts Avenue",
"city": "Cambridge",
"state": "MA",
"postal-code": "02139-4301",
"country": "United States"
},
"org-domain": "mit.edu",
"org-URL": "http://www.mit.edu/"
}
},
{
"@affiliation-id": "60075450",
"@source": "internal-ani",
"ip-doc": {
"@id": "60075450",
"@type": "parent",
"@relationship": "author",
"afnameid": "Frist Campus Center#60075450",
"afdispname": "Frist Campus Center",
"preferred-name": {
"@source": "internal-ani",
"$": "Frist Campus Center"
},
"sort-name": "Frist Campus Center",
"address": {
"@country": "usa",

"address-part": "Frist Ln",
"city": "Princeton",
"state": "NJ",
"postal-code": "08544-0001",
"country": "United States"
},
"org-domain": "princeton.edu",
"org-URL": "http://www.princeton.edu/frist/"
}
},
{
"@affiliation-id": "105951559",
"@source": "internal-ani",
"ip-doc": {
"@id": "105951559",
"@type": "parent",
"@relationship": "author",
"afnameid": "Volkswagen Foundation#105951559",
"afdispname": "Volkswagen Foundation",
"preferred-name": {
"@source": "internal-ani",
"$": "Volkswagen Foundation"
},
"sort-name": "Volkswagen Foundation"
}
}
]
}
}
}
]
}


3. In the next step, a new metric is created with the data retrieved from Scopus and the type of metric indicated in the <p> parameter, such as: if the <p> parameter is scopus-author-h-index and given the response from scopus, the result would be

| ***MetricType*** | **scopus-author-h-index** |
|---|---|
| *Last* | ***true*** |
| *MetricCount* | **68.0** (value contained in the tag “h-index“) |
| *AcquisitionDate* | **date** on which the metric was recorded |
| *Remark* | **null** |

In case a metric for this Publication was already on db, it is not overwritten, but its ‘Last’ flag is set to ‘false’.

**WOS (Web of Science) Metrics**

Web of Science metrics collection is handled by the process hereafter described. To have connection with Web of Science properly working, following configuration properties must be set

```
metrics.wos.citation-count.url = <wos query url i.e. https://wos-api.
clarivate.com/api/wos/?databaseId=WOS&lang=en&usrQuery=>
metrics.wos.citation-count.apiKey = <wos api key>
```

If last property is the same used to import publications from Web of Science, its value can be “inherited” from ${wos.apiKey} property already set, in this way

```
metrics.wos.citation-count.apiKey = ${wos.apiKey}
```

The updating of publication and researcher metrics from WOS (Web of Science) service is controlled by two processes, which can be run from both processes section and Command Line Interface:

| | |
|---|---|
| ***update-metrics wos*** | *updates metrics of type entityType=Publication* |
| **update-metrics wos-person** | *updates metrics of type entityType=Person* |

The script ***update-metrics wos*** applies the following steps to perform the update *Publication type metrics*:

1. performs a global search to retrieve all the items of type Publication that have a metadata dc.identifier.doi set.
2. taking one item at a time - extracts the metadata value from metadata dc.identifier.doi, with these value it constructs the query to be sent to the external WOS service which in turn returns the document containing the metric.

a generic answer from WOS can be:

---


```
{
   "Data": {
      "Records": {
         "records": {
            "REC": [
               {
                  "UID": "WOS:000544718700031",
                  "static_data": {
                     "summary": {
                        "pub_info": {
                           "coverdate": "JUL 2020",
                           "vol": 48,
                           "pubyear": 2020,
                           "issue": 7,
                           "sortdate": "2020-07-01",
                           "has_abstract": "Y",
                           "pubmonth": "JUL",
                           "pubtype": "Journal",
                           "page": {
                              "end": 1008,
                              "begin": 1001,
                              "page_count": 8,
                              "content": "1001-1008"
                           }
                        },
                        "names": {
                           "count": 6,
                           "name": [
                              {
                                 "seq_no": 1,
                                 "role": "author",
                                 "full_name": "Saffaran, Sina",
```

```
    "addr_no": 1,
    "last_name": "Saffaran",
    "display_name": "Saffaran, Sina",
    "wos_standard": "Saffaran, S",
    "daisng_id": 12953931,
    "first_name": "Sina"
  },
  {
    "seq_no": 2,
    "role": "author",
    "full_name": "Das, Anup",
    "addr_no": 1,
    "last_name": "Das",
    "display_name": "Das, Anup",
    "wos_standard": "Das, A",
    "daisng_id": 4813219,
    "first_name": "Anup"
  },
  {
    "seq_no": 3,
    "role": "author",
    "full_name": "Laffey, John G.",
    "addr_no": 2,
    "last_name": "Laffey",
    "display_name": "Laffey, John G.",
    "wos_standard": "Laffey, JG",
    "daisng_id": 80745,
    "first_name": "John G."
  },
  {
    "seq_no": 4,
    "role": "author",
    "full_name": "Hardman, Jonathan G.",
    "addr_no": 3,
    "last_name": "Hardman",
    "display_name": "Hardman, Jonathan G.",
    "wos_standard": "Hardman, JG",
    "daisng_id": 35285335,
    "first_name": "Jonathan G."
  },
```

```
        {
            "seq_no": 5,
            "role": "author",
            "full_name": "Yehya, Nadir",
            "reprint": "Y",
            "addr_no": 4,
            "last_name": "Yehya",
            "display_name": "Yehya, Nadir",
            "wos_standard": "Yehya, N",
            "daisng_id": 673390,
            "first_name": "Nadir"
        },
        {
            "seq_no": 6,
            "role": "author",
            "full_name": "Bates, Declan G.",
            "addr_no": 1,
            "last_name": "Bates",
            "display_name": "Bates, Declan G.",
            "wos_standard": "Bates, DG",
            "daisng_id": 273309,
            "first_name": "Declan G."
        }
    ]
},
"doctypes": {
    "doctype": "Article",
    "count": 1
},
"publishers": {
    "publisher": {
        "names": {
            "count": 1,
            "name": {
                "seq_no": 1,
                "role": "publisher",
                "full_name": "LIPPINCOTT WILLIAMS & WILKINS",
                "addr_no": 1,
                "display_name": "LIPPINCOTT WILLIAMS & WILKINS"
            }
        },
```

```
        "address_spec": {
          "city": "PHILADELPHIA",
          "addr_no": 1,
          "full_address": "TWO COMMERCE SQ, 2001 MARKET ST, PHILADELPHIA, PA 19103 USA"
        }
      }
    },
    "EWUID": {
      "WUID": {
        "coll_id": "WOS"
      },
      "edition": {
        "value": "WOS.SCI"
      }
    },
    "titles": {
      "count": 6,
      "title": [
        {
          "type": "source",
          "content": "CRITICAL CARE MEDICINE"
        },
        {
          "type": "source_abbrev",
          "content": "CRIT CARE MED"
        },
        {
          "type": "abbrev_iso",
          "content": "Crit. Care Med."
        },
        {
          "type": "abbrev_11",
          "content": "CRIT CARE M"
        },
        {
          "type": "abbrev_29",
          "content": "CRIT CARE MED"
        },
        {
          "type": "item",
```

```
                "content": "Utility of Driving Pressure and Mechanical Power to Guide Protective Ventilator Settings in Two Cohorts of Adult and Pediatric Patients With Acute Respiratory Distress Syndrome: A Computational Investigation"
            }
          ]
        }
      },
      "item": {
        "xsi:type": "itemType_wos",
        "coll_id": "WOS",
        "ids": {
          "avail": "N",
          "content": "ME5TS"
        },
        "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
        "bib_pagecount": {
          "type": "Journal",
          "content": 270
        },
        "keywords_plus": {
          "count": 10,
          "keyword": [
            "ACUTE LUNG INJURY",
            "END-EXPIRATORY PRESSURE",
            "TIDAL VOLUME",
            "RECRUITMENT MANEUVERS",
            "INTENSIVE-CARE",
            "HIGH PEEP",
            "CHILDREN",
            "OXYGENATION",
            "MORTALITY",
            "SURVIVAL"
          ]
        },
        "bib_id": "48 (7): 1001-1008 JUL 2020"
      },
      "fullrecord_metadata": {
        "addresses": {
          "count": 4,
          "address_name": [
            {
              "names": {
```

```
    "count": 3,
    "name": [
      {
        "seq_no": 1,
        "role": "author",
        "full_name": "Saffaran, Sina",
        "addr_no": 1,
        "last_name": "Saffaran",
        "display_name": "Saffaran, Sina",
        "wos_standard": "Saffaran, S",
        "daisng_id": 12953931,
        "first_name": "Sina"
      },
      {
        "seq_no": 2,
        "role": "author",
        "full_name": "Das, Anup",
        "addr_no": 1,
        "last_name": "Das",
        "display_name": "Das, Anup",
        "wos_standard": "Das, A",
        "daisng_id": 4813219,
        "first_name": "Anup"
      },
      {
        "seq_no": 6,
        "role": "author",
        "full_name": "Bates, Declan G.",
        "addr_no": 1,
        "last_name": "Bates",
        "display_name": "Bates, Declan G.",
        "wos_standard": "Bates, DG",
        "daisng_id": 273309,
        "first_name": "Declan G."
      }
    ]
  },
  "address_spec": {
    "country": "England",
    "city": "Coventry",
    "addr_no": 1,
```

```
        "organizations": {
          "organization": [
            "Univ Warwick",
            {
              "pref": "Y",
              "content": "University of Warwick"
            }
          ],
          "count": 2
        },
        "full_address": "Univ Warwick, Sch Engn, Coventry, W Midlands, England",
        "state": "W Midlands",
        "suborganizations": {
          "count": 1,
          "suborganization": "Sch Engn"
        }
      }
    },
    {
      "names": {
        "count": 1,
        "name": {
          "seq_no": 3,
          "role": "author",
          "full_name": "Laffey, John G.",
          "addr_no": 2,
          "last_name": "Laffey",
          "display_name": "Laffey, John G.",
          "wos_standard": "Laffey, JG",
          "daisng_id": 80745,
          "first_name": "John G."
        }
      },
      "address_spec": {
        "country": "Ireland",
        "city": "Galway",
        "addr_no": 2,
        "organizations": {
          "organization": [
            "NUI Galway",
```

```
                {
                    "pref": "Y",
                    "content": "National University of Ireland (NUI) Galway"
                }
            ],
            "count": 2
        },
        "full_address": "NUI Galway, Sch Med, Anaesthesia & Intens Care Med, Galway, Ireland",
        "suborganizations": {
            "count": 2,
            "suborganization": [
                "Sch Med",
                "Anaesthesia & Intens Care Med"
            ]
        }
    }
},
{
    "names": {
        "count": 1,
        "name": {
            "seq_no": 4,
            "role": "author",
            "full_name": "Hardman, Jonathan G.",
            "addr_no": 3,
            "last_name": "Hardman",
            "display_name": "Hardman, Jonathan G.",
            "wos_standard": "Hardman, JG",
            "daisng_id": 35285335,
            "first_name": "Jonathan G."
        }
    },
    "address_spec": {
        "country": "England",
        "city": "Nottingham",
        "addr_no": 3,
        "organizations": {
            "organization": [
                "Univ Nottingham",
                {
                    "pref": "Y",
```

```
                "content": "University of Nottingham"
            }
          ],
          "count": 2
        },
        "full_address": "Univ Nottingham, Sch Med, Div Clin Neurosci, Anaesthesia & Crit Care, Nottingham, England",
        "suborganizations": {
          "count": 3,
          "suborganization": [
            "Sch Med",
            "Div Clin Neurosci",
            "Anaesthesia & Crit Care"
          ]
        }
      }
    },
    {
      "names": {
        "count": 1,
        "name": {
          "seq_no": 5,
          "role": "author",
          "full_name": "Yehya, Nadir",
          "reprint": "Y",
          "addr_no": 4,
          "last_name": "Yehya",
          "display_name": "Yehya, Nadir",
          "wos_standard": "Yehya, N",
          "daisng_id": 673390,
          "first_name": "Nadir"
        }
      },
      "address_spec": {
        "zip": {
          "location": "AP",
          "content": 19104
        },
        "country": "USA",
        "city": "Philadelphia",
        "addr_no": 4,
```

```
                "organizations": {
                    "organization": [
                        "Univ Penn",
                        {
                            "pref": "Y",
                            "content": "University of Pennsylvania"
                        },
                        {
                            "pref": "Y",
                            "content": "Childrens Hospital of Philadelphia"
                        }
                    ],
                    "count": 3
                },
                "full_address": "Univ Penn, Childrens Hosp Philadelphia, Dept Anaesthesiol & Crit Care Med, Philadelphia, PA
19104 USA",
                "state": "PA",
                "suborganizations": {
                    "count": 2,
                    "suborganization": [
                        "Childrens Hosp Philadelphia",
                        "Dept Anaesthesiol & Crit Care Med"
                    ]
                }
            }
        }
    ]
},
"category_info": {
    "subheadings": {
        "count": 1,
        "subheading": "Life Sciences & Biomedicine"
    },
    "subjects": {
        "subject": [
            {
                "ascatype": "traditional",
                "content": "Critical Care Medicine"
            },
            {
                "ascatype": "extended",
```

```
                "content": "General & Internal Medicine"
            }
        ],
        "count": 2
    },
    "headings": {
        "heading": "Science & Technology",
        "count": 1
    }
},
"normalized_languages": {
    "count": 1,
    "language": {
        "type": "primary",
        "content": "English"
    }
},
"languages": {
    "count": 1,
    "language": {
        "type": "primary",
        "content": "English"
    }
},
"keywords": {
    "count": 6,
    "keyword": [
        "adult acute respiratory distress syndrome",
        "computer simulation",
        "mechanical ventilation",
        "pediatric acute respiratory distress syndrome",
        "protective ventilation",
        "ventilator-induced lung injury"
    ]
},
"refs": {
    "count": 31
},
"reprint_addresses": {
    "count": 1,
    "address_name": {
```

```
"names": {
    "count": 1,
    "name": {
        "seq_no": 1,
        "role": "author",
        "full_name": "Yehya, Nadir",
        "reprint": "Y",
        "addr_no": 1,
        "last_name": "Yehya",
        "display_name": "Yehya, Nadir",
        "wos_standard": "Yehya, N",
        "first_name": "Nadir"
    }
},
"address_spec": {
    "zip": {
        "location": "AP",
        "content": 19104
    },
    "country": "USA",
    "city": "Philadelphia",
    "addr_no": 1,
    "organizations": {
        "organization": [
            "Univ Penn",
            {
                "pref": "Y",
                "content": "University of Pennsylvania"
            },
            {
                "pref": "Y",
                "content": "Childrens Hospital of Philadelphia"
            }
        ],
        "count": 3
    },
    "full_address": "Univ Penn, Childrens Hosp Philadelphia, Dept Anaesthesiol & Crit Care Med, Philadelphia, PA 19104
USA",
    "state": "PA",
    "suborganizations": {
        "count": 2,
```

```
            "suborganization": [
               "Childrens Hosp Philadelphia",
               "Dept Anaesthesiol & Crit Care Med"
            ]
         }
      }
   },
   "abstracts": {
      "count": 1,
      "abstract": {
         "abstract_text": {
            "p": "Objectives: Mechanical power and driving pressure have been proposed as indicators, and possibly drivers, of ventilator-induced lung injury. We tested the utility of these different measures as targets to derive maximally protective ventilator settings. Design: A high-fidelity computational simulator was matched to individual patient data and used to identify strategies that minimize driving pressure, mechanical power, and a modified mechanical power that removes the direct linear, positive dependence between mechanical power and positive end-expiratory pressure. Setting: Interdisciplinary Collaboration in Systems Medicine Research Network. Subjects: Data were collected from a prospective observational cohort of pediatric acute respiratory distress syndrome from the Children's Hospital of Philadelphia (n= 77) and from the low tidal volume arm of the Acute Respiratory Distress Syndrome Network tidal volume trial (n= 100). Interventions: Global optimization algorithms evaluated more than 26.7 million changes to ventilator settings (approximately 150,000 per patient) to identify strategies that minimize driving pressure, mechanical power, or modified mechanical power. Measurements and Main Results: Large average reductions in driving pressure (pediatric: 23%, adult: 23%), mechanical power (pediatric: 44%, adult: 66%), and modified mechanical power (pediatric: 61%, adult: 67%) were achievable in both cohorts when oxygenation and ventilation were allowed to vary within prespecified ranges. Reductions in driving pressure (pediatric: 12%, adult: 2%), mechanical power (pediatric: 24%, adult: 46%), and modified mechanical power (pediatric: 44%, adult: 46%) were achievable even when no deterioration in gas exchange was allowed. Minimization of mechanical power and modified mechanical power was achieved by increasing tidal volume and decreasing respiratory rate. In the pediatric cohort, minimum driving pressure was achieved by reducing tidal volume and increasing respiratory rate and positive end-expiratory pressure. The Acute Respiratory Distress Syndrome Network dataset had limited scope for further reducing tidal volume, but driving pressure was still significantly reduced by increasing positive end-expiratory pressure. Conclusions: Our analysis identified different strategies that minimized driving pressure or mechanical power consistently across pediatric and adult datasets. Minimizing standard and alternative formulations of mechanical power led to significant increases in tidal volume. Targeting driving pressure for minimization resulted in ventilator settings that also reduced mechanical power and modified mechanical power, but not vice versa.",
            "count": 1
         }
      }
   },
   "fund_ack": {
      "grants": {
         "count": 4,
         "grant": [
            {
               "grant_agency": "Research Councils UK (RCUK)"
            },
            {
               "grant_ids": {
                  "grant_id": "EP/P023444/1",
                  "count": 1
               },
               "grant_agency": "Engineering and Physical Sciences RCUK"
```

```
                    },
                    {
                        "grant_agency": "National Heart, Lung, and Blood Institute"
                    },
                    {
                        "grant_ids": {
                            "grant_id": "NIH K23 HL-136688",
                            "count": 1
                        },
                        "grant_agency": "National Institutes of Health (NIH)"
                    }
                ]
            },
            "fund_text": {
                "p": "Drs. Das, Hardman, and Bates received support for article research from the Research Councils UK (RCUK). Drs. Hardman's and Bates's institution received funding from Engineering and Physical Sciences RCUK (Grant Number EP/P023444/1). Dr. Yehya's institution received funding from the National Heart, Lung, and Blood Institute and he received support for article research from the National Institutes of Health (NIH) (Grant Number NIH K23 HL-136688). The remaining authors have disclosed that they do not have any potential conflicts of interest."
            }
        },
        "normalized_doctypes": {
            "doctype": "Article",
            "count": 1
        }
    }
},
"r_id_disclaimer": "ResearcherID data provided by Clarivate Analytics",
"dynamic_data": {
    "citation_related": {
        "tc_list": {
            "silo_tc": {
                "coll_id": "WOS",
                "local_count": 87
            }
        }
    },
    "cluster_related": {
        "identifiers": {
            "identifier": [
                {
                    "type": "issn",
```

```
                        "value": "0090-3493"
                    },
                    {
                        "type": "eissn",
                        "value": "1530-0293"
                    },
                    {
                        "type": "doi",
                        "value": "10.1097/CCM.0000000000000000"
                    },
                    {
                        "type": "pmid",
                        "value": "MEDLINE:32574467"
                    }
                ]
            }
        }
    }
  }
 ]
}
},
"QueryResult": {
    "QueryID": 1,
    "RecordsSearched": 64666906,
    "RecordsFound": 1
}
}
```


---

1. In th next step, a new metric is created with the data retrieved from wos, such as:

| | |
|---|---|
| MetricType | ***wosCitation*** |
| Last | ***true*** |
| MetricCount | ***87***<br>( containing in the following path: $.Data.Records.records.REC[0].dynamic_data.citation_related.tc_list.silo_tc.local_count ) |
| AcquisitionDate | **date** on which the metric was recorded |
| Remark | this field not defined for this type of metric |

In case a previous metric for the given item was already present in our database, its ‘Last’ field is set to ‘false’

The **update-metrics wos-person** script applies the same sequence of steps,

to retrieve entities of type *Person* with metadata person.identifier.orcid set

, The response contains n records , one for each publication where one of the authors has  the provided orcid id, the value of this metric is the sum of the individual values contained in each record.

Example:

| | |
|---|---|
| MetricType | ***wosPersonCitation*** |
| Last | ***true*** |
| MetricCount | ***2453*** |
| AcquisitionDate | **date** on which the metric was recorded |
| Remark | this field not defined for this type of metric |

**Scanning WOS (Web of Science) for additional publications in profiles**

To import new publications from WOS, run the following script:

**import-publications wos**

where:

**import-publications** is the name of the script

**wos** the name of the external service from which we want to import

The script applies the following steps to perform the update:

1. performs a global search to retrieve all the items of type *Person* that have a metadata person.identifier.orcid or person.identifier.rid set.
2. taking one item at a time - extracts the metadata values from person.identifier.orcid and/or person.identifier.rid, with these values it constructs the query to be sent to the external WOS service which in turn returns the document containing records.

a generic response from Scopus containing 1 record can be:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<response xmlns="http://www.isinet.com/xrpc42">
    <map>
        <map name="Data">
            <val name="Records">
                <![CDATA[<records><REC r_id_disclaimer="ResearcherID
data provided by Clarivate Analytics"><UID>WOS:000490897000016<
/UID><static_data><summary><EWUID><WUID coll_id="WOS" /><edition value="
WOS.ISTP" /></EWUID><pub_info sortdate="2018-01-01" pubyear="2018"
has_abstract="N" coverdate="2018" vol="11100" pubtype="Book in series"
><page begin="341" end="348" page_count="8">341-348</page><
/pub_info><titles count="7"><title type="source">BRAVERMAN READINGS IN
MACHINE LEARNING: KEY IDEAS FROM INCEPTION TO CURRENT STATE<
/title><title type="series">Lecture Notes in Artificial Intelligence<
/title><title type="source_abbrev">LECT NOTES ARTIF INT</title><title
type="abbrev_11">LECT N A I</title><title type="abbrev_29">LECT NOTE
ARTIF INTELL</title><title type="item">Misha Braverman: My Mentor and
My Model</title><title type="book_series" translated="N">Lecture Notes
in Artificial Intelligence</title></titles><names count="4"><name
seq_no="1" role="author" reprint="Y" addr_no="1 2" daisng_id="31471066"
><display_name>Mirkin, Boris</display_name><full_name>Mirkin, Boris<
```

/full_name><wos_standard>Mirkin, B</wos_standard><first_name>Boris<
/first_name><last_name>Mirkin</last_name></name><name seq_no="2" role="
book_editor"><display_name>Rozonoer, L<
/display_name><full_name>Rozonoer, L</full_name><wos_standard>Rozonoer,
L</wos_standard><first_name>L</first_name><last_name>Rozonoer<
/last_name></name><name seq_no="3" role="book_editor"
><display_name>Mirkin, B</display_name><full_name>Mirkin, B<
/full_name><wos_standard>Mirkin, B</wos_standard><first_name>B<
/first_name><last_name>Mirkin</last_name></name><name seq_no="4" role="
book_editor"><display_name>Muchnik, I</display_name><full_name>Muchnik,
I</full_name><wos_standard>Muchnik I</wos_standard><last_name>Muchnik<
/last_name><suffix>I</suffix></name></names><doctypes count="1"
><doctype>Proceedings Paper</doctype></doctypes><conferences count="1"
><conference conf_id="331117"><conf_infos count="1"
><conf_info>International Conference on Braverman Readings in Machine
Learning - Key Ideas from Inception to Current State, APR 28-30, 2017,
Boston, MA</conf_info></conf_infos><conf_titles count="1"
><conf_title>International Conference on Braverman Readings in Machine
Learning - Key Ideas from Inception to Current State</conf_title><
/conf_titles><conf_dates count="1"><conf_date conf_start="20170428"
conf_end="20170430">APR 28-30, 2017</conf_date><
/conf_dates><conf_locations count="1"><conf_location><conf_host>NE Univ<
/conf_host><conf_city>Boston</conf_city><conf_state>MA</conf_state><
/conf_location></conf_locations></conference><
/conferences><publishers><publisher><address_spec addr_no="1"
><full_address>GEWERBESTRASSE 11, CHAM, CH-6330, SWITZERLAND<
/full_address><city>CHAM</city></address_spec><names count="1"><name
role="publisher" seq_no="1" addr_no="1"><display_name>SPRINGER
INTERNATIONAL PUBLISHING AG</display_name><full_name>SPRINGER
INTERNATIONAL PUBLISHING AG</full_name></name></names></publisher><
/publishers></summary><fullrecord_metadata><languages count="1"
><language type="primary">English</language><
/languages><normalized_languages count="1"><language type="primary"
>English</language></normalized_languages><normalized_doctypes count="1"
><doctype>Meeting</doctype></normalized_doctypes><refs count="0"
/><addresses count="2"><address_name><address_spec addr_no="1"
><full_address>Natl Res Univ Higher Sch Econ, Dept Data Anal &
Artificial Intelligence, Moscow, Russia</full_address><organizations
count="2"><organization>Natl Res Univ Higher Sch Econ<
/organization><organization pref="Y">HSE University (National Research
University Higher School of Economics)</organization><
/organizations><suborganizations count="1"><suborganization>Dept Data
Anal & Artificial Intelligence</suborganization><
/suborganizations><city>Moscow</city><country>Russia</country><
/address_spec><names count="1"><name seq_no="1" role="author" reprint="
Y" addr_no="1" daisng_id="31471066"><display_name>Mirkin, Boris<
/display_name><full_name>Mirkin, Boris</full_name><wos_standard>Mirkin,
B</wos_standard><first_name>Boris</first_name><last_name>Mirkin<
/last_name></name></names></address_name><address_name><address_spec
addr_no="2"><full_address>Birkbeck Univ London, Dept Comp Sci, London,

England</full_address><organizations count="3"><organization>Birkbeck Univ London</organization><organization pref="Y">University of London</organization><organization pref="Y">Birkbeck University London</organization></organizations><suborganizations count="1"><suborganization>Dept Comp Sci</suborganization></suborganizations><city>London</city><country>England</country></address_spec><names count="1"><name seq_no="1" role="author" reprint="Y" addr_no="2" daisng_id="31471066"><display_name>Mirkin, Boris</display_name><full_name>Mirkin, Boris</full_name><wos_standard>Mirkin, B</wos_standard><first_name>Boris</first_name><last_name>Mirkin</last_name></name></names></address_name></addresses><reprint_addresses count="2"><address_name><address_spec addr_no="1"><full_address>Natl Res Univ Higher Sch Econ, Dept Data Anal & Artificial Intelligence, Moscow, Russia</full_address><organizations count="2"><organization>Natl Res Univ Higher Sch Econ</organization><organization pref="Y">HSE University (National Research University Higher School of Economics)</organization></organizations><suborganizations count="1"><suborganization>Dept Data Anal & Artificial Intelligence</suborganization></suborganizations><city>Moscow</city><country>Russia</country></address_spec><names count="1"><name seq_no="1" role="author" reprint="Y" addr_no="1"><display_name>Mirkin, Boris</display_name><full_name>Mirkin, Boris</full_name><wos_standard>Mirkin, B</wos_standard><first_name>Boris</first_name><last_name>Mirkin</last_name></name></names></address_name><address_name><address_spec addr_no="2"><full_address>Birkbeck Univ London, Dept Comp Sci, London, England</full_address><organizations count="3"><organization>Birkbeck Univ London</organization><organization pref="Y">University of London</organization><organization pref="Y">Birkbeck University London</organization></organizations><suborganizations count="1"><suborganization>Dept Comp Sci</suborganization></suborganizations><city>London</city><country>England</country></address_spec><names count="1"><name seq_no="1" role="author" reprint="Y" addr_no="2"><display_name>Mirkin, Boris</display_name><full_name>Mirkin, Boris</full_name><wos_standard>Mirkin, B</wos_standard><first_name>Boris</first_name><last_name>Mirkin</last_name></name></names></address_name></reprint_addresses><category_info><headings count="1"><heading>Science & Technology</heading></headings><subheadings count="1"><subheading>Technology</subheading></subheadings><subjects count="2"><subject ascatype="traditional">Computer Science, Artificial Intelligence</subject><subject ascatype="extended">Computer Science</subject></subjects></category_info></fullrecord_metadata><item xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="itemType_wos" coll_id="WOS"><ids avail="N">BO0HJ</ids><bib_id>11100: 341-348 2018</bib_id><bib_pagecount type="Book">353</bib_pagecount><book_pages>353</book_pages><book_notes count="2"><book_note>Figures</book_note><book_note>Color plates</book_note></book_notes><book_desc><bk_binding>P</bk_binding><bk_publisher>SPRINGER INTERNATIONAL PUBLISHING AG, GEWERBESTRASSE 11, CHAM, CH-6330,

```
SWITZERLAND</bk_publisher><bk_prepay>N</bk_prepay><
/book_desc><book_desc><bk_binding>H</bk_binding><bk_publisher>SPRINGER
INTERNATIONAL PUBLISHING AG, GEWERBESTRASSE 11, CHAM, CH-6330,
SWITZERLAND</bk_publisher><bk_prepay>N</bk_prepay></book_desc></item><
/static_data><dynamic_data><citation_related><tc_list><silo_tc coll_id="
WOS" local_count="0" /></tc_list><
/citation_related><cluster_related><identifiers><identifier type="issn"
value="0302-9743" /><identifier type="eissn" value="1611-3349"
/><identifier type="eisbn" value="978-3-319-99492-5" /><identifier
type="isbn" value="978-3-319-99491-8" /><identifier type="doi" value="
10.1007/978-3-319-99492-5_16" /></identifiers></cluster_related><
/dynamic_data></REC></records>]]>
</val>
</map>
<map name="QueryResult">
    <map>
        <val name="QueryID">1</val>
        <val name="RecordsSearched">65117068</val>
        <val name="RecordsFound">1</val>
    </map>
</map>
</map>
</response>
```


3. For each record a check is made if a publication with the same metadata dc.identifier.other as the record does not already exist. If it does not exist, create a new workspaceItem in the dedicated collection (contained in the property wos.importworkspaceitem.collection-id). the metadata that are inserted into the new workspaceitem are configured in the wos-integration.xml file

The following fields must also be configured in the ***dspace.cfg*** file:

**directorios.community-id** : uuid of the community where to retrieve items of the type Person

**wos.importworkspaceitem.collection-id :** uuid of the collection into which the workspaceitems will be placed

#### Scanning Scopus for additional publications in profiles

To import new publications from scopus, run the following script: **import-publications scopus**

where:

**import-publications** is the name of the script

**scopus** the name of the external service from which we want to import

The script applies the following steps to perform the import:

1. performs a global search to retrieve all the items of type *Person* that have a metadata person.identifier.scopus-author-id set.
2. taking one item at a time - extracts the metadata value from metadata person.identifier.scopus-author-id, with these value it constructs the query to be sent to the external Scopus service which in turn returns the document containing records.

a generic response from Scopus containing 3 records can be:

---


<?xml version="1.0" encoding="UTF-8"?>

<search-results xmlns="http://www.w3.org/2005/Atom" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/" xmlns:prism="http://prismstandard.org/namespaces/basic/2.0/" xmlns:atom="http://www.w3.org/2005/Atom">

  <opensearch:totalResults>3</opensearch:totalResults>

  <opensearch:startIndex>0</opensearch:startIndex>

  <opensearch:itemsPerPage>3</opensearch:itemsPerPage>

<opensearch:Query role="request" searchTerms="(misha boychuk)" startPage="0"/>

<link ref="self" href="https://api.elsevier.com/content/search/scopus?start=0&count=25&query=%28misha+boychuk%29&view=COMPLETE" type="application/xml"/>

<link ref="first" href="https://api.elsevier.com/content/search/scopus?start=0&count=25&query=%28misha+boychuk%29&view=COMPLETE" type="application/xml"/>

<entry>

<link ref="self" href="https://api.elsevier.com/content/abstract/scopus_id/85056894828"/>

<link ref="author-affiliation" href="https://api.elsevier.com/content/abstract/scopus_id/85056894828?field=author,affiliation"/>

<link ref="scopus" href="https://www.scopus.com/inward/record.uri?partnerID=HzOxMe3b&scp=85056894828&origin=inward"/>

<link ref="scopus-citedby" href="https://www.scopus.com/inward/citedby.uri?partnerID=HzOxMe3b&scp=85056894828&origin=inward"/>

<prism:url>https://api.elsevier.com/content/abstract/scopus_id/85056894828</prism:url>

<dc:identifier>SCOPUS_ID:85056894828</dc:identifier>

<eid>2-s2.0-85056894828</eid>

<dc:title>Moss flora of Zeysky State Nature Reserve (Tukuringra Range, Amur Province, Russia)</dc:title>

<dc:creator>Dudov S.</dc:creator>

<prism:publicationName>Botanica Pacifica</prism:publicationName>

<prism:issn>22264701</prism:issn>

<prism:eIssn>24103713</prism:eIssn>

<prism:volume>7</prism:volume>

<prism:issueIdentifier>2</prism:issueIdentifier>

<prism:pageRange>83-104</prism:pageRange>

<prism:coverDate>2018-11-01</prism:coverDate>

<prism:coverDisplayDate>1 November 2018</prism:coverDisplayDate>

<prism:doi>10.17581/bp.2018.07204</prism:doi>

<dc:description>© Botanical Garden-Institute FEB RAS. 2018. An annotated list of the moss flora of Zeysky Nature Reserve is presented. It includes 310 species, with 140 species newly recorded for the reserve and 25 species new for Amur Province, including two species, Hondaella caperata and Hyophila involuta from the Red Data Book of Russian Federation. Other interesting records include recently described species (Amphidium asiaticum, Hedwigia kuzenevae, Sphagnum mirum), species on the western border of their distribution (Cryphaea amurensis, Dicranum pacificum, Hondaella caperata, Leucodon coreensis and Stereodon calcicola), on the southern edge of its distribution (Psilopilum cavifolium) and rare species with scattered localities in the southern Far East (Hyophila involuta, Seligeria donniana). A comparison with other moss floras in Asian Russia of about the same area indicates that the moss flora of Zeisky Reserve is more similar to other floras of the Amur River basin and also to Transbaicalian floras, rather than to the floras of Primorsky Territory, as the latter is much more enriched by East Asian flora elements.</dc:description>

<citedby-count>1</citedby-count>

<affiliation>

<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60110534</affiliation-url>

<afid>60110534</afid>

<affilname>Tsitsin Main Botanical Garden, Russian Academy of Sciences</affilname>

<affiliation-city>Moscow</affiliation-city>

<affiliation-country>Russian Federation</affiliation-country>

</affiliation>

<affiliation>

<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60007457</affiliation-url>

<afid>60007457</afid>

```
    <affilname>Lomonosov Moscow State University</affilname>
    <affiliation-city>Moscow</affiliation-city>
    <affiliation-country>Russian Federation</affiliation-country>
</affiliation>
<affiliation>
    <affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/116497163</affiliation-url>
    <afid>116497163</afid>
    <affilname>Zeya State Nature Reserve</affilname>
    <affiliation-city/>
    <affiliation-country>Russian Federation</affiliation-country>
</affiliation>
<prism:aggregationType>Journal</prism:aggregationType>
<subtype>ar</subtype>
<subtypeDescription>Article</subtypeDescription>
<author-count limit="100" total="5">5</author-count>
<author seq="1">
    <author-url>https://api.elsevier.com/content/author/author_id/57189892516</author-url>
    <authid>57189892516</authid>
    <authname>Dudov S.</authname>
    <surname>Dudov</surname>
    <given-name>Sergey V.</given-name>
    <initials>S.V.</initials>
    <afid>60007457</afid>
    <afid>116497163</afid>
</author>
<author seq="2">
    <author-url>https://api.elsevier.com/content/author/author_id/54401367100</author-url>
    <authid>54401367100</authid>
    <authname>Kozhin M.</authname>
    <surname>Kozhin</surname>
    <given-name>Mikhail N.</given-name>
    <initials>M.N.</initials>
    <afid>60007457</afid>
</author>
<author seq="3">
    <author-url>https://api.elsevier.com/content/author/author_id/55856047600</author-url>
    <authid>55856047600</authid>
    <authname>Fedosov V.</authname>
    <surname>Fedosov</surname>
    <given-name>Vladimir E.</given-name>
    <initials>V.E.</initials>
```

```
      <afid>60007457</afid>
    </author>
    <author seq="4">
      <author-url>https://api.elsevier.com/content/author/author_id/7003301657</author-url>
      <authid>7003301657</authid>
      <authname>Ignatova E.</authname>
      <surname>Ignatova</surname>
      <given-name>Elena A.</given-name>
      <initials>E.A.</initials>
      <afid>60007457</afid>
    </author>
    <author seq="5">
      <author-url>https://api.elsevier.com/content/author/author_id/6603847397</author-url>
      <authid>6603847397</authid>
      <authname>Ignatov M.</authname>
      <surname>Ignatov</surname>
      <given-name>Michael S.</given-name>
      <initials>M.S.</initials>
      <afid>60007457</afid>
      <afid>60110534</afid>
    </author>
    <authkeywords>Altitude zonation | Far East | Moss | Phytogeography | Zeysky Reserve |  |  |  |  | </authkeywords>
    <source-id>21100806944</source-id>
    <fund-acr>RSF</fund-acr>
    <fund-no>181400121</fund-no>
    <fund-sponsor>Russian Science Foundation</fund-sponsor>
    <openaccess>1</openaccess>
    <openaccessFlag>true</openaccessFlag>
    <freetoread>
      <value>all</value>
      <value>publisherfree2read</value>
    </freetoread>
    <freetoreadLabel>
      <value>All Open Access</value>
      <value>Bronze</value>
    </freetoreadLabel>
  </entry>
  <entry>
    <link ref="self" href="https://api.elsevier.com/content/abstract/scopus_id/85056898644"/>
    <link ref="author-affiliation" href="https://api.elsevier.com/content/abstract/scopus_id/85056898644?field=author,affiliation"/>
```

<link ref="scopus" href="https://www.scopus.com/inward/record.uri?partnerID=HzOxMe3b&scp=85056898644&origin=inward"/>

<link ref="scopus-citedby" href="https://www.scopus.com/inward/citedby.uri?partnerID=HzOxMe3b&scp=85056898644&origin=inward"/>

<prism:url>https://api.elsevier.com/content/abstract/scopus_id/85056898644</prism:url>

<dc:identifier>SCOPUS_ID:85056898644</dc:identifier>

<eid>2-s2.0-85056898644</eid>

<dc:title>Mosses of the southern Russian Far East, an annotated check-list</dc:title>

<dc:creator>Cherdantseva V.</dc:creator>

<prism:publicationName>Botanica Pacifica</prism:publicationName>

<prism:issn>22264701</prism:issn>

<prism:eIssn>24103713</prism:eIssn>

<prism:volume>7</prism:volume>

<prism:issueIdentifier>2</prism:issueIdentifier>

<prism:pageRange>53-81</prism:pageRange>

<prism:coverDate>2018-11-01</prism:coverDate>

<prism:coverDisplayDate>1 November 2018</prism:coverDisplayDate>

<prism:doi>10.17581/bp.2018.07206</prism:doi>

<dc:description>© Botanical Garden-Institute FEB RAS. 2018. The check-list of mosses of the southern part of the Russian Far East includes 816 species and 10 infraspecific taxa with references on their distribution in seven floristic regions within Primorsky and Khabarovsky Territories, Amurskaya and Sakhlinskaya Provinces and Evreiskaya Autonomous District. Seventy one species are excluded in the course of the check-list compilation, and 59 are commented as doubtful and erroneously reported from some of the Far Eastern regions, while 8 of them doubtful for the southern part of Russian Far East in general.</dc:description>

<citedby-count>7</citedby-count>

<affiliation>

<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60110534</affiliation-url>

<afid>60110534</afid>

<affilname>Tsitsin Main Botanical Garden, Russian Academy of Sciences</affilname>

<affiliation-city>Moscow</affiliation-city>

<affiliation-country>Russian Federation</affiliation-country>

</affiliation>

<affiliation>

<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60110166</affiliation-url>

<afid>60110166</afid>

<affilname>Botanical Garden-Institute FEB RAS</affilname>

<affiliation-city>Vladivostok</affiliation-city>

<affiliation-country>Russian Federation</affiliation-country>

</affiliation>

<affiliation>

<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60103842</affiliation-url>

<afid>60103842</afid>

<affilname>Central Siberian Botanical Garden, SB RAS</affilname>

<affiliation-city>Novosibirsk</affiliation-city>

<affiliation-country>Russian Federation</affiliation-country>

```
</affiliation>
<affiliation>
    <affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60007457</affiliation-url>
    <afid>60007457</afid>
    <affilname>Lomonosov Moscow State University</affilname>
    <affiliation-city>Moscow</affiliation-city>
    <affiliation-country>Russian Federation</affiliation-country>
</affiliation>
<prism:aggregationType>Journal</prism:aggregationType>
<subtype>ar</subtype>
<subtypeDescription>Article</subtypeDescription>
<author-count limit="100" total="7">7</author-count>
<author seq="1">
    <author-url>https://api.elsevier.com/content/author/author_id/6504776522</author-url>
    <authid>6504776522</authid>
    <authname>Cherdantseva V.</authname>
    <surname>Cherdantseva</surname>
    <given-name>Valentina Ya</given-name>
    <initials>V.Y.</initials>
    <afid>60103842</afid>
</author>
<author seq="2">
    <author-url>https://api.elsevier.com/content/author/author_id/57193766939</author-url>
    <authid>57193766939</authid>
    <authname>Pisarenko O.</authname>
    <surname>Pisarenko</surname>
    <given-name>Olga Yu</given-name>
    <initials>O.Y.</initials>
    <afid>60103842</afid>
</author>
<author seq="3">
    <author-url>https://api.elsevier.com/content/author/author_id/6603847397</author-url>
    <authid>6603847397</authid>
    <authname>Ignatov M.</authname>
    <surname>Ignatov</surname>
    <given-name>Michael S.</given-name>
    <initials>M.S.</initials>
    <afid>60007457</afid>
    <afid>60110534</afid>
</author>
<author seq="4">
```

<author-url>https://api.elsevier.com/content/author/author_id/7003301657</author-url>
<authid>7003301657</authid>
<authname>Ignatova E.</authname>
<surname>Ignatova</surname>
<given-name>Elena A.</given-name>
<initials>E.A.</initials>
<afid>60007457</afid>
</author>
<author seq="5">
<author-url>https://api.elsevier.com/content/author/author_id/55856047600</author-url>
<authid>55856047600</authid>
<authname>Fedosov V.</authname>
<surname>Fedosov</surname>
<given-name>Vladimir E.</given-name>
<initials>V.E.</initials>
<afid>60007457</afid>
</author>
<author seq="6">
<author-url>https://api.elsevier.com/content/author/author_id/57189892516</author-url>
<authid>57189892516</authid>
<authname>Dudov S.</authname>
<surname>Dudov</surname>
<given-name>Sergey V.</given-name>
<initials>S.V.</initials>
<afid>60007457</afid>
</author>
<author seq="7">
<author-url>https://api.elsevier.com/content/author/author_id/57201079804</author-url>
<authid>57201079804</authid>
<authname>Bakalin V.</authname>
<surname>Bakalin</surname>
<given-name>Vadim A.</given-name>
<initials>V.A.</initials>
<afid>60110166</afid>
</author>
<authkeywords>Amurskaya Province | Biodiversity | Bryophyte | Evreiskaya Autonomous District | Flora | Khabarovsky Territory | Primorsky Territory | Sakhlinskaya Province | Synonymy | Taxonomy | | | | | | | | | | </authkeywords>
<source-id>21100806944</source-id>
<fund-acr>RSF</fund-acr>
<fund-no>18-14-00121</fund-no>
<fund-sponsor>Russian Science Foundation</fund-sponsor>

<openaccess>1</openaccess>

<openaccessFlag>true</openaccessFlag>

<freetoread>

<value>all</value>

<value>publisherfree2read</value>

</freetoread>

<freetoreadLabel>

<value>All Open Access</value>

<value>Bronze</value>

</freetoreadLabel>

</entry>

<entry>

<link ref="self" href="https://api.elsevier.com/content/abstract/scopus_id/85030554178"/>

<link ref="author-affiliation" href="https://api.elsevier.com/content/abstract/scopus_id/85030554178?field=author,affiliation"/>

<link ref="scopus" href="https://www.scopus.com/inward/record.uri?partnerID=HzOxMe3b&scp=85030554178&origin=inward"/>

<link ref="scopus-citedby" href="https://www.scopus.com/inward/citedby.uri?partnerID=HzOxMe3b&scp=85030554178&origin=inward"/>

<prism:url>https://api.elsevier.com/content/abstract/scopus_id/85030554178</prism:url>

<dc:identifier>SCOPUS_ID:85030554178</dc:identifier>

<eid>2-s2.0-85030554178</eid>

<dc:title>A revision of the genus seligeria (Seligeriaceae, bryophyta) in russia inferred from molecular data</dc:title>

<dc:creator>Fedosov V.E.</dc:creator>

<prism:publicationName>Phytotaxa</prism:publicationName>

<prism:issn>11793155</prism:issn>

<prism:eIssn>11793163</prism:eIssn>

<prism:volume>323</prism:volume>

<prism:issueIdentifier>1</prism:issueIdentifier>

<prism:pageRange>27-50</prism:pageRange>

<prism:coverDate>2017-09-26</prism:coverDate>

<prism:coverDisplayDate>26 September 2017</prism:coverDisplayDate>

<prism:doi>10.11646/phytotaxa.323.1.2</prism:doi>



<dc:description>© 2017 Magnolia Press. The genus Seligeria is revised based on morphological and DNA sequence data of nuclear ITS and chloroplastic trnL-F. Fifteen species from most infrageneric units of the genus are recovered in two well supported phylogenetic clusters that are also distinctive in morphology. The clade with the type species of the genus, S. pusilla, includes also S. donniana, S. brevifolia, S. calcarea, S. patula, S. tristichoides, S. trifaria, and S. oelandica. These species are characterized by short, cupulate or turbinate capsules widened towards the mouth, and the lack of a stem central strand. Another clade includes species with rather long, mainly ovate to cylindrical capsules and more or less developed stem central strand: S. campylopoda, S. recurvata, S. subimmersa, S. diversifolia, and S. polaris. These two clusters do not show sister relationships, but the second one appears more closely related to the Blindia clade. To resolve the apparent paraphyly, the latter phylogenetic group is segregated in a genus Blindiadelphus. In some aspects of morphology and ecology it is intermediate between Seligeria s. str. and Blindia, but differs from both genera in subquadrate upper leaf cells and thin- to moderately thick-walled rectangular exothecial cells. Molecular phylogenetic analyses revealed heterogeneity within the specimens previously referred to Blindiadelphus campylopodus, indicating a presence in Asian Russia of an undescribed species that is described here as Blindiadelphus sibiricus. It differs from B. campylopodus by the larger spores and typically rounded leaf apices. The isotype specimen of S. galinae appeared to be nearly identical to S. donniana in the sequences of ITS and trnL-F, and examination of morphology revealed no substantial differences between these species. Thus, we consider S. galinae as a synonym of S. donniana. The genus Blindiadelphus includes species of Seligeria subg. Blindiadelphus and S. subg. Cyrtoseligeria, which however are found intermingled in the molecular phylogenetic analysis. Thus the genus Blindiadelphus is accepted without any infrageneric taxa. The phylogenetic tree is congruent with the subdivision of the genus Seligeria s.str into subg. Seligeria, subg. Anodon, subg. Megalosporia and one newly established subgenus Robustidontia for S. brevifolia.</dc:description>

<citedby-count>6</citedby-count>
<affiliation>
<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60110534</affiliation-url>
<afid>60110534</afid>
<affilname>Tsitsin Main Botanical Garden, Russian Academy of Sciences</affilname>
<affiliation-city>Moscow</affiliation-city>
<affiliation-country>Russian Federation</affiliation-country>
</affiliation>
<affiliation>
<affiliation-url>https://api.elsevier.com/content/affiliation/affiliation_id/60007457</affiliation-url>
<afid>60007457</afid>
<affilname>Lomonosov Moscow State University</affilname>
<affiliation-city>Moscow</affiliation-city>
<affiliation-country>Russian Federation</affiliation-country>
</affiliation>
<prism:aggregationType>Journal</prism:aggregationType>
<subtype>ar</subtype>
<subtypeDescription>Article</subtypeDescription>
<author-count limit="100" total="4">4</author-count>
<author seq="1">
<author-url>https://api.elsevier.com/content/author/author_id/55856047600</author-url>
<authid>55856047600</authid>
<authname>Fedosov V.E.</authname>
<surname>Fedosov</surname>
<given-name>Vladimir E.</given-name>
<initials>V.E.</initials>
<afid>60007457</afid>
</author>
<author seq="2">
<author-url>https://api.elsevier.com/content/author/author_id/56996923100</author-url>
<authid>56996923100</authid>
<authname>Fedorova A.V.</authname>
<surname>Fedorova</surname>
<given-name>Alina V.</given-name>
<initials>A.V.</initials>
<afid>60007457</afid>
</author>
<author seq="3">
<author-url>https://api.elsevier.com/content/author/author_id/7003301657</author-url>
<authid>7003301657</authid>
<authname>Ignatova E.A.</authname>

```
        <surname>Ignatova</surname>
        <given-name>Elena A.</given-name>
        <initials>E.A.</initials>
        <afid>60007457</afid>
      </author>
      <author seq="4">
        <author-url>https://api.elsevier.com/content/author/author_id/6603847397</author-url>
        <authid>6603847397</authid>
        <authname>Ignatov M.S.</authname>
        <surname>Ignatov</surname>
        <given-name>Michael S.</given-name>
        <initials>M.S.</initials>
        <afid>60007457</afid>
        <afid>60110534</afid>
      </author>
      <authkeywords>Blindia | Blindiadelphus | Cyrtoseligeria | Grimmiales | ITS | Molecular phylogenetics | Russia | Seligeria | TrnL-F< /authkeywords>
      <source-id>21100209326</source-id>
      <fund-acr>RSF</fund-acr>
      <fund-no>14-50-00029</fund-no>
      <fund-sponsor>Russian Science Foundation</fund-sponsor>
      <openaccess>0</openaccess>
      <openaccessFlag>false</openaccessFlag>
    </entry>
</search-results>
```


---

3. For each record a check is made if a publication with the same metadata dc.identifier.scopus as the record does not already exist. If it does not exist, create a new workspaceItem in the dedicated collection (contained in the property scopus.importworkspaceitem.collection-id). the metadata that are inserted into the new workspaceitem are configured in the scopus-integration.xml file

The following fields must also be configured in the ***dspace.cfg*** file:

**directorios.community-id** : uuid of the community where to retrieve items of the type Person

**scopus.importworkspaceitem.collection-id :** uuid of the collection into which the workspaceitems will be placed

### Usage statistics data generators

In **statistics.xml** file it is possible to configure one or many generators to be used to extract statistics data for a given DSpace object or objects associated to it by mean of DSpace-CRIS inverse relations mechanism.

Currently available generators, all implementations of `org.dspace.app.rest.statistics.UsageReportGenerator`, are:

| Java Class | Extracted Data |
|---|---|
| `org.dspace.app.rest.statistics.TopCitiesGenerator` | List of cities from where DSpace Object, or its related Objects, visits are coming, sorted by number of visits in decreasing order |
| `org.dspace.app.rest.statistics.TopCountriesGenerator` | List of countries from where DSpace Object, or its related Objects, visits are coming, sorted by number of visits in decreasing order |
| `org.dspace.app.rest.statistics.TopItemsGenerator` | Usage report of the items most popular over the entire site or a specific community, collection |
| | |

| | |
|---|---|
| `org.dspace.app.rest.statistics.TotalDownloadsAndVisitsGenerator` | Number of times a DSpace Object, or its related Objects, have been visited and its attachments have been downloaded |
| `org.dspace.app.rest.statistics.TotalDownloadsGenerator` | Number of times a DSpace Object, or its related Objects, attachments have been downloaded |
| `org.dspace.app.rest.statistics.TotalVisitGenerator` | Number of times a DSpace Object, or its related Objects, attachments have been visited |
| `org.dspace.app.rest.statistics.TotalVisitPerPeriodGenerator` | Number of times a DSpace Object, or its related Objects, attachments have been visited in a period of time, grouped by a period duration (month by month, year by year, etc.) |

For each generator, following properties are provided:

- **viewMode**: rendering of data, possible values are: table (default) , chart.line, chart.bar, map (geographical map)
- **maxResults**: maximum number of statistical data to be returned
- **relation**: If defined, it instructs the generator to extract data related to DSpaceObject received in input by mean of this inverse relation. This value, should match to one of inverse relations defined in **discovery.xml** file, for the type of DSpaceObject for which generator will provide data.

For `org.dspace.app.rest.statistics.TotalVisitPerPeriodGenerator` implementation, “periodType” and “increment” fields are provided and will be used by solr query responsible of extracting period related statistics. “periodType” specifies period granularity (“year”, “month” or “day”), “increment” is used to define period steps, for example, an increment of 1 for a “month” periodType will extract visits grouped 1 month per time. This configuration

```
<bean id="totalVisitPerPeriodGenerator" class="org.dspace.app.rest.
statistics.TotalVisitPerPeriodGenerator">
        <property name="viewMode" value="chart.line"/>
        <property name="maxResults" value="6"/>
        <property name="periodType" value="month"/>
        <property name="increment" value="1"/>
</bean>
```

extracts item visits of last 6 months, grouped by single month.

An simple generator, which extracts usage data for item received in input is configured in this way

```
<bean id="totalVisitGenerator" class="org.dspace.app.rest.statistics.
TotalVisitGenerator">
        <property name="viewMode" value="table"/>
</bean>
```

while this other generator will extract data related to objects related to item received in input by mean of inverse relation “RELATION.Person.researchoutputs“ defined in **discovery.xml** file

```
<bean id="totalVisitGeneratorRelationPersonResearchoutputs" class="org.
dspace.app.rest.statistics.TotalVisitGenerator">
        <property name="viewMode" value="table"/>
        <property name="relation" value="RELATION.Person.
researchoutputs"/>
</bean>
```

**statistics.xml** file entry point to define generators mapping is the bean of type `org.dspace.app.rest.statistics.StatisticsReportsConfiguration`. Its “mapping” property contains a map between DSpaceObject instance types and a list of `org.dspace.app.rest.model.UsageReportCategoryRest` instances. For each of those instances, following properties are set:

- **categoryType**: keyword identifying generated report(s) category
- **reports**: a List of org.dspace.app.rest.statistics.UsageReportGenerator instances (see above), which will be used to extract usage data and statistics for a given DSpaceObject and/or other DSpace Objects related to it.

DSpaceObject instance types defined as “key” in “mapping” map could be:

- site
- community
- collection
- item
- item of a specific type (i.e. “item-Person”, “item-Publication”, etc.)

REST contracts to obtain statistics data are defined at https://github.com/4Science/Rest7Contract/blob/dspace-cris-7/statistics-categories.md

https://github.com/4Science/Rest7Contract/blob/dspace-cris-7/statistics-reports.md

On DSpace-CRIS, statistics are available to logged users at link

http(s)://<dspace-cris-base-url>/statistics/items/<dspace-object-id>

#### Usage statistics on database

By mean of “`store-metrics`” process, which can be triggered via processes UI section or CLI, it is possible to generate view and download usage statistics for each item part of DSpace-CRIS repository, and have them stored as Item’s metric in cris_metrics table. In this way, view and download data can be made available within a DSpace-CRIS Item metrics box.

## How to configure and manage the translations

In this page we will describe how to enable and configure the multilingual support.

The platform allows a full internationalization of the UIs for both the frontend than the backend. To enable the support for multiple language the following properties must be defined in the dspace configuration

```
# Default Locale
# A Locale in the form country or country_language or
country_language_variant
# if no default locale is defined the server default locale will be
used.
default.locale = en

# All the Locales, that are supported by this instance of DSpace
# A comma-separated list of Locales. All types of Locales country,
country_language, country_language_variant
# Note that the appropriate file are present, especially that all the
Messages_x.properties are there
# may be used, e. g:
webui.supported.locales = en, es
```

The backend uses the JAVA messages properties file to translate text used in the REST API or dynamically resolved in other template. Such files MUST be available as resource in the JAVA classpatch of the webapp or the command line scripts. The English file can be used as default to make further translation, community managed translations are available in the dspace-api-lang github project.

The submission forms are configured via the submission-forms.xml configuration file that can be replicated with language postfix (i.e. `submission-forms_es.xml`) to provide input screens specific for each language. Similarly, the notification messages can be translated providing template in the different language.

The angular frontend uses JSON files to resolve keys in actual labels for the users. The English file can be used as default to make further translation, community managed translations are available in the same folder. The file must be named using the ISO 639-1 code of the language and the extension `.json5` (i.e `es.json5`)

## User agreement

Once logged in, the user who has not already accepted the terms and conditions must read and agree to the End User Agreement. After logging in, these users are then redirected to the **info/end-user-agreement** page which shows them the terms of use in their language, if available, or in English. Until the user declares that he has accepted the user agreement, he cannot browse other pages.

Once the terms and conditions have been accepted, the user can continue browsing the site normally and will not have to accept the same terms again at the next login, except if the administrator has changed them in the meantime and forced a new acceptance by all users. (see **"User agrement editing" section**).

The metadata **dspace.agreements.end-user** associated with the ePerson is used to store the information relating to the acceptance or not of the terms and conditions.

To not allow the user to browse the application if he has not already accepted the user agreement and to show him the page with these agreements, check/filters have been added both on the Angular side and on the REST side:

- Angular side the **EndUserAgreementCurrentUserGuard** redirect the user to the user agreement acceptance page if the current user has not already accepted them (the presence of the metadata dspace.agreements.end-user with value "true" is checked)
- REST side the **UserAgreementFilter** blocks calls to REST endpoints, returning a response with error code 403 Forbidden, if the current user wants to access endpoints that require a preliminary acceptance of the terms and conditions (see **"User agreement filter configuration"** section for more details) .

#### User agreement editing

The administrator has a specific page available on **admin/edit-user-agreement** to modify the terms and conditions in the various languages available. This page shows the texts of the user agreement in a series of textarea, one for each language, and allows them to be modified.

The various translations of the user agreement text are stored in the **dc.rights** metadata of the **site** object.

Once the editing is completed, by clicking on the save button, the administrator can choose whether to force all users to accept the new terms at their next login or not. This forcing is implemented by deleting all the values of the metadata dspace.agreements.end-user present in the database. To do this, a specific script called **metadata-deletion** is used; this script receives as an input parameter the name of the metadata for which its values must be deleted.

#### User agreement filter configuration

In order not to allow users who have not accepted the terms and conditions to be able to freely contact the REST endpoints, a security filter has been implemented through the class UserAgreementFilter which returns an error response in case the user had not already accepted the user agreement.

The filter can be disabled with the property **user-agreement.filter-enabled** and can be configured not to block requests to specific endpoints by setting the property **user-agreement.open-path-patterns**.

The filter checks if the user has accepted the terms or not by reading this information in the jwt token (userAgreementAccepted attribute). This attribute is set during login by the UserAgreementClaimProvider provider based on the presence or absence of metadata dspace.agreements. end-user with a value of true among the metadata of the ePerson associated with the current user. The same provider in the parse phase of the jwt then set the userAgreementAccepted attribute of the request with the value present in the jwt itself so that it can then be read later by the filter.

#### User agreement ignore

It is possible to allow an eperson to call rest endpoint without forcing such eperson to accept user agreement. By setting the metadata '`dspace.agreements.ignore`' to 'true' for an eperson, this person will be able to login and consume REST endpoint as authenticated user without forcing the acceptance of User agreement.

## How to configure the notification system

The email server settings are configured in the `dspace.cfg` (smtp server to use, credentials, from),

The application uses named email template build with Apache Velocity. The template are stored in the /config/email the **i18n support** can be added providing **template file with language postfix** (i.e. register, register_es, register_it).

Here is an example of a notification template (`/config/emails/register`)

```
#set($subject = "${config.get('dspace.name')} Account Registration")

To complete registration for a DSpace account, please click the link

below:
```

```
  ${params[0]}

If you need assistance with your account, please email

${config.get("mail.admin")} or call us at xxx-555-xxxx.

The DSpace Team
```

`#set($MAIL-HEADER = value)` allows to set a mail header such as the subject, a ccn, etc.

`${config.get('PARAM')}` provides access to a configuration parameters

`${params[0]}` provides access to parameters specific of the event to notify, they are usually documented as comment at the start of the template

### Notification Broker

This feature is the result of the OpenAIRE Advance Open Call for Innovation Project “Enrich local data via the OpenAIRE Graph” awarded to 4Science see https://www.4science.it/en/2020/09/07/openaire-advance-premia-4science-per-il-progetto-enrich-local-data-via-the-openaire-graph-fase-2/

The detailed documentation is maintained on a dedicated project website https://4science.github.io/oaire-eld/#/

### Items export

The item export functionality allows users to export the metadata of one or multiple items in a specific format among those configured. Based on the type and number of entities to be exported, it is possible to obtain results with different formats (XML, JSON, PDF, CSV etc…).

The two main export modes are the **single item export** and the **multiple items export**. Each of these two modes is associated with a specific script. With the same type of entity to be exported, the cardinality of the items to be exported (one or many) can affects the available export result formats. For example, if you decide to export a single researcher profile the available formats could be XML, JSON, PDF or RTF, while if you choose to export many profiles you could do it in XML, JSON, CSV and XSL. However, the available formats are not static but can be configured: it is possible to establish both which information to export and the structure of the file itself. The configuration of export formats is based on a series of editable text files that act as **templates**.

For example, export an item with the following template:

```
<person>
  <name>@dc.title@</name>
  <knows-languages>
    <language>@person.knowsLanguage@</language>
  </knows-languages>
</person>
```

might produce the following xml file:

```
<person>
  <name>John Smith</name>
  <knows-languages>
    <language>English</language>
    <language>Italian</language>
  </knows-languages>
</person>
```

#### StreamDisseminationCrosswalk and ItemExportCrosswalk

The export logic is implemented by a set of classes that implement the **ItemExportCrosswalk** interface and that serialize in a specific format one or multiple items. This interface extends **StreamDisseminationCrosswalk** interface and it allows to distinguish the classes that can be used for exporting the item in the various formats available. The current implementations used by the export functionality to obtain the items in a specific format are:

- **ReferCrosswalk:** Generates a textual representation of the item/items starting from a template file in which there are a set of placeholders.
- **DocumentCrosswalk:** Generates a document starting from the chosen item in the configured format (such as PDF or RTF). This implementation is based on an XSL transformation made from a template file written with the XSL-FO language.
- **TabularCrosswalk**: Abstract implementation that, starting from the items chosen for export, generates a table structure with configurable headings starting from a template file. The actual format of the table is determined by the classes that extend this abstract class. Currently available implementations are:
  - **XlsCrosswalk**: the data of the items in the tabular form is written into an xls file
  - **CsvCrosswalk**: items metadata are exported in csv format
- **CSLItemDataCrosswalk**: Generates textual representation using the Citation Style Language (CSL), an XML-based format to describe the formatting of citations, notes and bibliographies.

#### ReferCrosswalk

The ReferCrosswalk allows to serialize the metadata of an item in a textual format that mirrors that of the configured template. ReferCrosswalk bean configuration example:

```
<bean class="org.dspace.content.integration.crosswalks.ReferCrosswalk"
id="referCrosswalkPersonJson">
  <property name="templateFileName" value="crosswalks/template/person-
json.template"/>
  <property name="mimeType" value="application/json; charset=UTF-8"/>
  <property name="fileName" value="person.json"/>
  <property name="entityType" value="Person"/>
  <property name="crosswalkMode" value="#{T(org.dspace.content.
crosswalk.CrosswalkMode).SINGLE_AND_MULTIPLE}"/>
  <property name="multipleItemsTemplateFileName" value="crosswalks
/template/persons-json.template"/>
  <property name="converter" ref="jsonValueConverter" />
  <property name="linesPostProcessor" ref="jsonPostProcessor" />
</bean>
```

In the example shown, a ReferCrosswalk is configured to export items in xml format according to the template named crosswalks/template /person-xml.template. The configuration also indicates the template used to export multiple items and a converter to process the values obtained from the item before inserting them into the xml itself. Specifically, the properties to be configured are:

- **templateFileName**: the path of the template to use relative to the DSpace configuration folder
- **mimeType:** the format of the file obtained by processing the item; it should be consistent with the configured template.
- **fileName**: the default name of the file that can be generated starting from the ReferCrosswalk result
- **entityType**: the type of the items that can be processed by this instance of the ReferCrosswalk
- **crosswalkMode:** indicates whether the instance being configured can be used for single export (CrosswalkMode.SINGLE), multiple export (CrosswalkMode.MULTIPLE) or for both (CrosswalkMode.SINGLE_AND_MULTIPLE). If not specified, CrosswalkMode.SINGLE is considered.
- **multipleItemsTemplateFileName**: the template path to be used to process multiple items; if not specified, the configured instance will not support multiple export
- **converter**: implementation of org.springframework.core.convert.converter.Converter<S,T> which allows to process the values to be entered in the xml. For example thi converter can be used to escape the special characters of the specific format.
- **linesPostProcessor**: implementation of java.util.function.Consumer<List<String>> to process all the lines of the export result before actually writing them to the outputstream

The template that is used to produce the result in a given format is a text file of many lines in which can be placed a series of **placeholders**: the result of the process corresponds to a file similar to the template in which the data relative to the processed items are inserted instead of these placeholders. Each line of the template can contain at most one placeholder; in case one line does not contain a placeholder this line will be reported identical in the generated result. If a placeholder needs to be replaced by multiple values (for example due to multiple values of a metadata) the entire row is duplicated for each value to be written.

The placeholders are marked with the @ and depending on the type the effect on the output may be different. There are 5 type of placeholders:

- **metadata**: can be used to indicate that the specified metadata value must be entered instead of the placeholder. The syntax of this placeholder is **@<metadata-field>@**, where <metadata-field> represents a metadata field with the various sections divided by a period. Examples: @dc.title@, @dc.date.issued@
- **metadata-group**: placeholder with which some lines of the template can be delimited to indicate that the whole section must be repeated for each set of nested metadata identified. The syntax of this placeholder is **@group.<metadata-field>.start@** to delimit the beginning of the section to be replicated and **@group.<metadata-field>.end@** to indicate the end, where <metadata-field> is the metadata representing the group with its various sections separated by "-".

```
@group.dc-contributor-author.start@
<Author>
    <DisplayName>@dc.contributor.author@</DisplayName>
    <Affiliation>
      <OrgUnit>
        <Name>@oairecerif.author.affiliation@</Name>
      </OrgUnit>
    </Affiliation>
</Author>
@group.dc-contributor-author.end@
```

- **virtual field**: placeholder to be replaced with the results of the specified virtual field. The syntax of this placeholder is **@virtual.<name>.<qualifiers>@,** where name represents the virtual field identifiers and the qualifiers represents a set of info usefull for the virtual field processing divided by period.
  For more details see the **Virtual Field** section.

```
<Type>@virtual.mapConverter.fundingTypes.dc-type@</Type>
```

- **relation**: placeholder with which some lines of the template can be delimited to indicate that the whole section must be repeated for each item which has a specific relationship with the item being written. The syntax of this placeholder is **@relation.<relationName>.start@** to delimit the beginning of the section to be replicated and **@relation.<relationName>.end@** to indicate the end, where <relationName> could be:
  - the metadata that contains the relationship with the other item through authority, with the various sections separated by "-"
  - the last section of one of the discovery configuration named RELATION.<entityType>.<relationName>, where entityType is the type of the item being written.

**Please note:** in the lines placed between the relation placeholders of start and end, the references are no longer made to the original item being written but to the related items identified: any metadata to be written will therefore be read by these last items and not by the original item.

```
@relation.oairecerif-funder.start@
<OrgUnit id="@virtual.id@">
        <Name>@dc.title@</Name>
        <Acronym>@oairecerif.acronym@</Acronym>
</OrgUnit>
@relation.oairecerif-funder.end@
```

- **if**: placeholder with which some lines of the template can be delimited to indicate that the whole section must be printed or not based on a condition evaluation. The syntax of this placeholder is
  **@if.[not.]<conditonName>.[qualifiers.]start@** to delimit the beginning of the section to be replicated and **@if.[not.]<condtionName>.[qualifiers.]end@** to indicate the end, where:
  - not is an optional section to negate the whole evaluation result
  - qualifiers are a set of data that can be used to evaluate the condition, separated by period

- conditonName is the name of the particular **condition evaluator** to be used to evaluate the condition. For more details see the **C ondition Evaluator** section.

In addition to the previous types of placeholders, which can only be used in the template for the single export, there is the placeholder **@item. template@** that can only be used in the template for multiple export to indicate the point in which to insert, for each item, the single template appropriately filled. Example:

```
{
  "persons": [
    @item.template@,
  ]
}
```

#### DocumentCrosswalk

DocumentCrosswalk allows to produce a document with the configured format (such as pdf or rtf) starting from a single item. The **Apache FOP Project** is used to produce the document: is a print formatter driven by **XSL formatting objects** (**XSL-FO**) and an output independent formatter. It is a Java application that reads a formatting object (FO) tree and renders the resulting pages to a specified output. So using this project is it possible to create a document starting from an XSL template and an xml file containing the information to be printed. Currently the DocumentCrosswalk can be used to export only single item.
An example of DocumentCrosswalk bean configuration is the following:

```
<bean class="org.dspace.content.integration.crosswalks.
DocumentCrosswalk" id="pdfCrosswalkPerson">
  <property name="templateFileName" value="crosswalks/template/person-
template.xsl"/>
  <property name="fileName" value="person.pdf"/>
  <property name="mimeType" value="application/pdf"/>
  <property name="entityType" value="Person"/>
  <property name="referCrosswalk" ref="referCrosswalkPersonXml"/>
</bean>
```

In the example shown, a DocumentCrosswalk is configured to export item in pdf format according to the template named crosswalks/template /person-template.xsl. Specifically, the properties to be configured are:

- **templateFileName:** the path of the template to use relative to the DSpace configuration folder
- **fileName**: the default name of the file that can be generated starting from the DocumentCrosswalk result
- **mimeType**: the format of the file obtained by processing the item
- **entityType**: the type of the items that can be processed by this instance of the DocumentCrosswalk
- **referCrosswalk:** reference to the ReferCrosswalk to be used to generate the xml representation of the item. This xml representation will then be used by the XSLT transformation to generate the file in XSL-FO format with which the document will be produced.

Example of XSL-FO template that print the title and the description of a Project:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.1" xmlns:xsl="http://www.w3.org/1999/XSL
/Transform" xmlns:fo="http://www.w3.org/1999/XSL/Format" xmlns:pt="
https://www.openaire.eu/cerif-profile/vocab/COAR_Publication_Types"
exclude-result-prefixes="fo">
        <xsl:template match="Project">
                <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
                        <fo:layout-master-set>
                                <fo:simple-page-master master-name="
```

```
simpleA4" page-height="29.7cm" page-width="24cm" margin-top="2cm"
margin-bottom="2cm" margin-left="1cm" margin-right="1cm">
                                        <fo:region-body />
                                </fo:simple-page-master>
                        </fo:layout-master-set>
                        <fo:page-sequence master-reference="simpleA4">
                                <fo:flow flow-name="xsl-region-body">
                                 <fo:block margin-bottom="5mm" padding="
2mm">
                                                        <fo:block font-size="
26pt" font-weight="bold" text-align="center" >
                                                                <xsl:value-of
select="Title" />
                                                        </fo:block>
                                                </fo:block>
                                           <fo:block font-size="10pt" space-
after="5mm" text-align="justify" margin-top="5mm" >
                                                        <xsl:value-of select="
Abstract" />
                                                </fo:block>
                                </fo:flow>
                        </fo:page-sequence>
                </fo:root>
        </xsl:template>
</xsl:stylesheet>
```

#### TabularCrosswalk

The TabularCrosswalk is an abstract class that allows to export multiple items in tabular format. In particular, its extensions CsvCrosswalk and XlsCrosswalk content to produce files in csv and xls format respectively. The values to be shown in these tabular formats with respective headings can be configured through files that act as **templates**: the format of these files is similar to a common properties file in which the key represents the header of the field to be shown and the value represents its value. The values can be:

- a **metadata field**
- a **virtual field** with the syntax **virtual.<name>.<qualifiers>** (similar to the syntax used in the template for the ReferCrosswalk but without the delimiters @)
- a **group of nested metadata** expressed with the syntax **group.<metadata-field>**, where <metadata-field> represent the metadata group with its various sections separated by "-"

Example of template for tabular export:

```
Title = dc.title
Type = virtual.mapConverter.coarTypes.dc-type
Authors = group.dc-contributor-author
```

If a metadataField has several values, they are concatenated with a configurable character (|| by default).

All nested metadata belonging to a group are identified by reading the submission configuration. The various groups are concatenated with a configurable character (|| by default) and the various nested metadata within the group are concatenated with an additional configurable character (/ by default). Considering the template shown above, a possible excel product could have the following content:

| Title | Type | Authors |
|---|---|---|
| Title1\|\|Title2 | http://purl.org/coar/resource_type/c_e9a0 | Walter White\|\|Jesse Pinkman |
| | | |

| | | |
|---|---|---|
| Title3 | http://purl.org/coar/resource_type/c_7ad9 | John Smith/Company\|\|Edward Red |

Instances of the XlsCrosswalk and CsvCrosswalk classes can be configured as follows:

```
<bean class="org.dspace.content.integration.crosswalks.CsvCrosswalk"
id="csvCrosswalkPerson">
  <property name="templateFileName" value="crosswalks/template/person-
table.template"/>
  <property name="fileName" value="persons.csv"/>
  <property name="entityType" value="Person"/>
  <property name="crosswalkMode" value="#{T(org.dspace.content.
crosswalk.CrosswalkMode).MULTIPLE}"/>
</bean>

<bean class="org.dspace.content.integration.crosswalks.XlsCrosswalk"
id="xlsCrosswalkPerson">
  <property name="templateFileName" value="crosswalks/template/person-
table.template"/>
  <property name="fileName" value="persons.xls"/>
  <property name="sheetName" value="Persons"/>
  <property name="entityType" value="Person"/>
  <property name="crosswalkMode" value="#{T(org.dspace.content.
crosswalk.CrosswalkMode).MULTIPLE}"/>
</bean>
```

Specifically, the common properties to be configured are:

- **templateFileName:** the path of the template to use relative to the DSpace configuration folder
- **fileName**: the default name of the file that can be generated starting from the DocumentCrosswalk result
- **entityType**: the type of the items that can be processed by this instance of the DocumentCrosswalk
- **crosswalkMode:** indicates whether the instance being configured can be used for single export (CrosswalkMode.SINGLE), multiple export (CrosswalkMode.MULTIPLE) or for both (CrosswalkMode.SINGLE_AND_MULTIPLE).

In addition, the beans of the XlsCrosswalk class have the following configurable properties:

- **sheetName:** the name of the sheet in which to insert the data

For the csv, the following properties allow you to configure the various separator characters:

- **crosswalk.csv.separator.values:** separator between the values of the same metadata
- **crosswalk.csv.separator.nested-values:** separator between groups of nested metadata
- **crosswalk.csv.separator.inside-nested:** separator between nested metadata of same group
- **crosswalk.csv.separator.fields**: separator between values (comma by default)

For the xls, the following properties allow you to configure the various separator characters:

- **crosswalk.xls.separator.values:** separator between the values of the same metadata
- **crosswalk.xls.separator.nested-values:** separator between groups of nested metadata
- **crosswalk.xls.separator.inside-nested:** separator between nested metadata of same group

#### CSLItemDataCrosswalk

The CSLItemDataCrosswalk class allows to use of the **Citation Style Language** to export **Publication** type items in different formats. This Crosswalk implementation use the **citeproc-java** API to export publications in a certain output format and style. Both the format and the style are configurable among those supported by the API. Instances of the CSLItemDataCrosswalk classes can be configured as follows:

```
<bean class="org.dspace.content.integration.crosswalks.
CSLItemDataCrosswalk" id="referCrosswalkBibtex">
```

```
  <property name="style" value="bibtex.csl"/>
  <property name="mimeType" value="application/x-bibtex; charset=UTF-8"
/>
  <property name="format" value="text"/>
  <property name="fileName" value="references.bib"/>
  <property name="entityType" value="Publication" />
  <property name="crosswalkMode" value="#{T(org.dspace.content.
crosswalk.CrosswalkMode).SINGLE}"/>
</bean>
```

Specifically, the properties to be configured are:

- **style:** the CSL style to be adopted (select one from the 9000+ styles provided by CitationStyles.org or set the relative path of a custom csl file in the dspace config dir)
- **mimeType:** the output mime type
- **format**: the output formats. citeproc-java supports several formats, the most common ones are "html" and text" but you can also use "asciidoc", "fo", and "rtf".
- **fileName**: the default name of the file that can be generated starting from the DocumentCrosswalk result
- **entityType**: the entity type of the items that the configured crosswalk can process
- **crosswalkMode:** indicates whether the instance being configured can be used for single export (CrosswalkMode.SINGLE), multiple export (CrosswalkMode.MULTIPLE) or for both (CrosswalkMode.SINGLE_AND_MULTIPLE). If not specified, CrosswalkMode.SINGLE is considered.

The CSL processor needs an implementation of **de.undercouch.citeproc.ItemDataProvider** to generate the publications in the specified output. The implementation used in the CSLItemDataCrosswalk is the custom implementation **org.dspace.content.integration.crosswalks.csl.DSpaceListItemDataProvider** which allows to map the item used in DSpace with all its metadata in objects of the class **CSLItemData** which are used by the citeproc API to produce citations.

A prototype bean of the DSpaceListItemDataProvider class is configured in the Spring **csl-citation.xml** context configuration file with the mapping between the CSLItemData fields and the DSpace item metadata.

#### Virtual Field

A virtual field is a field whose values are not obtained directly by reading the values of a specific item metadata but are calculated with an additional configurable logic. It allow to add additional logic to the replacement of a placeholder present in a template. To add a new virtual field to be used in the ReferCrosswalk and TabularCrosswalk templates should be provided a new implementation of the **org.dspace.content.integration.crosswalks.virtualfields.VirtualField** interface.
The set of virtual fields that can be used in the templates must be configured through a bean of the **org.dspace.content.integration.crosswalks.virtualfields.VirtualFieldMapper** type that contains a map between the names of the virtual fields and the instances of the virtual fields themselves. This Mapper is currently configured into the **crosswalks.xml** file:

```
<bean class="org.dspace.content.integration.crosswalks.virtualfields.
VirtualFieldMapper">
  <constructor-arg>
    <map>
      <entry key="id" value-ref="virtualFieldId" />
      <entry key="reftype" value-ref="virtualFieldRefererType" />
      <entry key="authors" value-ref="virtualFieldAuthors" />
    </map>
  </constructor-arg>
</bean>

<bean class="org.dspace.content.integration.crosswalks.virtualfields.
VirtualFieldId" id="virtualFieldId"/>
<bean class="org.dspace.content.integration.crosswalks.virtualfields.
VirtualFieldRefererType" id="virtualFieldRefererType"/>
<bean class="org.dspace.content.integration.crosswalks.virtualfields.
VirtualFieldAuthors" id="virtualFieldAuthors"/>
```

Currently the provided implementations of the VirtualField interface are:

- org.dspace.content.integration.crosswalks.virtualfields.**VirtualFieldAuthority** returns all metadata authorities of the given metadata field for the specific item
- org.dspace.content.integration.crosswalks.virtualfields.**VirtualFieldAuthors** returns all dc.contributor.author associated with the given item by concatenating the names with and
- org.dspace.content.integration.crosswalks.virtualfields.**VirtualFieldBitstream** creates a temporary file with the contents of a bitstream of a specific bundle. If a different format of the same bitstream is present in the PREVIEW bundle then the content of that bitstream is written in the temporary file. If the bitstream selected for writing is a non-jpeg image then the written image is converted to jpeg format.
- org.dspace.content.integration.crosswalks.virtualfields.**VirtualFieldDateFormatter** returns either the current date or the value of a metadata formatted with a certain pattern
- org.dspace.content.integration.crosswalks.virtualfields.**VirtualFieldId** returns the id of the given item
- org.dspace.content.integration.crosswalks.virtualfields.**VirtualFieldPersonName** returns either the first or last name related to the given item, taking the name from the dc.title

#### Condition Evaluator

A condition evaluator is a class which allows to evaluate certain conditions on a item while parsing a template to decide whether or not to include certain lines in the final result. A condition evaluator must extends the abstract class **org.dspace.content.integration.crosswalks.evaluators.ConditionEvaluator** and it must be mapped with his name by inserting it into the map handled by the bean of class **org.dspace.content.integration.crosswalks.evaluators.ConditionEvaluatorMapper** configured into the **crosswalks.xml**.

```
<bean class="org.dspace.content.integration.crosswalks.evaluators.
ConditionEvaluatorMapper" id="conditionEvaluatorMapper">
  <constructor-arg name="conditionEvaluators">
    <map>
      <entry key="authority" value-ref="authorityConditionEvaluator" />
    </map>
  </constructor-arg>
</bean>

<bean class="org.dspace.content.integration.crosswalks.evaluators.
AuthorityNotBlankCondition" id="authorityConditionEvaluator"/>
```

#### Items export configuration

The export functionality can be configured using the context configuration file named **crosswalks.xml**. The crosswalks.xml configuration file contains the configuration of the following beans:

- an instance of the **StreamDisseminationCrosswalkMapper** class that contains the mapping between the instances of the various StreamDisseminationCrosswalk used for the export and a name that identifies the format of the export itself. Of these classes for the export item, only those that also implement the ItemExportCrosswalk interface will be considered.
- the various instances of **StreamDisseminationCrosswalk** (Refer Crosswalk, Document Crosswalk etc ...)
- an instance of the **VirtualFieldMapper** class that contains the mapping between the instances of the various VirtualField usable in the templates and a name that identifies them.
- the various instances of **VirtualField**
- an instance of the **ConditionEvaluatorMapper** class that contains the mapping between the instances of the various ConditionEvaluator usable in the ReferCrosswalk templates and a name that identifies them.
- the various instances of **ConditionEvaluator**

Example of StreamDisseminationCrosswalkMapper's configuration:

```
<bean class="org.dspace.content.integration.crosswalks.
StreamDisseminationCrosswalkMapper">
  <constructor-arg>
    <map>
      <entry key="bibtex" value-ref="referCrosswalkBibtex"></entry>
```

```
      <entry key="endnote" value-ref="referCrosswalkEndnote"></entry>
      <entry key="publication-xml" value-ref="
referCrosswalkPublicationXml"></entry>
      <entry key="publication-pdf" value-ref="pdfCrosswalkPublication"><
/entry>
      <entry key="publication-csv" value-ref="csvCrosswalkPublication"><
/entry>
      <entry key="publication-xls" value-ref="xlsCrosswalkPublication"><
/entry>

      <entry key="person-xml" value-ref="referCrosswalkPersonXml"><
/entry>
      <entry key="person-json" value-ref="referCrosswalkPersonJson"><
/entry>
      <entry key="person-pdf" value-ref="pdfCrosswalkPerson"></entry>
      <entry key="person-rtf" value-ref="rtfCrosswalkPerson"></entry>
      <entry key="person-csv" value-ref="csvCrosswalkPerson"></entry>
      <entry key="person-xls" value-ref="xlsCrosswalkPerson"></entry>
    </map>
  </constructor-arg>
</bean>
```

#### Single item export script

The export of a single item can be started using the script called **item-export**, both via REST and via CLI. The configurable options to start the process are:

| Name | Description | Required |
|---|---|---|
| **i (id)** | the ID of the item to export. It is required | Yes |
| **f (format)** | the format in which the item is to be exported. It must match one of the keys present in the map of the StreamDisseminationCrosswalkMapper bean | Yes |
| **n (name)** | the name of the file to generate | No |

#### Bulk items export script

The export of a multiple item can be started using the script called **bulk-item-export**, both via REST and via CLI. The export considers not only the archived items but also the workspace and workflow items. The configurable options to start the process are:

| Name | Description | Required |
|---|---|---|
| **f (format)** | the format in which the item is to be exported. It must match one of the keys present in the map of the StreamDisseminationCrosswalkMapper bean | Yes |
| **t (type)** | the entity type of the items to export. This type must be consistent with the type of format chosen with the -f option | Yes |
| **q (query)** | the Solr query to perform to find the items to be exported | No |
| **sf (filters)** | the filters to apply to the Solr query with the syntax <filter-name>=<value> with the possibility to concatenate multiple filters with the & character | No |
| **c (configuration)** | the discovery configuration to use for the Solr query | No |
| **s (scope)** | the scope to search into (uuid of one community, collection or item in case of RELATION configuration) | No |
| **so (sort)** | the sort field and order in the format <sort-field>,<order> | No |

The Solr query is composed by adding an additional filter to the filters passed in input, setting the entityType with the value specified through the type option.

### OAI-PMH Data Provider

DSpace-CRIS comes with an OAI-PMH data provider endpoint, available at url

https://<dspace-cris-base-url/server/oai

This endpoint can be enabled or disabled via configuration, using property “oai.enabled”

## ENRICH DATA SOURCE

Metadata xml returned by OAI-PMH Provider gets its data from ‘oai’ solr core. Its output is built on top of “item.compile” value of each solr document returned by a proper query, and it is trasformed via xsl rules as defined in xoai.xml file.

Interface `org.dspace.xoai.app.XOAIItemCompilePlugin` is available. By implementing this plugin interface, it is possible to enrich content of “item.compile” field, by adding custom metadata.

Each custom implementation of this interface must be initialized as Spring bean via proper configuration, in order to contribute its own content to “item.compile” field,

#### CERIF EXPORT

It is possible to enrich “item.compile” field with cerif compliant representation of a given Item. To reach this goal, `XOAICerifItemCompilePlugin` implementation of `org.dspace.xoai.app.XOAIItemCompilePlugin` is provided.

This class uses stream dissemination logic defined and configured in crosswalks.xml file to create an xml representation of the item. Multiple Spring Beans with type of this class can be instantiated. Each Spring Bean, configured in oai.xml file, must contain:

- generator: type of generator to be used (xml, xml-cerif, available generators are the one defined in crosswalks.xml for each DSpace-CRIS entity type)
- fieldName: name of field to be created and added to “item.compile”.
- ePerson: (optional) if set with an Eperson email, item exported contains all metadata such person can see, even not public ones.

For example, with this default configuration

```
<bean id="xoaiCerifGenerator" class="org.dspace.xoai.app.
XOAICerifItemCompilePlugin">
    <property name="generator" value="cerif-xml"/>
    <property name="fieldName" value="openaire"/>
</bean>
```

an element “cerif.openaire” will be added to “item.compile” field, and its content will be generated depending on Entity type, and using generator named cerif-xml xml.

while this other one

```
<bean id="xoaiCerifGenerator" class="org.dspace.xoai.app.
XOAICerifItemCompilePlugin">
    <property name="generator" value="cerif-xml"/>
    <property name="fieldName" value="openaire"/>
    <property name="ePerson" value="john.doe@example.com"/>
</bean>
```

an element “cerif.openaire” will be added to “item.compile” field, and its content will be generated depending on Entity type, and using generator named cerif-xml. Generated xml will contain all metadata values user “john.doe@example.com” is allowed to access.

To have the `XOAICerifItemCompilePlugin` configuration properly working, “generator” value should match with generators defined in crosswalks.xml file, more specifically, generator value should be a suffix of a generator key defined in crosswalks.xml, for example, given above configuration, generators entry keys with id ending in “cerif-xml”, i.e. “publication-cerif-xml”, “person-cerif-xml”, etc. If such generators are not defined, generated oai xml will be incomplete.

## Logical Item filtering

> This functionality was originally developed by the Library of Code and it is now part of the official DSpace code base since version 7.1. It has been adopted in DSpace-CRIS since the first version 7 release (2021.01.00) and can be used in several area of the system to constraints functionalities and behaviors to items with specific characteristic

Inspired by the powerful conditional filters in XOAI, this component offers a simple but flexible way to write logical statements and tests, and use the results of those tests in other services or DSpace code.

#### LogicalStatement

LogicalStatement is a simple interface ultimately implemented by all the other interfaces and classes described below. It just requires that a class implements a `Boolean getResult(context, item)` method.

#### Filters

Filters are at the root of any test definition, and it is the filter ID that is used to load up the filter in spring configurations for other services, or with DSpace Service Manager.

A filter bean is defined with a single “statement” property - this could be an Operator, to begin a longer logical statement, or a Condition, to perform a simple check.

There is one simple implementation of Filter included - DefaultFilter.

#### Operators

Operators are the basic logical building blocks that implement operations like AND, OR, NOT, NAND and NOR. An Operator can contain any number of other Operators or Conditions.

So statements like this can be created:

(**x** AND (**y** OR **z**) AND **a** AND (**b** OR NOT(**d**))

#### Conditions

Conditions are where the actual DSpace item evaluation code is written. A condition accepts a Map<String, Object> map of parameters. Conditions don’t contain any other LogicalStatement classes – the are at the bottom of the chain.

A condition could be something like MetadataValueMatchCondition, where a regex pattern and field name are passed as parameters, then tested against actual item metadata. If the regex matches, the boolean result is true.

Typically, commonly used Conditions will be defined as beans elsewhere in the spring config and then referenced inside Filters and Operators to create more complex statements.

#### Configuring Filters in Spring

Conditions, Operators and Filters are all defined in `${dspace}/config/spring/api/item-filters.xml`

Here’s a complete example of a filter definition that implements the same rules as the XOAI openAireFilter. As an exercise, some statements will be defined as beans externally, and some will be defined inline as part of the filter.

**New Condition: driver-document-type_condition**

This condition creates a new bean to test metadata values. In this case, we’re implementing “ends with” for a list of type patterns.

```
<!-- dc.type ends with any of the listed values, as per XOAI
"driverDocumentTypeCondition" -->
    <bean id="driver-document-type_condition"
          class="org.dspace.content.logic.condition.
MetadataValuesMatchCondition">
        <property name="parameters">
```

```
            <map>
                <entry key="field" value="dc.type" />
                <entry key="patterns">
                    <list>
                        <value>article$</value>
                        <value>bachelorThesis$</value>
                        <value>masterThesis$</value>
                        <value>doctoralThesis$</value>
                        <value>book$</value>
                        <value>bookPart$</value>
                        <value>review$</value>
                        <value>conferenceObject$</value>
                        <value>lecture$</value>
                        <value>workingPaper$</value>
                        <value>preprint$</value>
                        <value>report$</value>
                        <value>annotation$</value>
                        <value>contributionToPeriodical$</value>
                        <value>patent$</value>
                        <value>dataset$</value>
                        <value>other$</value>
                    </list>
                </entry>
            </map>
        </property>
    </bean>
```

**New Condition: item-is-public_condition**

This condition accepts group and action parameters, then inspects item policies for a match - if the supplied group can perform the action, the result is true.

```
<bean id="item-is-public_condition"
          class="org.dspace.content.logic.condition.
ReadableByGroupCondition">
        <property name="parameters">
            <map>
                <entry key="group" value="Anonymous" />
                <entry key="action" value="READ" />
            </map>
        </property>
</bean>
```

**New Filter: openaire_filter**

Here is the full definition for the OpenAIRE filter.

The first statement is an And Operator, with many sub-statements – four Conditions, and an Or statement.

The first two statements in this Operator are simple Conditions defined in-line, and just check for a non-empty value in a couple of metadata fields.

The third statement is a reference to the document type Condition we made earlier:
`<ref bean="driver-document-type_condition" />`

The fourth statement is another Operator, in this case an Or Operator with two Conditions (the is-public Condition we defined earlier, and an in-line definition of as “is-withdrawn” Condition)

The fifth statement is an in-line definition of a Condition that checks dc.relation metadata for a valid OpenAIRE identifier.

So the full logic implemented is:

`(has-title AND has-author AND has-driver-type AND (is-public OR is-withdrawn) AND has-valid-relation)`

```
<!-- An example of an OpenAIRE compliance filter based on the same
rules in xoai.xml
      some sub-statements are defined within this bean, and some are
referenced from earlier definitions
-->
<bean id="openaire_filter" class="org.dspace.content.logic.
DefaultFilter">
    <property name="statement">
        <bean class="org.dspace.content.logic.operator.And">
            <property name="statements">
                <list>
                    <!-- Has a non-empty title -->
                    <bean id="has-title_condition"
                          class="org.dspace.content.logic.condition.
MetadataValueMatchCondition">
                        <property name="parameters">
                            <map>
                                <entry key="field" value="dc.title" />
                                <entry key="pattern" value=".*" />
                            </map>
                        </property>
                    </bean>
                    <!-- AND has a non-empty author -->
                    <bean id="has-author_condition"
                          class="org.dspace.content.logic.condition.
MetadataValueMatchCondition">
                        <property name="parameters">
                            <map>
                                <entry key="field" value="dc.
contributor.author" />
                                <entry key="pattern" value=".*" />
                            </map>
                        </property>
                    </bean>
                    <!-- AND has a valid DRIVER document type (defined
earlier) -->
                    <ref bean="driver-document-type_condition" />
                    <!-- AND (the item is publicly accessible OR
withdrawn) -->
                    <bean class="org.dspace.content.logic.operator.Or">
                        <property name="statements">
```

```
                            <list>
                                <!-- item is public, defined earlier -->
                                <ref bean="item-is-public_condition" />
                                <!-- OR item is withdrawn, for
tombstoning -->
                                <bean class="org.dspace.content.logic.
condition.IsWithdrawnCondition">
                                    <property name="parameters"><map><
/map></property>
                                </bean>
                            </list>
                        </property>
                    </bean>
                    <!-- AND the dc.relation is a valid OpenAIRE
identifier
                          (starts with "info:eu-repo/grantAgreement/")
-->
                    <bean id="has-openaire-relation_condition"
                          class="org.dspace.content.logic.condition.
MetadataValueMatchCondition">
                        <property name="parameters">
                            <map>
                                <entry key="field" value="dc.relation"
/>
                                <entry key="pattern" value="^info:eu-
repo/grantAgreement/" />
                            </map>
                        </property>
                    </bean>
                </list>
            </property>
        </bean>
    </property>
</bean>
```

#### Running Tests on the Command Line

There is a launcher command that can arbitrarily run tests on an item or all items, eg.

`${dspace}/bin/dspace test-logic -f openaire_filter -i 123456789/100`

A simple `true` or `false` is printed for each item tested.

#### Using Filters in other Spring Services

The Filter beans can be referenced (or defined) in other services, for instance, here is adding the bean we configured earlier, as a `filterService` to a new `FilteredDOIIdentifierProvider`:

```
<bean id="org.dspace.identifier.DOIIdentifierProvider"
      class="org.dspace.identifier.FilteredDOIIdentifierProvider"
      scope="singleton">
    <property name="configurationService"
```

```
                ref="org.dspace.services.ConfigurationService" />
    <property name="DOIConnector"
                ref="org.dspace.identifier.doi.DOIConnector" />
    <property name="filterService"
                ref="openaire_filter"/>
</bean>
```

In the provider, we just define the property with the other services and class variables:

```
private Filter filterService;
```

And make sure there is a setter for it:

```
@Required
public void setFilterService(Filter filterService) {
    this.filterService = filterService;
}
```

Then you can actually run the tests with the service, like this:

```
try {
    Boolean result = filterService.getResult(context, (Item) dso);
    // do something with result
} catch(LogicalStatementException e) {
    // ... handle exception ...
}
```

In the TestLogicRunner, you can see a way to get the filters by name using the DSpaceServiceManager as well.

## Item validation

it is possible to validate the items in submission by establishing completely configurable rules. There are two different forms of validation:

- **submission step validation** validation concerning a single submission step
- **global validation** validation concerning the entire submission form

Currently the validation of the items is carried out, under the due conditions, during

- the submission of a new item
- bulk import from excel files
- import via OAI-PMH

using the org.dspace.validation.service.**ValidationService** implementation.

#### Submission step validation

The validations of the individual submission steps depend on the type of steps that compose the submission associated with a particular collection in which an item is to be submitted. Classes that allow this type of validation must implement the org.dspace.validation. SubmissionStepValidator interface and must be registered as a Spring context bean. Currently this interface has the following implementations:

- org.dspace.validation.**MetadataValidator** execute three validation check on fields validation (mandatory metadata missing, regex missing match and authority required metadata missing). This validation is associated with submission-form steps.
- org.dspace.validation.**LicenseValidator** check that the license has been grant for the inprogress submission looking for the presence of a license bitstream in the license bundle.

- org.dspace.validation.**UploadValidator** execute file required check validation

#### Global validation

it is possible to establish global rules that apply to multiple sections / metadata of the submission form. The classes that perform this validation must implement the org.dspace.validation.**GlobalSubmissionValidator** interface. Currently the only implementation is the org.dspace.validation.**LogicalStatementValidator** class which use a configured instance of org.dspace.content.logic.**Filter** type to perform a validation check. The basic concepts of this type of validation are:

- **filters** a filter bean is defined with a single “statement” property - this could be an Operator, to begin a longer logical statement, or a Condition, to perform a simple check. There is one simple implementation of Filter included - org.dspace.content.logic.**DefaultFilter**.
- **operators** operators are the basic logical building blocks that implement operations like AND, OR, NOT, NAND and NOR. An Operator can contain any number of other Operators or Conditions.

  So statements like this can be created: (x AND (y OR z) AND a AND (b OR NOT(d))
- **conditions** conditions are where the actual DSpace item evaluation code is written. A condition accepts a Map<String, Object> map of parameters. Conditions don’t contain any other LogicalStatement classes – the are at the bottom of the chain. A condition could be something like MetadataValueMatchCondition, where a regex pattern and field name are passed as parameters, then tested against actual item metadata. If the regex matches, the boolean result is true. Typically, commonly used Conditions will be defined as beans elsewhere in the spring config and then referenced inside Filters and Operators to create more complex statements.

Conditions, Operators and Filters are all defined in `${dspace}/config/spring/api/item-filters.xml`

Currently 3 LogicalStatementValidator are defined in the context, to perform the following validations:

- verify that a Person entity has at least one identifier set
- verify that a Peruvian cv Person has ubigeo set
- verify that Project entity without oa-mandate has policy url

These validations are defined in the `${dspace}/config/spring/api/addon-validation-services.xml`

```
<bean name="personHasAtLeastOneIdValidation" class="org.dspace.
validation.LogicalStatementValidator">
    <property name="errorKey" value="error.validation.personIdRequired"
/>
    <property name="metadataFields">
        <list>
            <value>person.identifier.orcid</value>
            <value>perucris.identifier.dni</value>
            <value>perucris.identifier.dina</value>
            <value>perucris.identifier.renacyt</value>
            <value>person.identifier.scopus-author-id</value>
            <value>person.identifier.rid</value>
        </list>
    </property>
    <property name="filter" ref="person-has-at-least-one-id_filter"/>
</bean>

<bean name="peruvianHasUbigeoSet" class="org.dspace.validation.
LogicalStatementValidator">
    <property name="errorKey" value="error.validation.ubigeoRequired"/>
    <property name="metadataFields">
        <list>
            <value>perucris.ubigeo</value>
        </list>
    </property>
    <property name="filter" ref="peruvian-cv-person-has-ubigeo_filter"/>
</bean>
```

```
<bean name="projectWithOaPolicyUrl" class="org.dspace.validation.
LogicalStatementValidator">
    <property name="errorKey" value="error.validation.
mandateUrlRequired"/>
    <property name="metadataFields">
        <list>
            <value>oairecerif.oamandate.url</value>
        </list>
    </property>
    <property name="filter" ref="project-without-oa-mandate-requires-
policy-url"/>
</bean>
```

### PreventMetadataSecurity projection

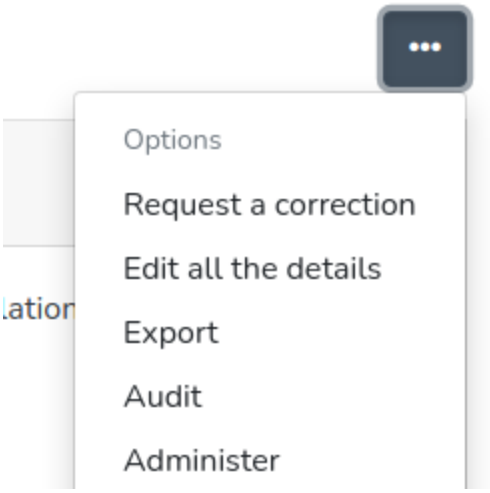
The `preventMetadataSecurity` projection is a particular “placeholder” projection. Having it as part of a request won’t add any particular data to the REST response.

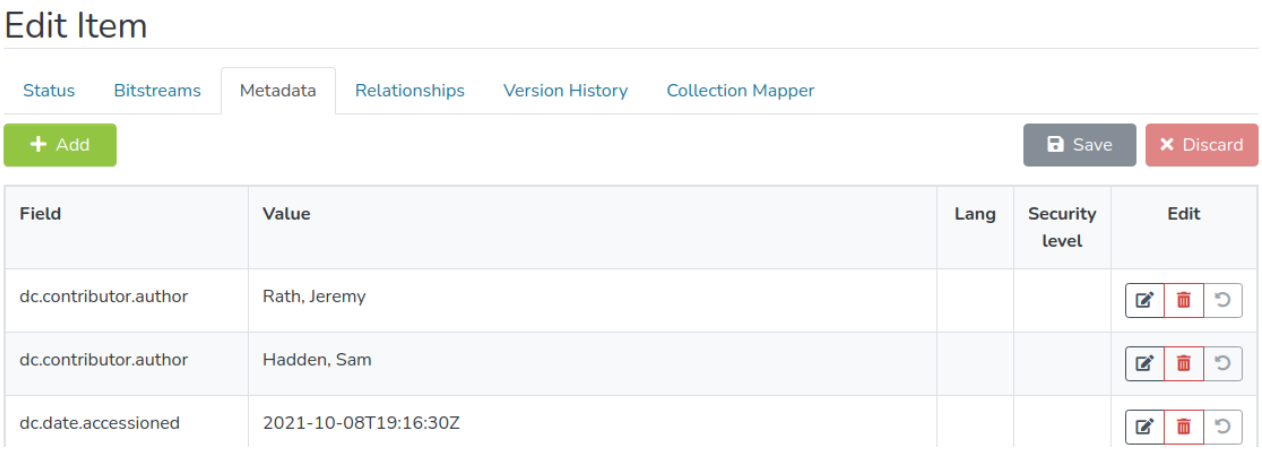
When this projection is included into the request, REST backend logic is instructed to skip the security evaluation, driven by the layout configuration, around each metadata composing the DSpaceObject to be returned.

In addition, no queries are performed to discover which metadata fields will be considered “public” during the building of the rest response. List of public fields is hard coded into `public-metadata.cfg` configuration file.

### Restrict Administer feature access

An user having write permissions on an Item, or on its holding collection / community, can administer (i.e. edit metadata values without any kind of validation) it, from “Administer” link of contextual menu.



It is possible to restrict this grant only to users members of a given group.

To enable this restriction, the uuid of the above mentioned group must be set in `edit.metadata.allowed-group` property.

If property is not set, no restrictions will be applied, and Administer action will be possible for every user having write permissions on DSpace Object

### Navbar

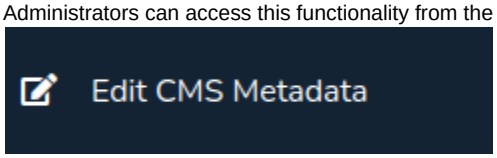
Environment properties:

- when property `layout.navbar.showCommunityCollection` is true, “Community and Collections” link is shown in the navbar; otherwise, it is shown in the admin sidebar.

### Home Page Customization - CMS metadata

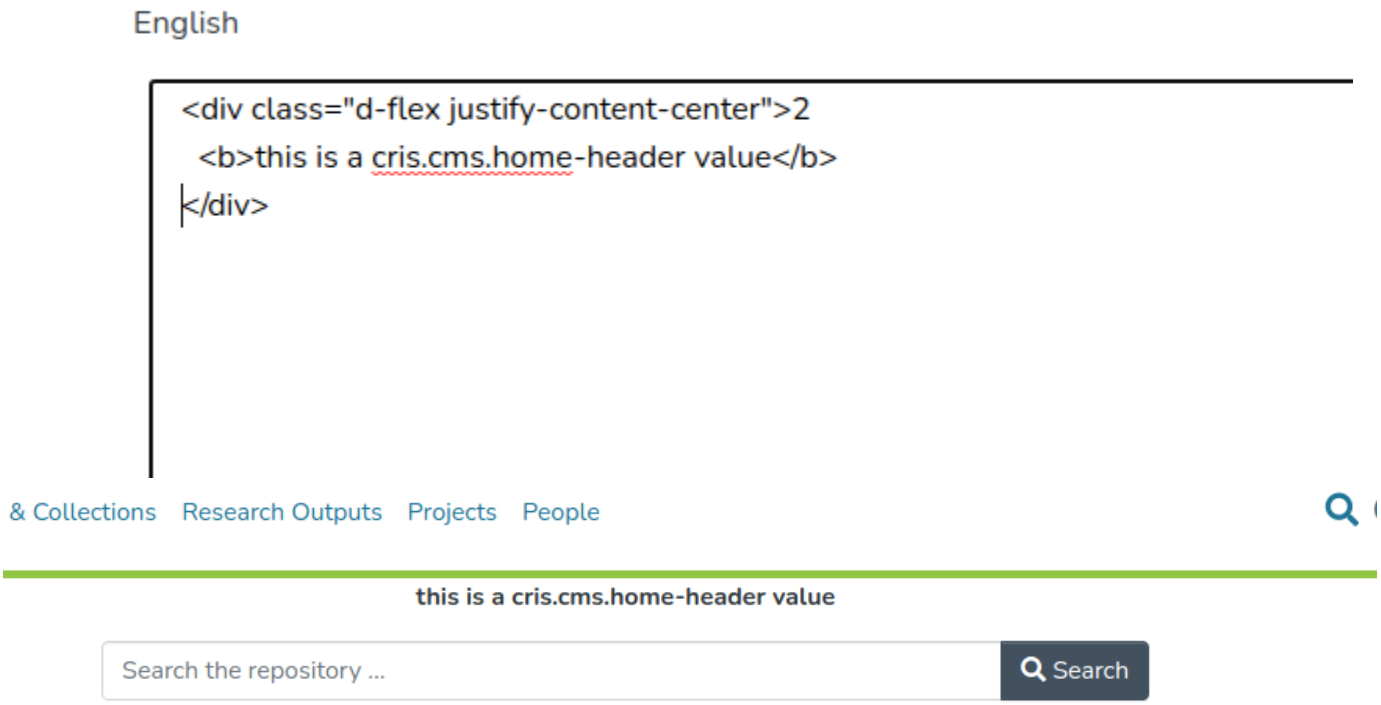
It is possible to add custom html to home page header, and news sections, by setting `cris.cms.home-header` and `cris.cms.home-news` metadata. Html code can be used to set metadata values.

Administrators can access this functionality from the navigation menu

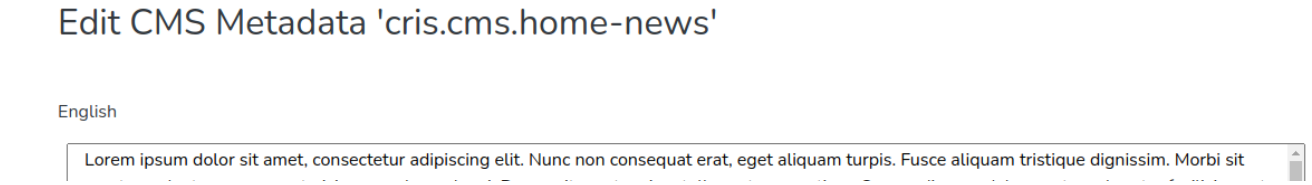



Metadata value can be different for each configured language in DSpace-CRIS 7 instance

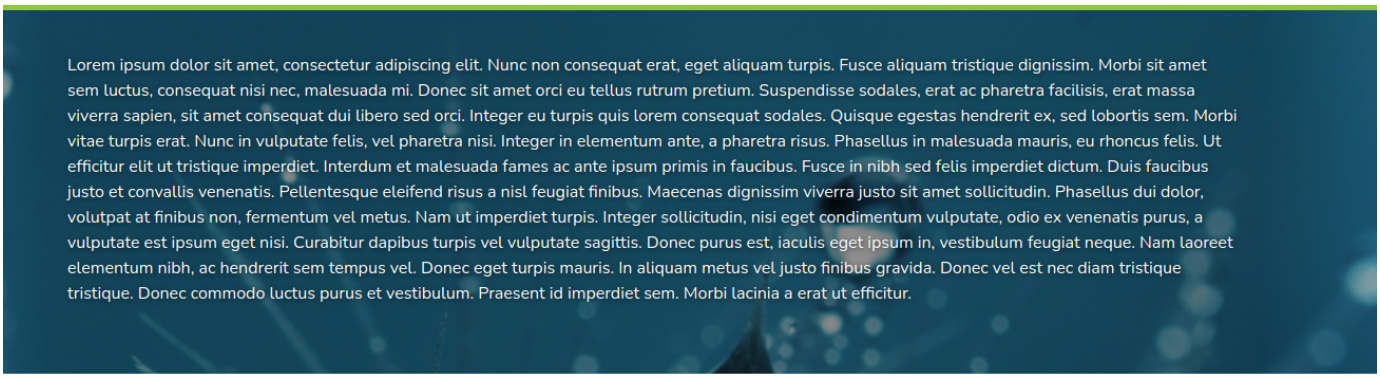
cris.cms.home-header allows to add a custom html text to home page header, this html will replace the header defined by theme in use.




cris.cms.home-news metadata is used to define a custom text in home news section, theme default background is preserved.



amet sem luctus, consequat nisi nec, malesuada mi. Donec sit amet orci eu tellus rutrum pretium. Suspendisse sodales, erat ac pharetra facilisis, erat massa viverra sapien, sit amet consequat dui libero sed orci. Integer eu turpis quis lorem consequat sodales. Quisque egestas hendrerit ex, sed lobortis sem. Morbi vitae turpis erat. Nunc in vulputate felis, vel pharetra nisi. Integer in elementum ante, a pharetra risus. Phasellus in malesuada mauris, eu rhoncus felis. Ut efficitur elit ut tristique imperdiet. Interdum et malesuada fames ac ante ipsum primis in faucibus. Fusce in nibh sed felis imperdiet dictum. Duis faucibus justo et convallis venenatis. Pellentesque eleifend risus a nisl feugiat finibus.




#### Custom CMS metadata

Further custom CMS metadata, whose value can be set by administrator by mean of Edit CMS metadata section, can be defined. They need to be added to metadata registry (REST side) and to cms.metadatalist array in angular environment configuration file (FrontEnd side).

Those custom metadata can be used together with Cris Sections configuration, this means that its value can be displayed either in the home page or in other explore sections.

By recalling one or many of those user-defined metadata in any cris section of cris-sections.xml configuration, the administrator can have their content displayed in a user-defined part of the home page or of an explore section.

For example, this xml bean, in `cris-sections.xml` configuration, related to a simple text component

```
<bean class="org.dspace.layout.CrisLayoutTextRowComponent">
  <property name="contentType" value="text-metadata"/>
  <property name="content" value="cris.cms.custom"/>
  <property name="style" value="style"/>
</bean>
```

will instruct page rendering logic to render a simple text among other parts of home page (in case of “site” section) / explore. Actual Content of this component will be the one which is defined within the `cris.cms.custom` metadata value. `cris.cms.custom` in this example is user defined, this means that needs to be added to registry and angular environment configuration as specified in previous paragraph.

Setting those metadata allows DSpace-CRIS7 administrators to customize part of the content home page or of a given explore section, or predefined home page parts like header and news sections, by editing a metadata value, without having to restart or re-compile the whole instance.

### Share Content

By means of the AddThis plugin it is possible to share DspaceCris7 content with common Social Media Platforms.

- Configuration
- User preferences

#### Configuration

To plug the feature you’ll need to follow these steps.

1) Create a Site in the AddThis website and retrieve the **SiteId** as described on the AddThis documentation.

The SiteId will be appended to generate the correct AddThis **script url:**

http://s7.addthis.com/js/300/addthis_widget.js#pubid=**<SiteId>**

2) Configure the **Angular environment** properties like below.

```
addThisPlugin: {
  siteId: '<SiteId>',
  scriptUrl: 'http://s7.addthis.com/js/300/addthis_widget.js#pubid=',
  socialNetworksEnabled: true
}
```

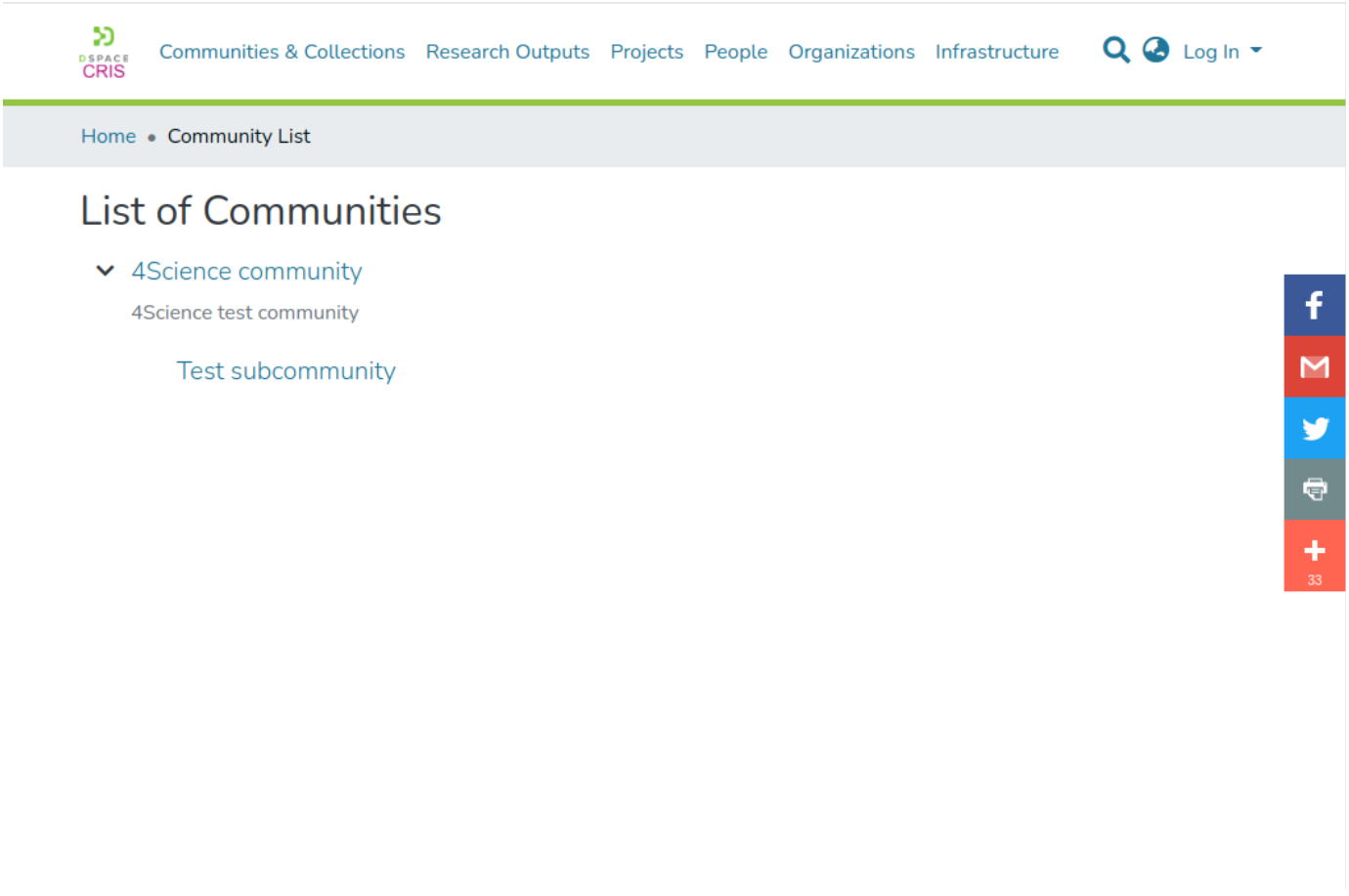
Use `socialNetworksEnabled` property to activate/deactivate globally the share content feature.

3) Configure the **Angular Router** to activate the feature in every desired route (you may want to include just informative pages in order to exclude administrative pages or edit pages).

Add `showSocialButtons: true` to the data object of each configured route

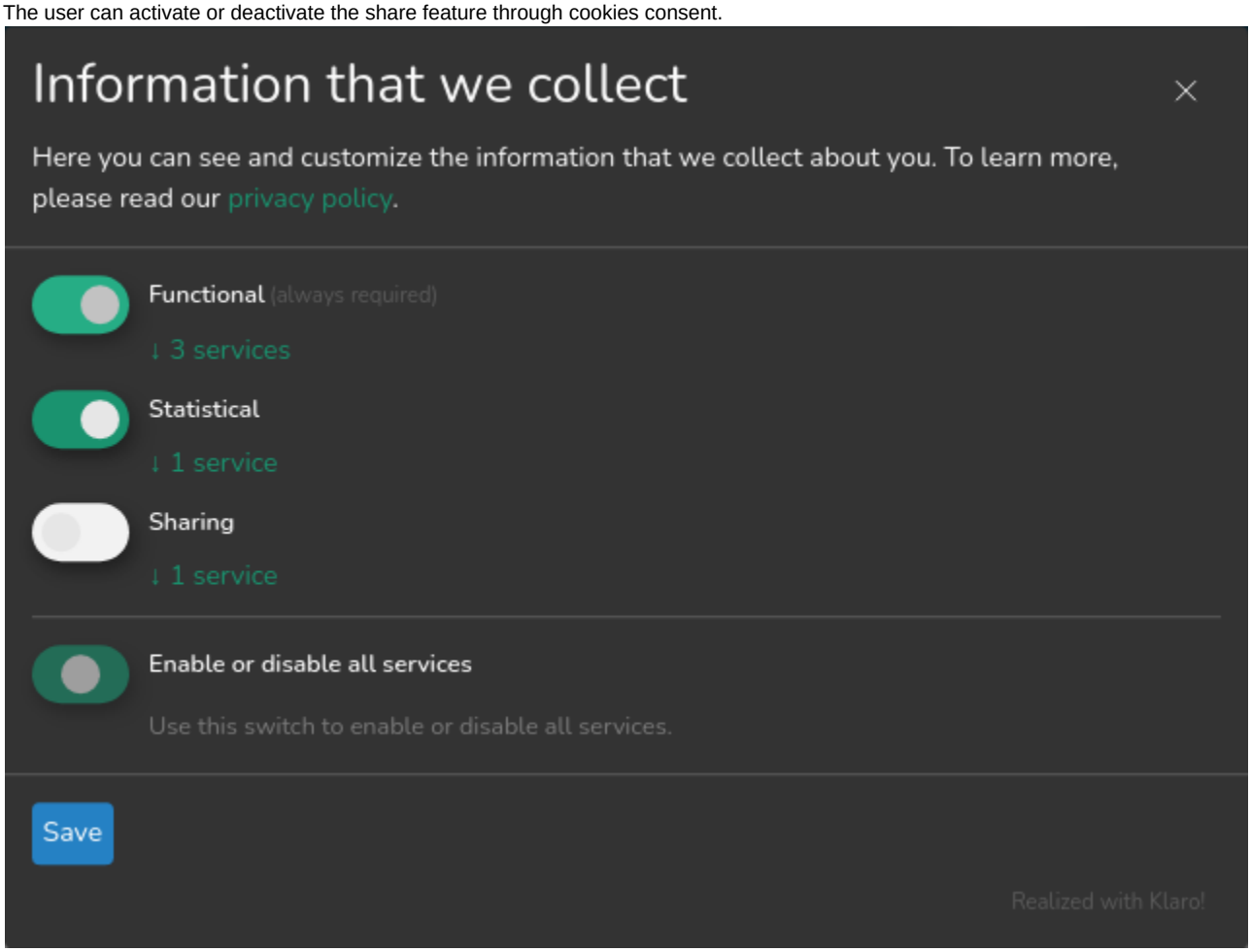
```
{
  path: ':id',
  data: {
    showSocialButtons: true
  }
}
```

The AddThis container element will then be positioned inside the page accordingly to the widget configuration.




User preferences

The user can activate or deactivate the share feature through cookies consent.




## Data Dictionary

| Table Name | Description | Table Fields | Primary Key | Linked tables | Linking tables |
|---|---|---|---|---|---|
| bitstream | Includes all the DSpace bitstreams related to DSpace Items, Communities (logo), Collections (logo and template), process (output) | "uuid" uuid [not null]<br>"bitstream_id" integer<br>"bitstream_format_id" integer<br>"checksum" "character varying(64)"<br>"checksum_algorithm" "character varying(32)"<br>"internal_id" "character varying(256)"<br>"deleted" boolean<br>"store_number" integer<br>"sequence_id" integer<br>"size_bytes" bigint<br>} | uuid | bitstream_format_id<br><br>bitstreamformatregistry. bitstream_format_id<br><br>uuid dspaceobject. uuid | bundle<br><br>bundle2bitstream<br><br>collection<br><br>checksum_history<br><br>process2bitstream<br><br>requestitem |
| bitstreamformatregistry | Includes the bitstream formats recognizable by DSpace | {<br>"bitstream_format_id" integer [not null]<br>"mimetype" "character varying(256)"<br>"short_description" "character varying (128)"<br>"description" text<br>"support_level" integer<br>"internal" boolean<br>} | "bitstream_format_id" | | bitstream |
| bundle | Includes bundles by which the bitstreams related to DSpace items are divided | {<br>"bundle_id" integer<br>"uuid" uuid [default: `gen_random_uuid()`]<br>"primary_bitstream_id" uuid<br>} | "uuid" | primary_bitstream_id bitstream.uuid<br><br>uuid dspaceobject. uuid | item2bundle |

| | | | | | |
|---|---|---|---|---|---|
| bundle2bitstream | Manages the relationship between bundle and bitstreams | {<br>"bitstream_order_legacy" integer<br>"bundle_id" uuid [not null]<br>"bitstream_id" uuid [not null]<br>"bitstream_order" integer [not null]<br>} | "bitstream_order"<br>"bundle_id"<br>"bitstream_id" | bitstream_id bitstream.uuid<br><br>bundle_id bundle.uuid | |
| checksum_history | List the result of the assetstore integrity check for a specific bitstream. | {<br>"check_id" bigint [not null]<br>"process_start_date" timestamp<br>"process_end_date" timestamp<br>"checksum_expected" "character varying"<br>"checksum_calculated" "character varying"<br>"result" "character varying"<br>"bitstream_id" uuid<br>} | "check_id" | result checksum_results.result_code<br><br>bitstream_id bitstream.uuid | |
| checksum_results | Dictionary table to provide a description of the result_code used in other table | {<br>"result_code" "character varying" [not null]<br>"result_description" "character varying"<br>} | "result_code" | | checksum_history |
| collection | Includes all the collections by which DSpace items are grouped | {<br>"collection_id" integer<br>"uuid" uuid [default: `gen_random_uuid()`]<br>"submitter" uuid<br>"template_item_id" uuid<br>"logo_bitstream_id" uuid<br>"admin" uuid<br>} | "uuid" | submitter eperson.uuid<br><br>template_item_id item.uuid<br><br>logo_bitstream_id item.uuid<br><br>uuid dspaceobject.uuid | item<br><br>collection2item<br><br>cwf_collectionrole<br><br>community2collection<br><br>harvested_collection<br><br>imp_record<br><br>subscription |
| collection2item | Manages the relationship between collections and items | {<br>"collection_id" uuid [not null]<br>"item_id" uuid [not null]<br>} | "collection_id"<br><br>"item_id" | collection_id collection.uuid<br><br>item_id item.uuid | |
| community | Includes communites and subcommunities by which collections are grouped | {<br>"community_id" integer<br>"uuid" uuid [default: `gen_random_uuid()`]<br>"admin" uuid<br>"logo_bitstream_id" uuid<br>} | "uuid" | admin epersongroup.uuid<br><br>logo_bitstream_id item.uuid<br><br>uuid dspaceobject.uuid | community2community<br><br>community2collection |
| community2collection | Manages the relationship between collections and communities | {<br>"collection_id" uuid [not null]<br>"community_id" uuid [not null]<br>} | "collection_id"<br><br>"community_id" | collection_id collection.uuid<br><br>community_id community.uuid | |
| community2community | Manages the relationship between communities and subcommunites | {<br>"parent_comm_id" uuid [not null]<br>"child_comm_id" uuid [not null]<br>} | "parent_comm_id"<br><br>"child_comm_id" | parent_comm_id community.uuid<br><br>child_comm_id community.uuid | |
| cris_layout_box | Manages the configuration of boxes within entities' pages layout | {<br>”id” integer [not null]<br>”entity_id” integer [not null]<br>”type” character varying(255)<br>”collapsed” boolean [not null]<br>”shortname” character varying(255)<br>”header” character varying(255)<br>”minor” boolean [not null]<br>”security” integer<br>”style” character varying(255)<br>”clear” boolean<br>} | "id" | entity_id entity_type.id | |
| cris_layout_box2securitymetadata | Define which metadata control the security of a | {<br>"box_id" integer [not null] | n/a | | |

| | | | | | |
|---|---|---|---|---|---|
| | box when the security level is set to 4 (Custom policy) | "metadata_field_id" integer [not null]<br>} | | | |
| cris_layout_field | Define a single field to be included in a specific box (via the cris_layout_box2fields association table). A field can be of two `type`: “metadata” or “bitstream” | {<br>"field_id" integer [not null]<br>"metadata_field_id" integer<br>"bundle" "character varying(255)"<br>"rendering" "character varying(255)"<br>"row" integer [not null]<br>"priority" integer [not null]<br>"type" "character varying(255)"<br>"label" "character varying(255)"<br>"style" "character varying(255)"<br>"box_id" integer [not null]<br>"metadata_value" "character varying (255)"<br>"style_label" "character varying(255)"<br>"style_value" "character varying(255)"<br>} | "field_id" | | |
| cris_layout_field2nested | Define a relation between a field and some metadata to be displayed as nested | {<br>"nested_field_id" integer [not null]<br>"rendering" "character varying(255)"<br>"priority" integer [not null]<br>"label" "character varying(255)"<br>"style" "character varying(255)"<br>"style_label" "character varying(255)"<br>"style_value" "character varying(255)"<br>"metadata_field_id" integer [not null]<br>"field_id" integer [not null]<br>} | "nested_field_id" | field_id<br>cris_layout_field.<br>field_id<br><br>medatada_field_id<br>metadatafieldregistry.metadata_field_id | |
| cris_layout_metric2box | | {<br>"metric_type" "character varying(255)" [not null]<br>"cris_layout_box_id" int4 [not null]<br>"position" int4 [not null]<br>"id" int4 [not null]<br>} | “id” | cris_layout_box_id<br>cris_layout_box.id | |
| cris_layout_tab | Manages the tabs related to entities pages | {<br>"id" integer [not null]<br>"entity_id" integer [not null]<br>"priority" integer [not null]<br>"shortname" "character varying(255)"<br>"header" "character varying(255)"<br>"security" integer<br>} | "id" | | |
| cris_layout_tab2box | Manages the relationships between tabs and boxes | {<br>"cris_layout_tab_id" integer [not null]<br>"cris_layout_box_id" integer [not null]<br>”position” integer<br>} | "cris_layout_tab_id"<br>"cris_layout_box_id" | | |
| cris_layout_tab2securitymetadata | Define which metadata control the security of a tab when the security level is set to 4 (Custom policy) | {<br>"tab_id" integer [not null]<br>"metadata_field_id" integer [not null]<br>} | “tab_id" integer”<br>"metadata_field_id" | tab_id<br>cris_layout_tab.<br>cris_layout_tab_id<br><br>metadata_field_id<br>metadatafieldregistry.metadata_field_id | |
| cris_metrics | Collects metrics related to items having them | {<br>"id" integer [not null]<br>"metrictype" "character varying(255)"<br>"metriccount" float8 [not null]<br>"acquisitiondate" timestamp<br>"startdate" timestamp<br>"enddate" timestamp<br>"resource_id" uuid [not null]<br>"last" bool<br>"remark" text<br>"deltaperiod1" float8<br>"deltaperiod2" float8<br>"rank" float8<br>} | “id” | resource_id item.<br>uuid | |
| cwf_claimtask | Manages the workflow tasks that have been claimed by an user | {<br>"claimtask_id" integer [not null, default: `nextval('cwf_claimtask_seq')`] | "claimtask_id" | workflowitem_id<br>cwf_workflow.<br>workflowitem_id | |

| | | | | | |
|---|---|---|---|---|---|
| | | "workflowitem_id" integer<br>"workflow_id" text<br>"step_id" text<br>"action_id" text<br>"owner_id" uuid<br>} | | owner_id eperson.uuid | |
| cwf_collectionrole | Manages which groups perform a specific workflow role for a given collection | {<br>"collectionrole_id" integer [not null, default: `nextval('cwf_collectionrole_seq')`]<br>"role_id" text<br>"collection_id" uuid<br>"group_id" uuid<br>} | "collectionrole_id" | collection_id collection.uuid<br><br>group_id epersongroup.uuid | |
| cwf_inprogressuser | Manages the relationship between the item in workflow and the users who are performing a workflow task | {<br>"in_progress_user_id" integer [not null, default: `nextval('cwf_in_progress_user_seq')`]<br>"workflowitem_id" integer<br>"finished" boolean<br>"user_id" uuid<br>} | "in_progress_user_id" | workflowitem_id cwf_workflow.workflowitem_id | |
| cwf_pooltask | Manages the tasks within the workflow that sit in the pool queue | {<br>"pooltask_id" integer [not null, default: `nextval('cwf_pooltask_seq')`]<br>"workflowitem_id" integer<br>"workflow_id" text<br>"step_id" text<br>"action_id" text<br>"group_id" uuid<br>"eperson_id" uuid<br>} | "pooltask_id" | workflowitem_id cwf_workflow.workflowitem_id<br><br>group_id epersongroup.uuid<br><br>eperson_id eperson.uuid | |
| cwf_workflowitem | Manages the items that are going in a workflow | {<br>"workflowitem_id" integer [not null, default: `nextval('cwf_workflowitem_seq')`]<br>"multiple_titles" boolean<br>"published_before" boolean<br>"multiple_files" boolean<br>"item_id" uuid<br>"collection_id" uuid<br>} | "workflowitem_id" | item_id item.uuid<br><br>collection_id collection.uuid | cwf_claimtask<br><br>cwf_inprogressuser<br><br>cwf_pooltask<br><br>cwf_workflowitemrole |
| cwf_workflowitemrole | Manages the custom workflow roles, if any, assigned for a specific item | {<br>"workflowitemrole_id" integer [not null, default: `nextval('cwf_workflowitemrole_seq')`]<br>"role_id" text<br>"workflowitem_id" integer<br>"group_id" uuid<br>"eperson_id" uuid<br>} | "workflowitemrole_id" | workflowitem_id cwf_workflow.workflowitem_id<br><br>group_id epersongroup.uuid<br><br>eperson_id eperson.uuid | |
| deduplication | Manages the decision recorded about the potential duplicate detected by the system | {<br>"deduplication_id" integer [not null]<br>"fake" boolean<br>"tofix" boolean<br>"note" "character varying(256)"<br>"admin_time" timestamp<br>"reader_time" timestamp<br>"reader_note" "character varying(256)"<br>"reject_time" timestamp<br>"submitter_decision" "character varying (256)"<br>"workflow_decision" "character varying (256)"<br>"admin_decision" "character varying (256)"<br>"eperson_id" uuid<br>"admin_id" uuid<br>"reader_id" uuid<br>"first_item_id" uuid<br>"second_item_id" uuid<br>} | "deduplication_id" | eperson_id eperson.uuid<br><br>admin_id eperson.uuid<br><br>reader_id eperson.uuid<br><br>first_item_id item.uuid<br><br>second_item_id item.uuid | |
| doi | Manages the DOI Mint by the platform | {<br>"doi_id" integer [not null] | "doi_id" | dspace_object dspaceobject.uuid | |

| | | | | | |
|---|---|---|---|---|---|
| | | "doi" "character varying(256)"<br>"resource_type_id" integer<br>"resource_id" integer<br>"status" integer<br>"dspace_object" uuid<br>} | | | |
| dspaceobject | Manages the UUID of all the objects in DSpace (bitstrems, bundles, items, collections, communities, persons, groups) | {<br>"uuid" uuid [not null]<br>} | "uuid" | | community<br>collection<br>item<br>bundle<br>bitstream<br>eperson<br>epersongroup<br>resourcepolicy<br>doi<br>handle |
| entitytype | Includes all the entities defined within DSpace-CRIS | {<br>"id" integer [not null]<br>"label" "character varying(32)" [not null]<br>} | "id" | | cris_layout_tab<br>cris_layout_box |
| eperson | Includes all the accounts defined within DSpace-CRIS | {<br>"uuid" uuid [default: `gen_random_uuid()`]<br>"eperson_id" integer<br>"email" "character varying(64)"<br>"password" "character varying(128)"<br>"can_log_in" boolean<br>"require_certificate" boolean<br>"self_registered" boolean<br>"last_active" timestamp<br>"sub_frequency" integer<br>"netid" "character varying(64)"<br>"salt" "character varying(32)"<br>"digest_algorithm" "character varying(16)"<br>"session_salt" "character varying(32)"<br>} | "uuid" | uuid dspaceobject.uuid | epersongroup2eperson<br>item<br>subscription<br>cwf_workflowitemrole<br>resourcepolicy<br>cwf_claimtask<br>cwf_pooltask<br>cwf_in_progress_user<br>imp_record<br>imp_workflow_nstate |
| epersongroup | Includes all the groups defined within DSpace | {<br>"eperson_group_id" integer<br>"uuid" uuid [default: `gen_random_uuid()`]<br>"permanent" boolean [default: false]<br>"name" "character varying(250)"<br>} | "eperson_group_id" | uuid dspaceobject.uuid | epersongroup2eperson<br>group2group<br>group2groupcache<br>collection<br>cwf_collectionrole<br>cwf_workflowitemrole<br>resourcepolicy<br>cwf_claimtask<br>cwf_pooltask |
| epersongroup2eperson | Manages the relationship between groups and persons | {<br>"eperson_group_id" uuid [not null]<br>"eperson_id" uuid [not null]<br>} | "eperson_group_id"<br><br>"eperson_id" | eperson_group_id epersongroup.uuid<br><br>eperson_id eperson.uuid | |
| | | | | | |

| | | | | | |
|---|---|---|---|---|---|
| epersongroup2workspaceitem | Manages the groups that have given access to a specific workspace item (deprecated: supervisor feature) | {<br>"workspace_item_id" integer [not null]<br>"eperson_group_id" uuid [not null]<br>} | "workspace_item_id"<br><br>"eperson_group_id" | workspace_item_id workspaceitem. workspace_item_id<br><br>eperson_group_id epersongroup.uuid | |
| fileextension | Includes the file extensions to be associated with the bitstream formats | {<br>"file_extension_id" integer [not null]<br>"bitstream_format_id" integer<br>"extension" "character varying(16)"<br>} | file_extension_id | bitstream_format_id<br><br>bitstreamformatregistry. bitstream_format_id | |
| group2group | Manages the direct relationships between groups | {<br>"parent_id" uuid [not null]<br>"child_id" uuid [not null]<br>} | "parent_id"<br><br>"child_id" | parent_id group. uuid<br><br>child_id group.uuid | |
| group2groupcache | Cache table to improve the membership lookup performance. Includes all the group to group relation both direct (parent - children) than indirect (grandparent - grandchildren) | {<br>"parent_id" uuid [not null]<br>"child_id" uuid [not null]<br>} | "parent_id"<br><br>"child_id" | parent_id group. uuid<br><br>child_id group.uuid | |
| handle | Manages the handles mint by DSpace | {<br>"handle_id" integer [not null]<br>"handle" "character varying(256)"<br>"resource_type_id" integer<br>"resource_legacy_id" integer<br>"resource_id" uuid<br>} | "handle_id" | resource_id dspaceobject.uuid | |
| harvested collection | Holds the harvesting configuration associated to some collection and the corresponding harvesting status | {<br>"harvest_type" integer<br>"oai_source" "character varying"<br>"oai_set_id" "character varying"<br>"harvest_message" "character varying"<br>"metadata_config_id" "character varying"<br>"harvest_status" integer<br>"harvest_start_time" timestamp<br>"last_harvested" timestamp<br>"id" integer [not null]<br>"collection_id" uuid<br>} | “id” | collection_id collection.uuid | |
| harvested_item | Includes the source information for items harvested via OAI-PMH | {<br>"last_harvested" timestamp<br>"oai_id" "character varying"<br>"id" integer [not null]<br>"item_id" uuid<br>} | “id” | item_id item.uuid | |
| imp_bitstream | Table used by the DBMS Import feature which includes bitstreams to be imported in DSpace-CRIS | {<br>"imp_id" integer [not null]<br>"imp_bitstream_id" integer [not null]<br>"filepath" "character varying(512)" [not null]<br>"description" "character varying(512)"<br>"bundle" "character varying(512)"<br>"bitstream_order" integer<br>"primary_bitstream" boolean<br>"assetstore" integer<br>"name" "character varying(512)"<br>"imp_blob" bytea<br>"embargo_policy" integer<br>"embargo_group" uuid<br>"embargo_start_date" "character varying(100)"<br>"md5value" "character varying(32)"<br>} | "imp_bitstream_id" | imp_id imp_record. imp_id | |
| imp_bitstream_metadatavalue | Table used by the DBMS Import feature which includes the metadata related to the bitstreams to be imported in DSpace-CRIS | {<br>"imp_bitstream_metadatavalue_id" integer [not null]<br>"imp_bitstream_id" integer [not null]<br>"imp_schema" "character varying(128)" [not null]<br>"imp_element" "character varying(128)" | "imp_bitstream_metadatavalue_id" | imp_bitstream_id imp_bitstream. imp_bistream_id | |

| | | | | | |
|---|---|---|---|---|---|
| | | [not null]<br>"imp_qualifier" "character varying(128)"<br>"imp_value" text [not null]<br>"imp_authority" "character varying(256)"<br>"imp_confidence" integer<br>"metadata_order" integer [not null]<br>"text_lang" "character varying(32)"<br>} | | | |
| imp_metadatavalue | Table used by the DBMS Import feature which includes the metadata related to the items to be imported in DSpace-CRIS | {<br>"imp_metadatavalue_id" integer [not null]<br>"imp_id" integer [not null]<br>"imp_schema" "character varying(128)" [not null]<br>"imp_element" "character varying(128)" [not null]<br>"imp_qualifier" "character varying(128)"<br>"imp_value" text [not null]<br>"imp_authority" "character varying(256)"<br>"imp_confidence" integer<br>"metadata_order" integer [not null]<br>"text_lang" "character varying(32)"<br>} | "imp_metadatavalue_id" | imp_id imp_record. imp_id | |
| imp_record | Table used by the DBMS Import feature which includes the items to be imported in DSpace-CRIS using the DBMS import framework | {<br>"imp_sourceref" "character varying (256)"<br>"imp_record_id" "character varying (256)" [not null]<br>"imp_id" integer [not null]<br>"imp_eperson_uuid" uuid [not null]<br>"imp_collection_uuid" uuid [not null]<br>"status" "character varying(1)"<br>"operation" "character varying(64)"<br>"last_modified" timestamp<br>"handle" "character varying(64)"<br>} | “imp_id” | imp_eperson_uuid eperson.uuid<br><br>imp_collection_uuid collection.uuid | |
| imp_record_wstate | Table used by the DBMS Import feature which manages the association between the record and the operation to perform in hte workflow | {<br>"imp_id" integer [not null]<br>"imp_wnstate_op_id" integer [not null]<br>} | "imp_id"<br><br>"imp_wnstate_op_id" | imp_id item_record. imp_id<br><br>imp_wnstate_od_id imp_workflow_nstate. imp_wnstate_op_id | |
| imp_workflow_nstate | Table used by the DBMS Import feature containing the details about each action to trigger into the workflow of the resulting item | {<br>"imp_wnstate_op_id" integer [not null]<br>"imp_wnstate_desc" "character varying (64)"<br>"imp_wnstate_op" "character varying (64)" [not null]<br>"imp_wnstate_op_par" "character varying(64)"<br>"imp_wnstate_order" integer [not null]<br>"imp_wnstate_eperson_uuid" uuid<br>} | "imp_wnstate_op_id" | imp_wnstate_eperson_uuid eperson. uuid | |
| item | Includes all the DSpace items | {<br>"owning_collection" uuid<br>"item_id" integer<br>"in_archive" boolean<br>"withdrawn" boolean<br>"last_modified" timestamp<br>"discoverable" boolean<br>"uuid" uuid [default: `gen_random_uuid()`]<br>"submitter_id" uuid<br>} | "uuid" | owning_collection collection.uuid<br><br>submitter_id eperson.uuid<br><br>uuid dspaceobject. uuid | collection2item<br><br>collection<br><br>item2bundle<br><br>harvested_item<br><br>deduplication<br><br>relationship<br><br>requestitem<br><br>versionitem<br><br>workspaceitem<br><br>cwf_workflowitem |
| item2bundle | | {<br>"bundle_id" uuid [not null] | "bundle_id"<br><br>"item_id" | bundle_id bundle. uuid | |

| | | | | | |
|---|---|---|---|---|---|
| | Manages the relationship between items and bundles | "item_id" uuid [not null]<br>} | | item_id item.uuid | |
| metadatafieldregistry | Includes the metadata used within DSpace.CRIS | {<br>"metadata_schema_id" integer [not null]<br>"metadata_field_id" integer [not null, default: `nextval ('metadatafieldregistry_seq')`]<br>"element" "character varying(64)"<br>"qualifier" "character varying(64)"<br>"scope_note" text<br>} | "metadata_field_id" | metadata_schema_id metadataschemaregistry. metadata_shema_id | metadatavalue |
| metadataschemaregistry | Includes the metadata schema to which the metadata used within DSpace-CRIS are related | {<br>"metadata_schema_id" integer [not null, default: `nextval ('metadataschemaregistry_seq')`]<br>"namespace" "character varying(256)"<br>"short_id" "character varying(32)"<br>} | "metadata_schema_id" | | metadatafieldregistry |
| metadatavalue | Includes the values of the metadata related to all DSpace-CRIS objects | {<br>"metadata_value_id" integer [not null, default: `nextval ('metadatavalue_seq')`]<br>"metadata_field_id" integer<br>"text_value" text<br>"text_lang" "character varying(24)"<br>"place" integer<br>"authority" "character varying(100)"<br>"confidence" integer<br>"dspace_object_id" uuid<br>"security_level" integer<br>} | "metadata_value_id" | dspace_object_id dspaceobject.uuid<br><br>metadata_field_id metadatafieldregistry.,metadata_field_id | |
| most_recent_checksum | Process table used by the integrity checed to identify which bitstream needs to be checked | {<br>"to_be_processed" boolean [not null]<br>"expected_checksum" "character varying" [not null]<br>"current_checksum" "character varying" [not null]<br>"last_process_start_date" timestamp [not null]<br>"last_process_end_date" timestamp [not null]<br>"checksum_algorithm" "character varying" [not null]<br>"matched_prev_checksum" boolean [not null]<br>"result" "character varying"<br>"bitstream_id" uuid<br>} | none | bitstream_id bitstream.uuid | |
| nbevent_processed | Stores technical informations about processed notifications of broker events | {<br>"nbevent_id" "character varying(255)"<br>"nbevent_timestamp" timestamp<br>"eperson_uuid" uuid<br>"item_uuid" uuid<br>} | nbevent_id | eperson_uuid eperson.uuid<br><br>item_uuid item.uuid | |
| openurltracker | Tracks openurl failed transmissions | {<br>"tracker_id" integer<br>"tracker_url" "character varying(100)"<br>"uploaddate" date<br>} | tracker_id | | |
| orcid_history | Stores the attempts and results of sending an entity to ORCID. | {<br><br>"id" integer [not null]<br>"owner_id" uuid [not null]<br>"entity_id" uuid<br>"put_code" "character varying"<br>"timestamp_last_attempt" timestamp<br>response_message text<br>status integer<br>"metadata" text<br>"operation" "character varying(255)"<br>"record_type" "character varying(255)"<br>"description" "character varying(255)" | “id” | owner_id item<br>entity_id item | |

| | | | | | |
|---|---|---|---|---|---|
| | | } | | | |
| orcid_queue | Stores the entities to be sent to ORCID | {<br><br>"id" integer [not null]<br>"owner_id" uuid [not null]<br>"entity_id" uuid<br>"attempts" integer<br>"put_code" "character varying(255)"<br>"record_type" "character varying(255)"<br>"description" "character varying(255)"<br>"operation" "character varying(255)"<br>"metadata" text<br><br>} | “id” | owner_id item<br>entity_id item | |
| process | Includes information about processes started via REST in DSpace-CRIS | {<br>"process_id" integer [not null]<br>"user_id" uuid [not null]<br>"start_time" timestamp<br>"finished_time" timestamp<br>"creation_time" timestamp [not null]<br>"script" "character varying(256)" [not null]<br>"status" "character varying(32)"<br>"parameters" "character varying(512)"<br>} | "process_id" | | process2bitstream |
| process2bitstream | Manages the relationship between processes and the involved bitstreams | {<br>"bitstream_id" uuid [not null]<br>"process_id" integer [not null]<br>} | "bitstream_id"<br><br>"process_id" | bitstream_id<br>bitstream.uuid<br><br>process_id<br>processes.<br>process_id | |
| process2group | Relationship between a process and group(s) user was | {<br>"process_id" integer [not null]<br>"group_id" uuid [not null]<br>} | "process_id"<br>"group_id" | process_id process.<br>process_id<br><br>group_id<br>epersongroup.<br>group_id | |
| registrationdata | Manages the self-registration procedure | {<br>"registrationdata_id" integer [not null]<br>"email" "character varying(64)"<br>"token" "character varying(48)"<br>"expires" timestamp<br>} | "registrationdata_id" | | |
| registrationdata2group | Links registration data to eperson group | {<br>"registrationdata_id" integer [not null]<br>"group_id" uuid [not null]<br>} | "registrationdata_id"<br>"group_id" | registrationdata_id<br>registrationdata.<br>registrationdata_id<br><br>group_id<br>epersongroup.<br>eperson_group_id | |
| relationship | Manages functional relationships between DSpace-CRIS items | {<br>"id" integer [not null]<br>"left_id" uuid [not null]<br>"type_id" integer [not null]<br>"right_id" uuid [not null]<br>"left_place" integer<br>"right_place" integer<br>"leftward_value" "character varying"<br>"rightward_value" "character varying"<br>} | "id" | left_id item.uuid<br><br>right_id item.uuid<br><br>type_id<br>relationship_type.id | |
| relationship_type | Describes the functional relationships types between DSpace-CRIS items | {<br>"id" integer [not null]<br>"left_type" integer<br>"right_type" integer<br>"leftward_type" "character varying(32)" [not null]<br>"rightward_type" "character varying (32)" [not null]<br>"left_min_cardinality" integer<br>"left_max_cardinality" integer<br>"right_min_cardinality" integer<br>"right_max_cardinality" integer<br>"copy_to_left" boolean [not null, default: false] | "id" | | relationship |

| | | | | | |
|---|---|---|---|---|---|
| | | "copy_to_right" boolean [not null, default: false]<br>"tilted" integer<br>} | | | |
| requestitem | Manages requests for accessing non public items and bitstreams | {<br>"requestitem_id" integer [not null]<br>"token" "character varying(48)"<br>"allfiles" boolean<br>"request_email" "character varying(64)"<br>"request_name" "character varying(64)"<br>"request_date" timestamp<br>"accept_request" boolean<br>"decision_date" timestamp<br>"expires" timestamp<br>"request_message" text<br>"item_id" uuid<br>"bitstream_id" uuid<br>} | "requestitem_id" | item_id item.uuid<br><br>bitstream_id<br>bitstream.uuid | |
| resourcepolicy | Manages the access policies related to all DSpace objects | {<br>"policy_id" integer [not null]<br>"resource_type_id" integer<br>"resource_id" integer<br>"action_id" integer<br>"start_date" date<br>"end_date" date<br>"rpname" "character varying(30)"<br>"rptype" "character varying(30)"<br>"rpdescription" text<br>"eperson_id" uuid<br>"epersongroup_id" uuid<br>"dspace_object" uuid<br>} | "policy_id" | dspace_object<br>dspaceobject.uuid<br><br>eperson_id<br>eperson.uuid<br><br>epersongroup_id<br>epersongroup.uuid | |
| schema_version | Contains information about the current version of the system and any performed upgrade | {<br>"installed_rank" integer [not null]<br>"version" "character varying(50)"<br>"description" "character varying(200)" [not null]<br>"type" "character varying(20)" [not null]<br>"script" "character varying(1000)" [not null]<br>"checksum" integer<br>"installed_by" "character varying(100)" [not null]<br>"installed_on" timestamp [not null, default: now( )]<br>"execution_time" integer [not null]<br>"success" boolean [not null]<br>} | "installed_rank" | | |
| site | Single row table for future use. Holds the uuid assigned to the whole repository | {<br>"uuid" uuid [not null]<br>} | "uuid" | uuid dspaceobject.<br>uuid | |
| subscription | Manages the user email subscription to new content | {<br>"subscription_id" integer [not null]<br>"eperson_id" uuid<br>"collection_id" uuid<br>} | "subscription_id" | eperson_id<br>eperson.uuid<br><br>collection_id<br>collection.uuid | |
| versionhistory | Used by the versioning system to generate an unique ID common to all the versions of a specific item | {<br>"versionhistory_id" integer [not null]<br>} | "versionhistory_id" | | versionitem |
| versionitem | Manages the different versions of the items | {<br>"versionitem_id" integer [not null]<br>"version_number" integer<br>"version_date" timestamp<br>"version_summary" "character varying (255)"<br>"versionhistory_id" integer<br>"eperson_id" uuid<br>"item_id" uuid<br>} | "versionitem_id" | versionhistory_id<br>versionhistory.<br>versionhistory_id<br><br>item_id item.uuid | |
| webapp | | | "webapp_id" | | |

| | | | | | |
|---|---|---|---|---|---|
| | Holds the information about the webapplication connected to the database | {<br>"webapp_id" integer [not null]<br>"appname" "character varying(32)"<br>"url" "character varying"<br>"started" timestamp<br>"isui" integer<br>} | | | |
| workspaceitem | Manages items in persons' workspace | {<br>"workspace_item_id" integer [not null]<br>"multiple_titles" boolean<br>"published_before" boolean<br>"multiple_files" boolean<br>"stage_reached" integer<br>"page_reached" integer<br>"item_id" uuid<br>"collection_id" uuid<br>} | "workspace_item_id" | item_id item.uuid<br><br>collection_id<br>collection.uuid | |