The OpenAIRE Action Manager Framework

Michele Artini, Claudio Atzori, Sandro La Bruzzo

Consiglio Nazionale delle Ricerche, Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", Via Moruzzi 1, 56124 Pisa, Italy

name.surname@isti.cnr.it

Abstract - The OpenAire infrastructure offers services for collecting records (publications, datasets, persons, organizations, data sources, projects) from external data sources with the purpose of identifying relationships between them. The collected objects and their relationships are stored in HBASE according to the OpenAIRE data model. The Action Manager framework has been designed to offer an OpenAIRE data model oriented API for the enrichment and the fixing of the OpenAIRE HBASE information space.

Introduction

The OpenAire infrastructure offers services for collecting publications and datasets from external data sources (e.g. publication and dataset repositories, CRIS systems, entity registries, and the OpenAIRE Zenodo Repository) with the purpose of identifying relationships between them. The infrastructure also collects information from so-called "entity registries", which bear authoritative lists about research funding or activities, such as projects (e.g. EC-CORDA, WellcomeTrust), data sources (e.g. OpenDOAR), authors (e.g. ORCID), with the purpose of "contextualizing" publication and datasets with curated information. As a result of collecting external data sources, the OpenAIRE information space can be conceived as a graph of interconnected objects. Data in such graph can be further curated (e.g. enriched, updated) via three main services:

- End-user Claim Services: "claims" are statement from authorized users, who can, through the portal, select publication metadata (by providing the relative DOI or browsing the DRIVER infrastructure information space) and specify the relative EC projects. The resulting information is to be preserved into the system, together with an association to the end-users for future updates, as it represents a form of "native" content for the information space.

- End-user feedback Services: registered end-users browsing the information space can specify corrections to existing objects, such as adding/removing relationships, updating a property, etc. Such “actions” are preserved and, before being applied, need to be validated by OpenAIRE data curators. As we shall see, several kinds of actions may be conceived: to be validated by data curators, to be validated by specific data curators, to be immediately applied.
- Information Inference Services: such services provide inference/mining algorithms to analyze the graph of objects in the information space or the documents (e.g. PDFs, XMLs) associated to them, in order to identify relevant relationships between such objects, new objects, or object property values.

The infrastructure needs to provide tools to (i) store and preserve the three kinds of information collected above and (ii) to insert such information in the information space at the proper data flow phase.

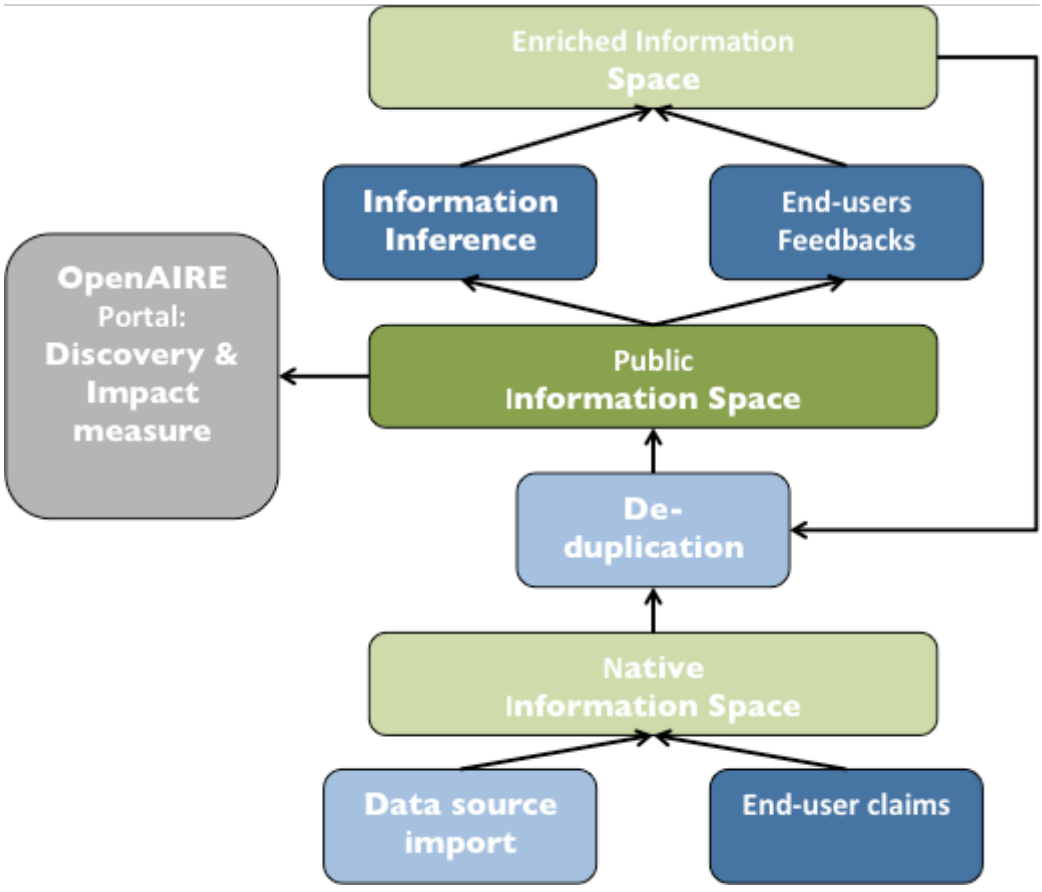

Figura 1 - OpenAIRE Data Flow

As shown in Figure 1, before any process or data curation takes place, the information space might be entirely rebuilt by simply re-collecting all content from external data sources. During the "collection" phase, the information space should also be enriched with information claimed by end-users. Indeed, such information should be considered as "native", i.e. cannot be collected from anywhere else. The combination of external sources and "claimed data" forms the so-called native information space. Native objects may hide duplicates, which are found and resolved (operating "record merges") by De-Duplication Services during the "de-duplication" phase. The resulting information space can be directly made accessible to OpenAIRE portal users (i.e. indexed), as it represents the "native version" of the public information space, containing publications, datasets, projects, persons and relationships between them from European data sources and OpenAIRE user claims.

However, the OpenAIRE infrastructure allows for such graph to be further curated by the intervention of end-users providing "feedbacks" and by inference/mining services. Figure 1 illustrates that the data flow, after such curation actions occur, generates an enriched information space, where duplicated objects may again be present. All object updates, both from end-user feedbacks and inference services, are encoded by:

1. Introducing duplicates of the objects to be updated;
2. Assigning a higher level of TRUST to the duplicates;
3. Placing the changes in the duplicate objects.

Accordingly, a further de-duplication run delivers the new "enriched version" of the public information space, to be again indexed and delivered to OpenAIRE portal users.

The data flow is therefore divided into four main interdependent phases: data collection, first de-duplication, data enrichment, and second disambiguation. Through all such phases, objects and relationships are enriched with provenance information that allows distinguishing in which phase the object was inserted and by which agent (human or service). Such tracking allows the phases to be rolled-back and repeated without losing the results obtained in the previous phases. This is not the case when moving bottom-up. For example, if a data source needs to be recollected, then all phases after data collection need to be rolled-back and repeated. This is because de-duplication phases, as well as enrichment phases may had to do with the objects delivered by such data source.

This document was intended to describe the End-user Feedback Service, to be used by data curators and registered end-users to submit and validate changes to the information space. However, the design of the data flow above has proven the need of a common management layer for the information space, capable of managing cycles of update, enrich and roll-backs operations due to all services willing to change the information space. Once the End-user feedback service will be introduced, the document will provide the specification of a D-NET service sub-framework called Action Management, which provides general-purpose tools for managing all data flows phases described above in a systematic, autonomous, and controlled way. The idea is that all inputs from end-users claims, end-user feedbacks, and inference services will be encoded as "sets of actions" to applied to the information space. Action sets can be preserved and "promoted", i.e. applied over a given information space, following a temporal scheme to be implemented by an Action Manager Service.

End-users Feedbacks

End-users who are regularly registered to the infrastructure will be provided with services for searching and browsing the public information space and for submitting "advices" on how to improve the information space. Such advices, called actions, may regard:

- Updates of properties to individual objects in the information space (as described by D6.1)
- Addition of objects;
- Addition and removal of relationships between such objects
- Merging two objects considered to be descriptions of the same real-world object
- Splitting of one object into two or more objects (opposite of merging).

As shown in Figure 2, actions are submitted by users in a "pending" status, i.e. not yet visible as part of the public information space. End-user feedback services will also support data curators with tools to be notified of new pending actions and to "validate" such actions, i.e. apply them to the information space. Data curators will also be able to "rollback" such actions that is to remove their consequences on the information space. Given their "expert" role, via the service data curators can directly submit "validated actions".

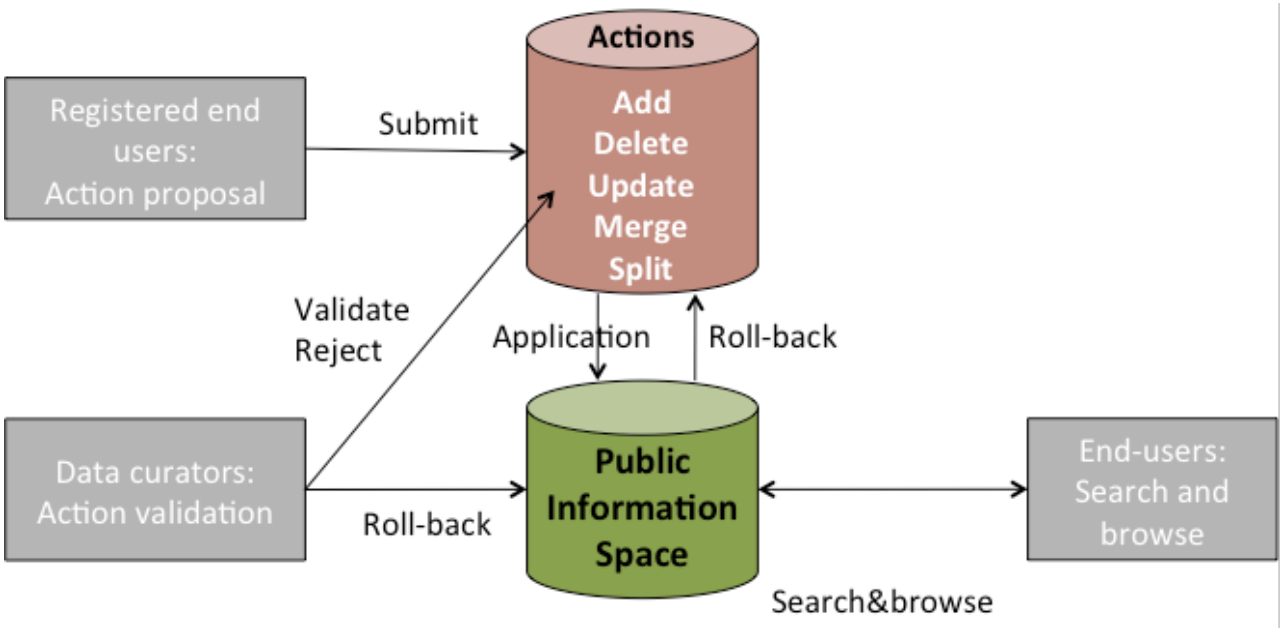

Figura 2 - End-users Feedbacks: workflows

The web interface of the End-User Feedback Services will be developed by NKUA and benefit from the back-end provided by CNR. The architecture of such back-end is described in the next section.

NOTE: Actions are applied over native objects only. This means that if an action is applied over a "representative object" (i.e. an object resulting from the merge of a set of objects), the action is to be applied to ALL the records in the set (except from "split actions"). For such a reason, the user interfaces should instruct the end-user trying to update the representative object that this would be the case.

Action Management Framework

Objects in the OpenAIRE data model (e.g. publications, datasets, persons, organizations, data sources, projects) are represented in HBASE as rows with columns storing the relationships with other objects. The nature and internal structure of such columns (which may change over time based on high level requirements) should not be known to the services surrounding the information space and willing to update or enrich its content.

The Action Manager framework has been designed to offer an OpenAIRE data model oriented API to the OpenAIRE HBASE information space (see Figure 3). It consists of two main services: **Action Store Service** and **Action Interpreter Service**.

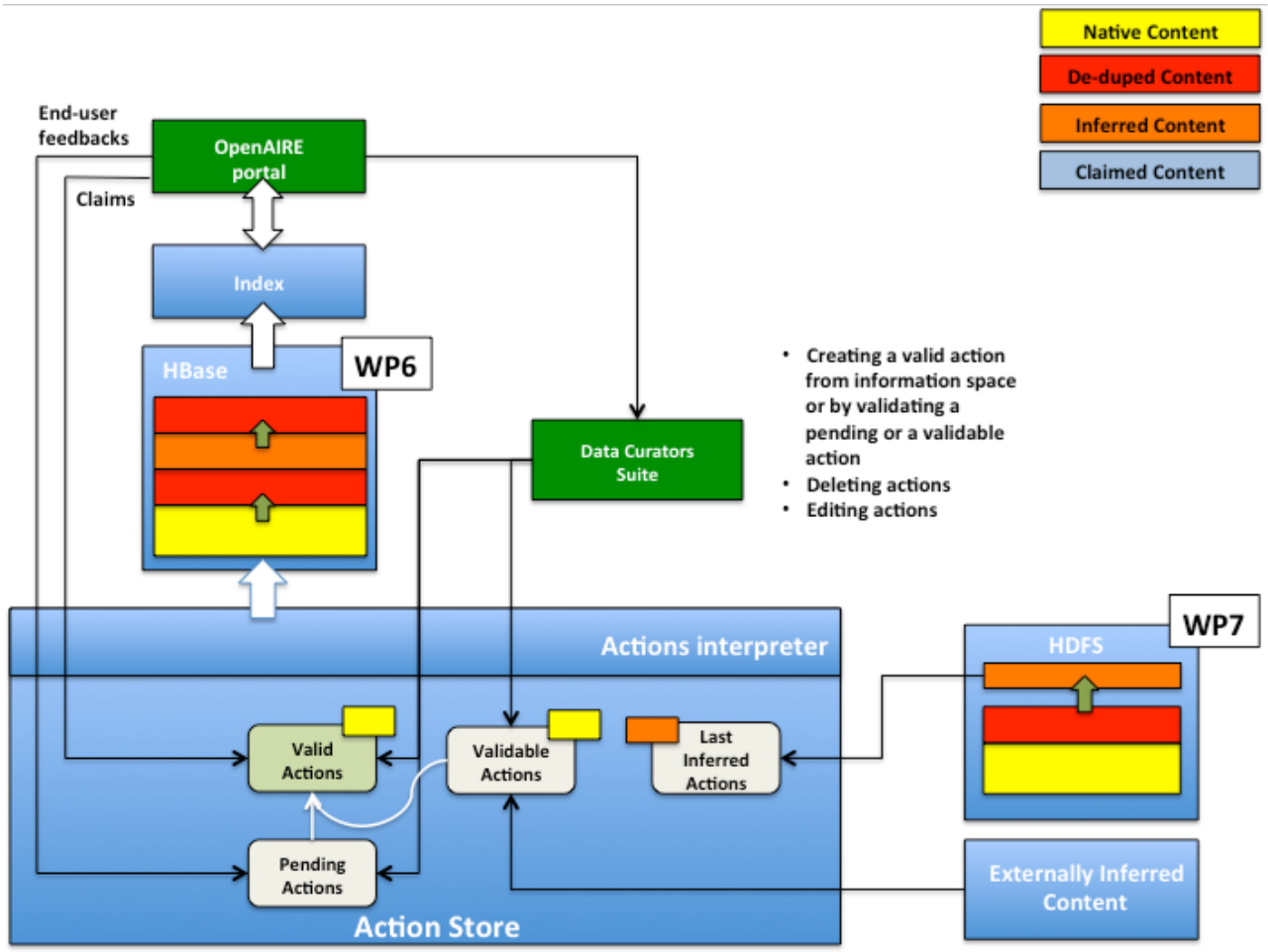

Figura 3 - Action Management Service: the high-level architecture

The **Action Interpreter Service** is designed to manage sets of complex actions. Complex actions handle inserts or updates of OpenAIRE data model objects. In turn, a complex action is encoded as a set of atomic actions relative to the HBASE encoding of the aforementioned objects. As such, they represent operations such as: adding a row or a relationship between two rows in HBASE according to the current physical representation of objects.

Actions can be of three different kinds, depending on the degree of validation they require:

- Validation not necessary, i.e. directly published: this is the case of data curators that submit actions or of registered end-users modifying properties of the publications they have claimed;
- Validation by a given user
    - Pessimistic: the user has to validate for the action to be applied

    - Optimistic: the action is applied (hence visible) and the user can approve the choice (validation) or reject it. In OpenAIRE, this is the case of relationships inferred from the PDF between a publication and a project, where such relationships must be validated by the project coordinator before being published
- Validation by a class of users (role)
    - Pessimistic: one user with the given role has to validate the action to make it visible. In OpenAIRE this is the case for actions relative to end-user feedbacks, which need data curators to validate the actions before publishing their effects.
    - Optimistic: the actions are applied and one user with the given role may reject the actions in a second stage. In OpenAIRE this is the case for all actions fired by modules in the Information Inference Services.

With respect to Figure 3:

- Valid-able actions are actions with validation by a given user or role under optimistic assumptions;
- Pending actions are actions with validation by a given user or role under pessimistic assumptions;
- Valid actions are actions with validation not necessary.

Below is the list of properties describing one action:

- actionID: ID
- timeOfCreation: timestamp
- operationType:
    - insert relationships, insert object, update object, insert information package (e.g. DC record + reference to project)
    - future developments: split, delete object, delete relationship, other syntactic sugar options (e.g. insert cluster and elements of cluster)
- objectID:
    - the ID of the object to be updated if the operation is “update object”
    - the **stateless** ID of the object to be inserted if the operation is “insert object” (following the rules for OPENAIRE ID generation)
- parameters: property-value pairs
- bidirectional (if inserting a relationship): true/false

- actionProvenance: describing the process leading to the creation of the action. The possible vocabulary terms are:
    - sysimport:crosswalk:repository (if one day OpenAIRE will decide to use sets of actions to store the records)
    - sysimport:crosswalk:aggregator
    - sysimport:crosswalk:entityregistry
    - sysimport:crosswalk:datasetarchive
    - sysimport:crosswalk:cris
    - sysimport:mining:repository (e.g.publication-project relationship mining)
    - sysimport:mining:aggregator
    - sysimport:mining:entityregistry
    - sysimport:mining:datasetarchive
    - sysimport:mining:cris
    - userclaim:crossref (e.g. user interface, publication claim via DOI), o userclaim:driver
    - userclaim:orcid
- agent: a string, encoding the agent (human, service, algorithm) generating the action
- agentID: ID of the agent (e.g., end-user identifier)
- validationKind:
    - notNecessary
    - user (e.g. project coordinator)
    - class of users (e.g. data curators)
- validationStatus: TRUE/FALSE
- validationTime: pessimistic or optimistic
- validationClassOfUsers: if validationKind is “class of users”
- validationUserID: if validationKind is “user”
- trust: 0<= K <= 1 Or infinite Or Neutral (see D6.1)

Sets are bags of actions, used to organize actions into blocks of execution. To this aim a set is uniquely identified by a name and has the following properties:

- Applied: TRUE/FALSE; if TRUE the actions are currently active on the information space;

- Last execution date: date (this is the date propagated to all actions, once executed, in the context of the set)
- Phase: describes in which phase of the data flow the set has to be applied,
    - datacollection, e.g. claimed publications (CrossRef, ORCID), publication-relationship inference from PDFs;
    - after first data de-duplication, e.g. inference, end-user feedbacks
- The list of actions.

Sets are registered to the infrastructure Information Service as Data Structure resources. This will allow the Manager Service to automatically orchestrate the "execution" of Sets of actions based on the data flow phases mentioned above. In particular, the Information Service will introduce a notion of "workflow" Data Structure Resource, describing the status of the data flow phases. Combined with time-based events, this data structure will enable the Manager Service to fire the whole data flow over time and to automatically schedule the execution of its different phases by:

- Harvesting the external data sources
- Adding sets of actions relative claimed publications and publication-project inference
- De-duplicating for the first time
- Adding the inference actions and end-user feedbacks
- De-duplicating for the second time

It should be noticed that the notions of action sets and workflow phases are configurable and the relative management services are agnostic from the specific configuration. This gives to the system maximum flexibility and ability to cope with evolving requirements.

The service offers APIs to create Sets and execute (i.e. promote) the actions in a set, i.e. apply the actions to the information space. The APIs also offer methods to add, update, delete or search complex actions by set, agent (creator of the action), and time interval. Complex actions are defined by means of XML records corresponding to OpenAIRE Dublin Core and OpenAIRE DataCite profiles, to simplify the life of services that need to interface to the info space to apply changes. The API also offers interfaces (libraries) to insert low- level actions, for those services willing to operate at a more detailed level.

The **Action Store Service** is implemented on top of HBASE, in a table different from the one of the Information Space. This allows running MapReduce jobs to efficiently apply or remove the actions from the information space benefiting from the framework ability of reading and writing across different tables in the same HBASE cluster. The data stored in the table dedicated to the Action Store Service can be divided into two distinct groups:

- Complex Actions are sets of Atomic Actions, typically originating from Information packages (xml records), e.g. a Dublin Core record will yield one publication action and a set of author actions. Atomic Actions are generated according to a specific domain dependent mapping. Complex Actions are persisted so that the Action Manager Framework can generate different sets of Atomic Actions (by updating the mapping), according to changes in the domain specific requirements.
- Atomic Actions are the minimum datum stored by the Action Store Service. In the OpenAIRE infrastructure they can represent the set of attributes belonging to a publication, or representing a person entity, e.g. an author.

The “promote actions” workflow reads all the Atomic Actions belonging to a specific action set (or all the action sets defined in the system), and writes them in the Information Space HBASE table. The operation is made possible by the coordinate attributes available in the atomic actions:

- Target row ID: the row key identifier in the Information Space table that will receive the write operation;
- Target Column Family: column family in the Information Space table that will receive the write operation;
- Target Column Qualifier: column qualifier in the Information Space table that will receive the write operation.

Thanks to the coordinates above the Atomic Action value can be written in the Information Space, contributing to enrich or fix entity attributes, or adding relationships among existing entities.

Bibliography

1. Manghi P., Manola N., Horstmann W., Peters D. (2010). An infrastructure for managing EC funded research output - The OpenAIRE Project. In: TGJ An International Journal on Grey Literature, vol. 6 (1) pp. 31 - 40. Grey Literature Network Service.

2. Manghi P., Bolikowski L., Manola N., Schirrwagen J., Smith T. (2012). OpenAIREplus: the European Scholarly Communication Data Infrastructure. In: D-Lib Magazine, vol. 18 (9/10) article n. 1. Corporation for National Research Initiatives (CNRI).

3. Schirrwagen J., Manghi P., Manola N., Bolikowski L., Rettberg N., Schmidt B. (2013). Data curation in the OpenAire scholarly communication infrastructure. In: Information Standards Quarterly, vol. 25 (3) pp. 13 - 19. NISO (ed.). National Information Standards Organization.

4. Tom White. Hadoop the Definitive Guide, 3rd Edition. 2012. O'Reilly Media.

5. Lars George. Hbase the Definitive Guide. 2011. O'Reilly Media.

6. Nicholas Dimiduk and Amandeep Khurana. Hbase in Action. 2012. Manning.

7. Donald Miner and Adam Shook. MapReduce Design Patterns. 2012. O'Reilly Media.