SEVENTH FRAMEWORK PROGRAMME

CAPACITIES

Research Infrastructures

INFRA-2009-1 Research Infrastructures

OpenAIREplus

Grant Agreement 283595

“2nd-Generation Open Access Infrastructure for Research in Europe

OpenAIREplus”

D6.4 Specification of the Authority File Service (De-duplication Service)

Deliverable Code: D6.4

Document Description

Project	
Title:	OpenAIREplus, 2nd Generation Open Access Infrastructure for Research in Europe
Start date:	1st December 2011
Call/Instrument:	INFRA-2011-1.2.2
Grant Agreement:	**283595**

Document	
Deliverable number:	D6.4
Deliverable title:	Specification of the Authority File Service
Contractual Date of Delivery:	30th of September, 2012
Actual Date of Delivery:	
Editor(s):	Paolo Manghi
Author(s):	Paolo Manghi, Marko Mikulicic, Claudio Atzori
Reviewer(s):	
Participant(s):	
Workpackage:	WP6
Workpackage title:	OpenAIREplus data model and content management services
Workpackage leader:	CNR
Workpackage participants:	NKUA, CNR, UNIBI, UNIWARSAW, CERN, DTU, EKT-NHRF, EMBL, KNAW-DANS, STFC
Distribution:	Public
Nature:	Deliverable
Version/Revision:	v1.0
Draft/Final:	Draft
Total number of pages: (including cover)	
File name:	

Key words:	Deduplication, merge, authority file management

Disclaimer

This document contains description of the OpenAIREplus project findings, work and products. Certain parts of it might be under partner Intellectual Property Right (IPR) rules so, prior to using its content please contact the consortium head for approval.

In case you believe that this document harms in any way IPR held by you as a person or as a representative of an entity, please do notify us immediately.

The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content.

This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the OPENAIRE consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 27 Member States of the Union. It is based on the European Communities and the member states cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors. (http://europa.eu.int/)

OpenAIREplus is a project funded by the European Union

Table of Contents

Table of Figures

Summary

The goal of this deliverable is to (i) describe the motivations behind the realization of an authority file management service in OpenAIREplus, (ii) explicit the requirements of such a service, and (iii) define the specification of the service. Finally, it proposes a high-level description of the technical solution devised in the project and currently driving the implementation of the service.

1 Motivations and requirements

As depicted in Figure 1, and described in Deliverable D7.2, the overall materialization of the the **native information space**, i.e. the space containing objects collected from external data sources or provided by users, is to be *cached* in MDStores and Relational databases and *materialized* in an HBase cluster.

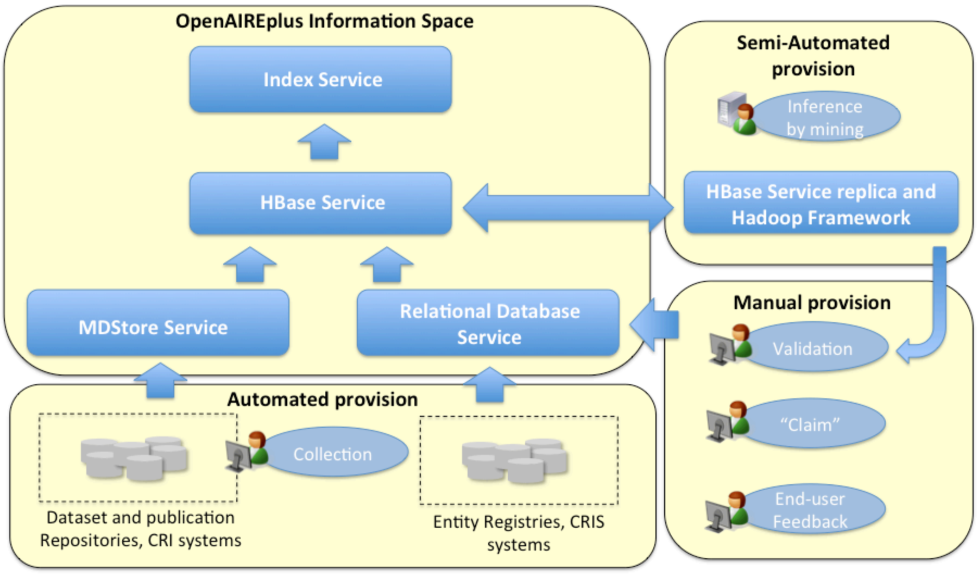
OpenAIREplus Information Space
Index Service
HBase Service
MDStore Service
Relational Database Service
Semi-Automated provision
Inference by mining
HBase Service replica and Hadoop Framework
Manual provision
Validation
"Claim"
End-user Feedback
Automated provision
Collection
Dataset and publication Repositories, CRI systems
Entity Registries, CRIS systems

Figure 1 – Information Space services and data flow

An obvious consequence of "aggregating" content from various sources is to collect and insert duplicated content in the information space. In fact, although different sources generally deliver objects that are unique in the data sources scope (even though in some critical case this is not the case), it is often the case that different data sources provide objects (e.g. metadata descriptions) relative to the same real-world entity (e.g. the same publication in two different repositories, the same person information from different entity registries).

In order to provide an unambiguous information space, thus effective browsing functionality and precise statistics, duplicates must be removed. Removing duplicates from the information space is an action that cannot be performed manually by data curators. For example, consider person objects collected from datasets and publication repositories. One new person object will be created for each author of any publication collected and inserted in to the space; the process will for generate different records for the same author, even if the author publications are collected from the same repository (other techniques for generating stable identifiers for persons are possible, see deliverable D6.1). This process generates an average 7-8 authors per paper, making it impossible for humans to analyze, identify and finally *merge* duplicate authors into one. Moreover, independently of the entity typology, when two or more objects are identified as equivalent and merged to produce one *representative object*, the relationships with other objects that merged objects had, must be reassigned to the representative object following a precise strategy. Again, humans cannot undertake the application of such strategies without running the risk of generating serious data inconsistencies.

Authority control is an activity operated by a group of expert data curators, whose task is to ensure disambiguation of collections of objects. Disambiguation of a collection is ensured by resolving *object duplication* that is when a set of objects of the collection represents the same real-world entity. Duplication issues are addressed by *merging* actions, whose consequences are to collapse the objects into one *representative object*.

Authority control tools are systems supporting a group of data curators with automatic *candidate identification techniques* capable of efficiently spotting candidate object sets for merging in object collections of arbitrary dimension.

2 De-duplication strategy

The De-duplication Service is inspired by well-known de-duplication strategies, based on two main concepts:

1 *similarity functions,* which return a 0 . . . 1 similarity distance measure between two records (with 1 they are equal), and

2 *object matching algorithms,* which cope with the optimization of otherwise $O(n^2)$ complexity required to compare all pairs in a collection of n records.

2.1.1 Similarity function

The De-duplication Service offers a highly configurable similarity function. Given two rows r1 and r2 of identical structure [l_1:K_1,... ,l_n : K_n] (i.e. same fields possibly different values), the similarity distance of two objects (HBase Service rows) is measured by the formula:

$$F_s(r_1,r_2)=\sum_{i=1}^{n}(f_i(r_1,l_i,r_2,l_i)\times w_i)\,/\,\sum_{i=1}^{n}w_i$$

Where $r_1.l_i$ is the value of the field l_i *of the record* r_1 and f_i is a distance function 0…1 on strings between two given strings. Finally w_i is the weight of the field l_i with respect to the overall distance, and the sum of all w_i is 1. The service already supports several standard distance functions between strings and allows to plug-in new ones. Some of the available ones are: Jaro-Winkler (Winkler, 1990), Cohen's variants of the TFIDF metrics (Cohen et al., 2003), Edit distance, Biagram distance or the 'person name and address matching' problem (Christen and Zhu, 2002) (Churches et al., 2002).

Two objects are considered as equal when their distance is beyond a given threshold *T*, to be tailored by the data curators based on the functionality function and the collection of objects at hand.

2.1.2 Objects matching

As mentioned above, the complexity upper bound for the identification of all possible candidate object pairs is O(n^2), where n is the number of records in the collection. Ideally, the function F_S is calculated over all possible object pairs in the collection and the results are then filtered out to keep only those pairs whose similarity is beyond *T*. Due to the amount of objects involved in OpenAIRE aggregation collections, possibly scaling up to tens of millions for persons and results, and the potentially high number of duplicates, the time for such calculation as well as the time for the subsequent creation of representative entities might be "unacceptable", in the order of days. The technical challenge is therefore dual:

1. Finding methods and algorithms capable of reducing complexity while identifying candidates with minimal false positives/negatives and
2. Delivering implementations capable of scaling up with the number of records and the I/O costs needed for reading, writing and sorting by fine balancing between RAM and disk operations.

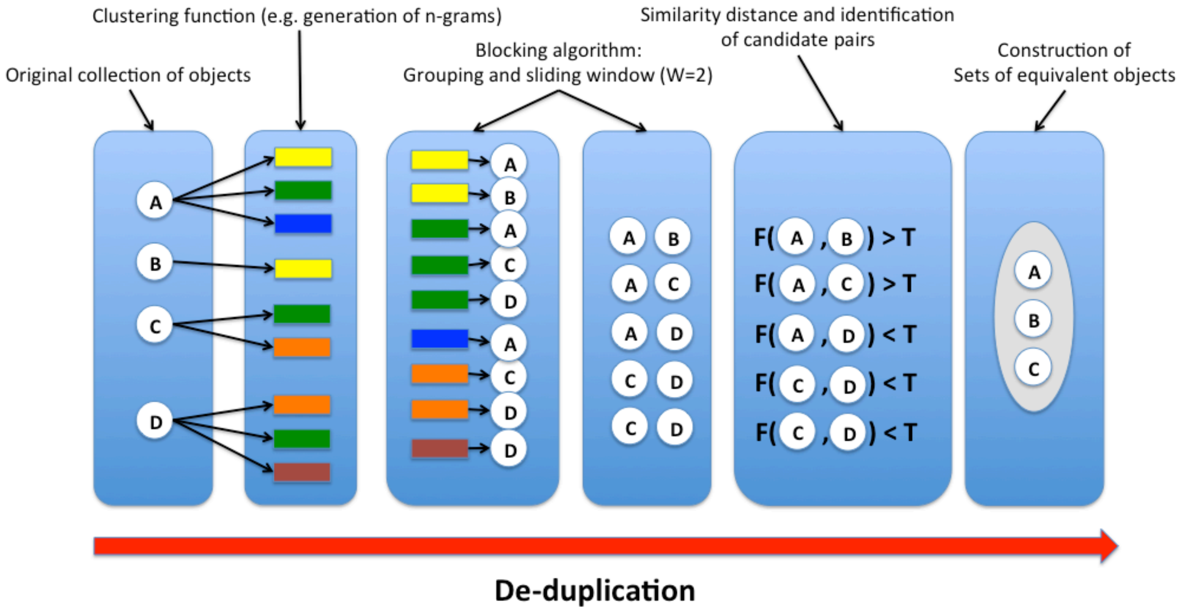
Clustering function (e.g. generation of n-grams)
Similarity distance and identification of candidate pairs
Blocking algorithm: Grouping and sliding window (W=2)
Original collection of objects
Construction of Sets of equivalent objects
A
B
C
D
F(A , B) > T
F(A , C) > T
F(A , D) < T
F(C , D) < T
F(C , D) < T
De-duplication

Figure 2 - De-duplication of a collection: logical steps

The De-duplication Service addresses these problems by means of the logical steps depicted in Figure 2. The Service implements a *blocking algorithm* based on an n-gram "clustering" function. The idea behind blocking algorithms is that of grouping objects by a common pattern, i.e. values resulting from applying the clustering function to the object. Such a pattern should be defined in such a way that resulting groups may pre-suggest objects candidates for merges, so that only pairs in a collection would be suitable for distance matching. For example, a collection of person objects may be ordered by the field *surname*, so that only objects with the same surname values would be paired up to calculate a distance function over all fields. Clearly, the choice of a proper clustering function is crucial since the object scopes will yields may bring in a loss of recall and precision due to the likelihood that true candidate record pairs have been excluded from the matching.

The Service allows administrators to customize the clustering function, but supports an implementation of an n-gram function. The function, given a string, returns all its substrings of length *n*. It is therefore very useful to group together objects whose field values are equal but differ due to missing characters, typos, character swaps, etc. In its current implementation, the function generates a number of n-grams for each object, based on (i) a given *n* and (ii) on a subset of the objects fields to be specified as parameters. n-grams are pre-fixed with the name of the field from which they were generated (<*field_name*>*:*<*ngram*>) and in general several n-grams are generated from the same field values.

Once objects are grouped by the values <*field_name*>*:*<*ngram*> (boxes of the same color in the Figure), for each group the Service calculates the distance between all pairs of objects in the group. Such pairs are calculated by applying a *sliding window algorithm*. The algorithm operates over a group of objects as obtained at the previous steps; it starts from the first object, namely the *pivot*, and generates all pairs of objects including the pivot and the objects that follow the pivot up to a window of length *W*. When finished, the algorithm takes as pivot the second object and iterates again. Until all objects have been used as pivot. The sliding window allows to further reducing the number of candidate pairs, thereby offering another tool to reduce time at the cost of precision and recall.

For each pair, the Service calculates the distance function and rules out pairs of objects whose measure falls below the threshold *T*. The resulting set of object pairs, however, is not enough to conclude the de-duplication process, since such pairs may in fact lead to sets of equivalent objects. The Service calculates such sets and returns them to the last de-duplication step, to apply merging actions.

Table 1 specifies the list of parameters available to data curators willing to customize the de-duplication process.

Table 1 - De-duplication Service: parameters

Parameter	***Description***
n	Length of the n-grams to be extracted by the clustering function
Fields	Subset of object fields to be used by the clustering function
K	Minimal threshold of similarity between two objects
W	Length of the sliding window in the blocking algorithm
w_i	Weight of the field l_i in the evaluation of the similarity function F_S
f_i	Distance functions on strings

3 De-duplication implementation

The De-duplication Service is designed to identify sets of candidate objects within a collection. To this aim, the service operates over the OpenAIREplus information space as stored in the HBase Service and by exploiting Map-Reduce Hadoop framework facilities. The HBase Service stores information space objects as described in deliverable D6.2. In particular, a collection of objects is encoded as a set of HBase rows whose columns contain serialized versions of the objects properties and relationships with other objects.

The Service implements the strategy presented in the previous section by means of two sequential map-reduce jobs.

Map-Reduce job to produce candidate duplicate pairs

- Map: extraction of n-grams from records. The configuration of the algorithm allows defining the length of the n-gram as well as the fields to be used for the extraction (note that multiple different fields may also be treated as one). As a result, a Map run produces pairs of the kind (field_name:ngram, record).
- Reduce: the reduce process groups all pairs with the same keys <field_name>:<ngram> and applies "blocking" techniques to generate all possible pairs of candidates together with their *similarity measure*. The effect is to create a relationship "similar" between two objects when their similarity measure goes beyond a given threshold *T*.

Map-Reduce job to build sets of equivalent records from pairs of equivalent records

The Map-Reduce job visits the graph of "equivalent pairs" built in the previous step and returns the sets of records that are considered equivalent to each other. This action has the effect of creating a new representative object for the set as well as relationships *mergedWith* from the merged objects to the representative object.

3.1 Exploiting de-duplication by contextual information

As mentioned in deliverable D6.1, object de-duplication is designed as a flexible process, completely agnostic from the structures, i.e. the properties, of the objects to be de-duplicated. This means that, whatever contextual information can be attached to the objects and be encoded in terms of properties, can be properly used to improve the accuracy of the de-duplication process. The HBase implementation of the information space is particularly apt for object contextual enrichment, in the sense that the model allows for an arbitrary number of columns to be added to the rows (objects). In principle, data managers can run map-reduce jobs to build contextual information of the objects and attach it to the objects, in order to refine their de-duplication algorithms. For example, Person objects in the OpenAIRE data model may be enriched with the list of their co-authors and the list of the co-authors of their co-authors. This can be obtained by running map-reduce jobs to identify co-authorships via the author publications up to the second level of publications. Matching such sets for two candidate Person objects may notably reduce the chance to collapse homonyms, i.e. different authors with the same name and surname, into one object by mistake.