Publication Claim

Scenario: A new researcher joins the institution and logs in for the first time in the repository. The publication claim services found most of their publications in the OpenAIRE network and prompts for import. The researcher reviews the list, confirms the authorship and imports the publication saving a significant amount of (often publicly paid) time. Moreover, the authorship confirmation will come back later to OpenAIRE offering useful information about the data quality and potential enrichment. The same applies for publications authored by researchers in different institutes, having the data in multiple repositories makes the data more reliable and raises the chance to get more information and content from any of the authors.

The goal of the Publication Claim service is to support the scenario above.

The service has been designed to be independent from a specific provider or implementation so that it can be easily extended and maintained over time. Moreover, multiple providers can be active at the same time improving the chance to save researchers time.

This feature currently only supports the OpenAIRE Research Graph via the Publication REST API. Other good (future) candidates to be integrated via such framework are ORCID or commercial databases via their authors’ IDs.

Data source

The OpenAIRE Publication REST API are used to retrieve publication that could be authored by researcher at the Institution. The OpenAIRE Publication REST API are queried using the names known by the repository for its researchers, the retrieve list is later reduced passing identified publications to a pipeline of JAVA classes that can promote or reject his inclusion in the suggestion list. Publications previously discarded by the researcher are automatically filter out avoiding to re-present the same publication again and again.

The suggestion providers are defined in the dspace/config/spring/api/suggestions.xml spring configuration file. Indeed, the system can be extended to more provider than the one implemented to query the OpenAIRE Researcher Graph.

```xml
<util:map id="suggestionProviders" map-class="java.util.HashMap" key-type="java.lang.String" value-type="org.dspace.app.suggestion.SuggestionProvider">
  <entry key="oaire" value-ref="OAIREPublicationLoader"/>
</util:map>

<bean id="OAIREPublicationLoader" class="org.dspace.app.suggestion.oaire.OAIREPublicationLoader">
  <property name="sourceName" value-ref="oaire"/>
  <property name="primaryProvider" ref="openaireLiveImportDataProviderByAuthor"/>
  <property name="otherProviders">
    <list>
      <ref bean="openaireLiveImportDataProviderByTitle"/>
    </list>
  </property>
  <property name="names">
    <list>
      <value>dc.title</value>
      <value>crisrp.name</value>
      <value>crisrp.name.translated</value>
      <value>crisrp.name.variant</value>
    </list>
  </property>
  <property name="pipeline">
    <list>
      <bean class="org.dspace.app.suggestion.oaire.AuthorNamesScorer">
        <property name="contributorMetadata">
          <list>
            <value>dc.contributor.author</value>
          </list>
        </property>
        <property name="names">
          <list>
            <value>dc.title</value>
            <value>crisrp.name</value>
            <value>crisrp.name.translated</value>
            <value>crisrp.name.variant</value>
          </list>
        </property>
        <property name="birthDateMetadata">
          <value>person.birthDate</value>
        </property>
        <property name="publicationDateMetadata">
          <value="dc.date.issued"/>
        </property>
      </bean>
    </list>
  </property>
</bean>
```
Each suggestionProvider is identified by an unique name used as key in the suggestionProviders map.

The OpenAIRE implementation is represented by the java class org.dspace.app.suggestion.oaire.OAIREPublicationLoader and configured via the following properties:

- the primaryProvider property defines which DSpace ExternalDataProvider use to retrieve the record
- the otherProviders property defines which DSpace ExternalDataProviders other than the primary could offer the same records. This is used to automatically remove from the suggestion list records that are imported manually by the researcher from these other providers
- the name property defines the metadata to use to build the search query over the openAIRE Research Graph to retrieve the list of publications to evaluate as suggestions. It is responsibility of the scorers defined in the pipeline to compute a score for each retrieved publication and eventually discard the ones that are not good enough.
- the pipelines property allows a future refinement of the procedure introducing for instance support for researcher preference that could exclude specific sources (pubmed, crossref, datacite, etc.) or keywords/subjects unrelated with his research interests. Right now two scorers are in place:
  - AuthorNamesScorer to validate the finding against the researcher name as it has been found that searching the openAIRE Publication API for author such as Bollini Susanna would find also publications co-authored by Andrea Bollini and Susanna Mornati;
  - DateScorer to validate the finding against a guessed range of years that the system expect to be the productivity or interested windows for the researcher. This range is calculated using the graduation date if available or the birthday but can be also set manually by the researcher in his profile.

The dspace script class org.dspace.app.suggestion.OAIREPublicationLoaderRunnableCli is used to run the queries and store the identified publication in the dedicated SOLR core suggestion for further processing.

The dspace script can be run both from the CLI than from the UI.

To run the loader from the dspace installation bin folder

```bash
dspace import-oaire-suggestions [-s uuid-of-single-researcher]
```

without the `-s` parameter the script will process all the researcher available in the system.

The script can be also run from the Script UI so that it is also available to repository manager that cannot be access the CLI

blocked URL

Two external source providers, openAIRE Publications By Title and By Author have been defined according to the standard DSpace External Sources framework. It is activated in the config/spring/api/external-services.xml as follow

```xml
<bean id="openaireLiveImportDataProviderByAuthor" class="org.dspace.external.provider.impl.LiveImportDataProvider">
  <property name="metadataSource" ref="openaireImportServiceByAuthor"/>
  <property name="sourceIdentifier" value="openaire"/>
  <property name="recordIdMetadata" value="dc.identifier.other"/>
  <property name="supportedEntityTypes">
    <list/>
    <value>Publication</value>
  </property>
</bean>

<bean id="openaireLiveImportDataProviderByTitle" class="org.dspace.external.provider.impl.LiveImportDataProvider">
  <property name="metadataSource" ref="openaireImportServiceByTitle"/>
  <property name="sourceIdentifier" value="openaireTitle"/>
  <property name="recordIdMetadata" value="dc.identifier.other"/>
  <property name="supportedEntityTypes">
    <list/>
    <value>Publication</value>
  </property>
</bean>
```

with the importer services defined via the Live Import Framework in /dspace-api/src/main/resources/spring/spring-dspace-addon-import-services.xml as follow

```xml
<bean id="openaireImportServiceByAuthor" class="org.dspace.importer.external.openaire.service.OpenAireImportMetadataSourceServiceImpl" scope="singleton">
  <property name="metadataFieldMapping" ref="openaireMetadataFieldMapping"/>
</bean>

<bean id="openaireImportServiceByTitle" class="org.dspace.importer.external.openaire.service.OpenAireImportMetadataSourceServiceImpl" scope="singleton">
  <property name="metadataFieldMapping" ref="openaireMetadataFieldMapping"/>
</bean>

<bean id="openaireMetadataFieldMapping" class="org.dspace.importer.external.openaire.service.metadatamapping.OpenAireFieldMapping">
</bean>
```
The mapping between the openAIRE Publications metadata and the dspace metadata is provided in the config/spring/api/openaire-integration.xml using the usual xpath approach of the DSpace Live Import Framework.

Having used the Live Import Framework internally to the loader to perform the query has had the side benefit to make available the publication data of the openAIRE Research Graph also to the direct import functionality of DSpace, so that the researcher can now query the openAIRE graph and import publication on demand.

The SOLR suggestion core has the following structure

```xml
<fields>
  <field name="source" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="suggestion_fullid" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="suggestion_id" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="target_id" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="title" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="display" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="abstract" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="category" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="external-url" type="string" indexed="true" stored="true" omitNorms="true" />
  <field name="processed" type="boolean" indexed="true" stored="true" omitNorms="true" />
  <field name="trust" type="double" indexed="true" stored="true" omitNorms="true" />
  <field name="evidences" type="string" indexed="false" stored="true" omitNorms="true" />
</fields>
```

The source field would allow the reuse of such structure by other sources than openAIRE.

Three endpoints have been designed to expose the result of the processing to the DSpace UI and so to the Repository Managers and single researchers:

- `/api/integration/suggestionsources` to provide access to summary information about the available suggestion from each source (openaire, orcid, etc.)
- `/api/integration/suggestiontargets` to provide access to summary information about the available suggestions for a specific researcher
- `/api/integration/suggestions` to provide access to the detailed suggestions so that they can be reviewed and managed by the repository manager or the researcher to whom they related

The detailed REST contract for such endpoints are available on the 4Science Rest7Contract repository and embedded at the bottom of the page for easy reference.

**Repository Manager UI**

The resulting UI is accessible for the Repository Manager from the administrative menu. As entry point for the features a “Notifications” menu entry has been added to the DSpace administrative menu, from where the repository manager will be able to manage the suggestions got from the different sources.

A list of local profiles with candidate publications will be shown so that the repository manager can review them directly or support the researcher: blocked URL

For each candidate the available suggestions are shown, sorted by the evaluated total score (summing up all the processed evidences). Using the bottom see evidence is possible to get detailed information about the score

The suggested authorship of each article can be confirmed importing the data locally, or rejected. This operation can be performed individually but also simultaneously for all the selected suggestions, speeding up the process. The decision can also be guided by inspecting the matching evidences which are displayed for each suggestion by clicking on ‘See evidence’

The suggestions list can be sorted by total score descending or ascending (highlighting the weakest candidates).

**Researcher UI**

This functionality requires to implement a mechanism to uniquely link user accounts with Person profiles. Such mechanism is implemented out-of-box in DSpace-CRIS. Where the link is not implemented, the Repository Manager UI can still be used.

The single researcher is also allowed to directly review his suggestions. Upon login he is informed about the availability of suggestions from one or more providers

blocked URL
and can proceed to review the suggestions list in the same way than the Repository Manager, the notification message is also always available at the top of the mydspace.

Processing the decisions

The backend is responsible to process the repository manager or researcher decisions taken over the received suggestions. The publication to be imported are processed according to the Import from External Sources normal data flow of DSpace 7. Upon import the suggestion document is removed from the SOLR core, in case of rejection the document is updated flagging it as rejected so that it will be not longer proposed to the user.

Rest Contract

Three endpoints have been designed to interact with the publication claim service. They are documented in our REST Contract.

- /api/integration/suggestionsources to provide access to summary information about the available suggestion from each source (openaire, orcid, etc.)
- /api/integration/suggestiontargets to provide access to summary information about the available suggestions for a specific researcher
- /api/integration/suggestions to provide access to the detailed suggestions so that they can be reviewed and managed by the repository manager or the researcher to whom they related