DELIVERABLE

Project Acronym:	**Europeana Cloud**
Grant Agreement number:	**325091**
Project Title:	**Europeana Cloud: Unlocking Europe's Research via The Cloud**

D4.3 A report and a plan on future directions for improving metadata in the Europeana Cloud

Revision: FINAL

Authors:

Marian Lefferts, Consortium of European Research Libraries (editor)
Cesare Concordia, ISTI-CNR
Lucas Anastasiou, Open Univeristy
Alexander Jahnke, Data Conversion Group
Maike Kittelmann, Data Conversion Group
Hugo Manguinhas, Europeana Foundation

Project co-funded by the European Commission within the ICT Policy Support Programme		
Dissemination Level		
P	Public	P
C	Confidential, only for members of the consortium and the Commission Services	

Revision History

Revision	Date	Author	Organisation	Description
0.1	20/01/201	Cesare Concordia	ISTI-CNR	Initial draft of report on task 4.3.1.
0.1	20/01/2016	Lucas Anastasiou	The Open University	Initial draft of report on task 4.3.2.
0.1	18/01/2016	Alex Jahnke and Maike Kittelmann	Data Conversion Group, Göttingen, on behalf of CERL	Initial draft of report on task 4.3.3.
0.2	26/01/2016	Marian Lefferts	Consortium of European Research Libraries (CERL)	Editing
0.2	27/01/2016	Cesare Concordia	ISTI-CNR	Additional example
0.2	27/01/2016	Marian Lefferts	CERL	Executive Summary, Introduction, Conclusions and Recommendations
0.2	28/1/2016	Els Jacobs	Europeana Foundation	Edited Executive Summary
0.2	29/1/2016	Lucas Anastasiou	The Open University	Second draft of report on task 4.3.2.
0.2	29/01/2016	Cesare Concordia	ISTI-CNR	Additional example
1.0	2/2/2016	Marian Lefferts Hugo Manguinhas	CERL Europeana Foundation	Final edit References to work done on the Europeana Semantic Enrichment Framework
REVIEW	11/2/2016	Vladimir Alexiev	Ontotext	Review and comments
FINAL	13-15 Feb 2016	Cesare Concordia, Lucas Anastasiou and Marian Lefferts	ISTI-CNR OU CERL	Responses to suggestions put forward by the reviewer

Executive Summary

The aim of the task reported in this Deliverable 4.3 was to explore how we could arrive at shared metadata enrichment, by making the most of the large amount of data already gathered in Europeana and the Cloud environment developed in the project.

We explored whether we could enrich the data by comparing Europeana data with data from **external** sources (task 4.3.1). Secondly, we explored whether there was data **internally** in the large Europeana set that could meaningfully be connected to other data in the set (task 4.3.2). Both approaches would offer better contextualisation of Europeana data for the end user.

In Task 4.3.1, ISTI-CNR demonstrated the use of image recognition techniques to explore possible overlap between the datasets of Europeana and the WikiArt data set. As a side effect, it showed that duplicates within the Europeana data set might also surface. The pilot showed how information from an external resource such as WikiArt can potentially be used to enrich the data that is already held in Europeana. A second test phase in February 2016 will increase the Europeana and WikiArt datasets, ensure a more precise selection and focus on paintings held in Europeana, improve matching against WikiArt, and enhance performance.

In Task 4.3.2 the Open University set up two similarity services, one based on Solr and one on ElasticSearch, which are made available through a REST API, combined with appropriate documentation. The tests show the usefulness of this service in discovering near-duplicates (or near identical records) deposited in Europeana Cloud. The desired level of similarity can be set by the end-user, and so that we have the opportunity to improve contextualisation by providing recommendations on related (but not identical) content that exists in the Europeana Cloud platform.

Task 4.3.3 , executed by the Data Conversion Group, Göttingen, on behalf of CERL, aimed to explore whether editing of data in the Cloud would be possible (either manually or via batch-editing). It became apparent that this is not an appropriate use case for the Cloud infrastructure. Data correction typically takes place on the side of the institutions, and Europeana benefits from these improvements after the data is (re-)uploaded in the Cloud. DCG explored what features the European Cloud infrastructure must have to support the necessary grouping and accessing of the various versions of datasets, in a variety of formats, uploaded over time. For Europeana Cloud to operate as a viable host for purely metadata based applications, such as CERL's Heritage of the Printed Book database, it must (1) give access to the digital object itself, (2) offer version control on dataset level, (3) support handling of hierarchically structured metadata, (4) offer users rights management on dataset level (in addition to (5) rights management on individual record level, which is already provided).

Table of contents

Introduction

The aim of the task reported in this Deliverable 4.3 was to explore how we could arrive at shared metadata enrichment, by making the most of the large amount of data already gathered in Europeana and the Cloud environment developed in the project.

We looked at approaches to detect duplicates and analysed data to explore opportunities for data enrichment. Ultimately, the work in this Europeana Cloud project task should lead to better contextualisation of Europeana data for the end user.

Task 4.3.1 explored whether we could enrich the data by comparing Europeana data with data from **external** sources (task 4.3.1). In its report below, ISTI-CNR demonstrates the use of image recognition techniques to explore possible overlap between the datasets of Europeana and the WikiArt data set, and notes that WikiArt metadata can potentially be used to enrich Europeana records.

Task 4.3.2 explored whether there was data **internally** in the large Europeana set that could meaningfully be connected to other data in the set. The task was executed by the Open University, who set up two similarity services, one based on Solr and one on ElasticSearch, which are made available through a REST API, combined with appropriate documentation.

From the Description of Work:

Task 4.3 Connecting Research Material in the Cloud: Exploring Shared Metadata Enrichment [M6-36]

4.3.1 Enrichment of metadata (ISTI-CNR) [M24-36] ISTI-CNR will mine external databases for events with the aim to enrich the Europeana metadata by identifying specific events that are cited within EDM-compliant metadata. The outcome will be additional enrichment plugins in UIM to enrich Cloud data as part of the content ingestion workflow. Milestone 4.8 [M36]

4.3.2 Improve metadata based on content available in the Cloud. [M24-36] OU will compare and implement different models for connecting content based on semantic similarity of full text or metadata. Selected methods will be provided as a service for the Cloud. The outcome will be additional plugins in UIM to enrich Cloud metadata as part of the content ingestion workflow [M36]. Milestone 4.9

4.3.3 Establish Pilots for editing of data in the Cloud [M24-36]
a. manual editing
Together with CERL, EF will set up a pilot for manual data curation of subsets of metadata or specific aspects of metadata and explore its impact on the Europeana Research platform. E.g. a curator might want to edit a single metadata record or replace a specific value in the metadata with a controlled term.

b. batch editing
Together with CERL, EF will set up a pilot for editing of batches of records executed by an expert organisation that does not hold data in the cloud, and will explore its impact on the Europeana Research platform. The outcomes of the pilots will be included in Deliverable D4.3 [M36]

Task 4.3.3 , executed by the Data Conversion Group, Göttingen (DCG), on behalf of CERL, deviated from the original Description of Work, which called for an exploration of manual and batch editing of data. Such editing typically takes place at the host institution, and the data is then (re-)uploaded in the Cloud. This means that a (large) number of versions of datasets and individual records may be processed by any given institution. For this task, DCG therefore explored what features the European Cloud infrastructure must have to support the necessary grouping and accessing of the various versions of datasets, in a variety of formats, uploaded over time.

Report on task 4.3.1. - an application for Metadata Enrichment and for Duplicate Detection

1. Introduction

This document describes the activities of the ISTI-CNR team in task 4.3: the design and implementation of an application for Metadata Enrichment and for Duplicate Detection based on image recognition techniques. The application developed is called IREnrichment, and has been implemented in Java. The Europeana team working on Semantic Enrichment are following this task with considerable interest.

2. Metadata Enrichment: definition and terminology

In this document we adopt the definition of metadata enrichment given in the final report released by the Europeana Enrichment and Evaluation task force [1] (the ISTI-CNR team were members of this task force). Basically, 'enriching a metadata record describing an object' means 'add to it new properties about the object described, or correct existing properties.' The added properties can be *typed links, untyped links,* or simple literal *tag* values.

The enrichment process comprises a number of manual or automatic activities and the main components involved in these activities are [1]:

- *Source*: the objects whose metadata is being enriched (by extension it will also refer to the metadata set about these objects)
- *Target*: the datasets used to enrich the source metadata. Targets can be of different types, from literals to resources published as linked data
- *Rules*: enrichment rules specify how the enrichment between the source and target should be executed.

In the context of this document, 'enrichment' is always conceived as being applied to Europeana metadata records, so our source is the Europeana dataset.

3. Using Image Recognition in enrichment rules

Typically, in the context of automatic enrichment, rules take the form of instructions based on matches between a defined set of property values describing the source object and the correspondent values in the target resources. A resource in the target is considered as a candidate for the enrichment of a source record if these values match.

Some enrichment frameworks are based on rather simple rules to discover when a resource from the target should be connected to a source record, for instance lexical or numerical value comparison of the metadata properties. More sophisticated enrichment services introduce *distance measurements* for the matching, for instance the DM2E project uses the Silk[1] framework to align places and agents with DBPedia; this improves precision but has introduced some heavy performance problems.

[1] http://silk-framework.com

Generally speaking the definition of rules requires an excellent knowledge of both source and target datasets, and even with such a knowledge there could be issues that are not easy to resolve, e.g. ambiguous matching and definition of priorities for metadata property matching (see the section titled 'Linking rules for enrichment' of [1] for an extensive discussion on this topic).

Essentially we can say that approaches for rule definition presented in [1] consider mostly **textual** content: exact string matching, string similarity, Natural Language Procesing (NLP) techniques etc.; our idea is to extend these rules with **image based** options, by adopting Image Recognition techniques.

In practice we want to define rules for target resource discovery by using the digital image of the object as one of the properties used for matching. This, of course, is valid for those objects having a digital image representation, for instance paintings or photos.

4. Image Recognition metadata enrichment in Europeana

The current Europeana metadata enrichment process is described in detail in [2]. Essentially it consists of "the creation of links to controlled vocabularies representing contextual resources such as places, concepts, agents and time periods", these links are called contextual links and are added to the original metadata records. For every contextual class of resources at least one target is used: Geonames for places, DBPedia and GEMET for concepts, DBPedia for agents and Semium Time for time [2].

The main goal of our activity is to investigate if adding a 'visual' option to enrichment rules could i) expand the enrichment processes in use at Europeana and ii) help Europeana to fix the ambiguous matching issue in a more effective way.

Out of 48M objects in Europeana, 28M are Images, and about 10M of them have links to digital images of the resource described ("items with links to media"). These objects are clearly candidates for our image recognition enrichment rules (e.g. if we have a painting then the image describing it is (most likely) a picture of the painting). By contrast the image associated to an audio resource can be chosen according to variable criteria: for a song we can have a picture of the record sleeve (http://www.europeana.eu/portal/record/2022608/DF_DF_13399.html), a photo of the singer (http://www.europeana.eu/portal/record/2051912/data_euscreenXL_513570.html), a wave-form, or even the musical score. Many Text objects in Europeana have images representing the scanned text, but it is not such a good candidate for Image Retrieval.

This means that the approach we are adopting is more likely to give meaningful results for metadata records describing paintings and pictures, and we have focused on these kinds of records in our activity.

5. Implementation

As stated above, a goal of our work is to understand if this approach can help us to supplement the array of metadata enrichment processes already in use at Europeana.[2] To do this we have designed and built a framework implementing the image recognition enrichment approach for the Europeana

[2] There is a significant difference between the enrichments that Europeana currently undertakes and image recognition proces described here. Europeana links object metadata with external sources. It does not import metadata statements as suggested here. This is why this approach needs further careful consideration before it can be implemented.

source. The first three steps in our implementation work have been: select an Image Recognizer service, choose an appropriate target and define enrichment rules.
From the implementation point of view a major issue of the IREnrichment application is the choice of the IR service. In this phase of the activity we are interested in a service that:

1. is freely available,
2. enables us to define a large target (WikiArt has 60000 images)
3. gives us the possibility of execute an unlimited number of queries for image matching.

Additionally the IR service should have a web API, referring it as a web resource would avoid us to create direct dependencies between the IREnrichment and the specific service.
Another important requirement for this choice is the licence of use of the service, in order to be used in Europeana platform it must have an Open Source compliant licence.

A number of Image Recognition services are available on the web[3] and many of them have interesting features, however the size of target dataset can be a critical point: usually free services doesn't allow users to build large image indexes or in some cases don't give this possibility.

The first two candidates we found are: Pastec and VeMIR. VeMIR is a Recognizer Builder tool developed by Visual Engines [5] providing a web GUI and a Web API. The Visual Engines hosts the Image Recognizer (Software as a Service) and the image index on their servers, and apparently there is no limits to the size of the index. The other option we considered is the framework Pastec (http://pastec.io) that would have required us to install and manage the Image Recognition platform on a local server. In particular using Pastec with our current target dataset, would have required a powerful server with a big storage size to store images and indexes, indeed WikiArt publishes about 60,000 paintings, available in VeMIR.

We decided to start our activity using VeMIR as our image recognizer, mainly because it is ready to use, but we are also setting up the requested resources to create a server where we will install the Pastec framework, in order to run comparative tests between the two solutions[4]. In the future we will select and test other frameworks as recognizer server, in particular we are also interested in evaluating the use of Google Images.

[3] An incomplete list is here:https://market.mashape.com/api-collections/image-recognition
[4] Note that the VeMIR IR service is freely available but not open source, this means that, unless there will be particular agreements, it could not be used in a possible production release.

Figure 1 The VeMIR recognizer builder tool

Using the Web GUI functionalities of VeMIR it is possible for a registered user (registration is free) to create or delete an index of images; the index is hosted on the VeMIR server. Remote applications can access the index via the VeMIR Web API to index images or to execute queries for image recognition. In the index every image can be associated to a hyperlink.

The main criterion adopted in the choice of the target has been the need of having a dataset providing images of paintings with a medium or high "image resolution" (in order to reduce the possibility of creating indexing errors) and for every image a set of metadata properties.

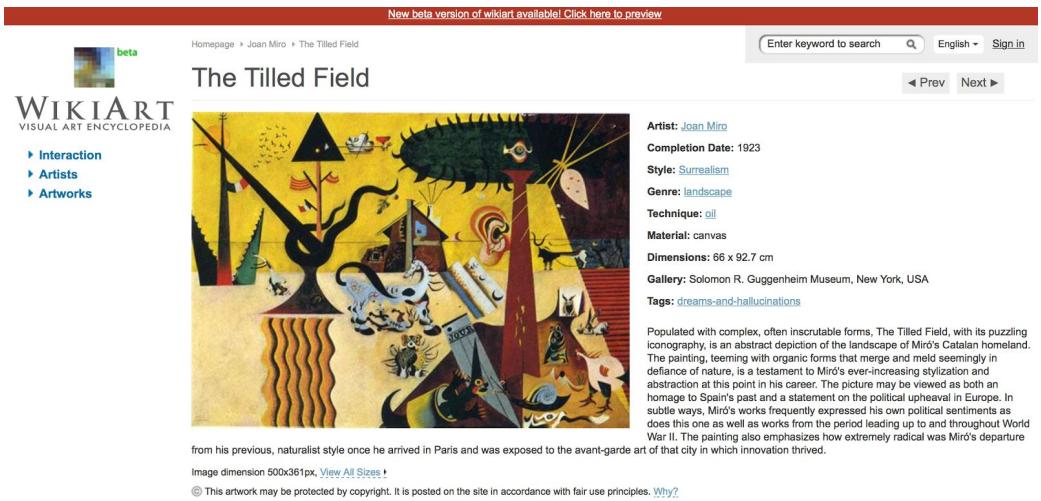


Figure 2 a WikiArt page describing a painting

We decided to use WikiArt [6] an online, user-editable visual art portal (formerly called WikiPaintings). Figure 2 shows a WikiArt web page with the description of a Miro painting.[5]

The metadata property set used in WikiArt is quite basic: the title, the creation date, the style of the painting, the technique adopted, the material, dimensions etc.

In defining rules we started with a minimal approach: the only option used in enrichment rules defined for our first implementation is image matching, i.e. the target resource is an enrichment candidate if its image is *recognized* as being similar to the image of the source. In the next phase we plan to introduce more complex rules.

6. How the IREnrichment application works

Since a dump file with the dataset of WikiArt is not available,[6] we implemented a web application that crawled the WikiArt portal and ingested images in the VeMIR index. In the VeMIR index every image is stored together with the URL of the WikiArt web page describing it. Currently we have indexed roughly 30,000 images of paintings.

The IREnrichment application gets images from the source (Europeana dataset), checks if those images match with images in the VeMIR index and stores information in case of a positive match.

The IREnrichment application works as follows:

1. Queries the Europeana API for Cultural Heritage Objects (CHO) containing images whose author is a painter (see section 8 for details), and creates a local database for these images.
2. For every image performs the image recognition procedure (described below)
3. If a match is found, it creates a new enrichment candidate entry and adds two properties: the image id as identifier and the URL of the WikiArt page returned by the VeMIR recognizer as an edm:hasView value.
4. Downloads the WikiArt web page and parses it to find property values, and these values are added to the enrichment record (mapping described below)
5. Saves the enrichment record in an RDF file and in a local database (MongoDB)[7]

The image recognition is implemented using a specific "recognize" web service published in the VeMIR Web API[8]: using the HTTP POST method, the IREnrichment application sends the source image, the credentials of the user and the name of the index to the recognize service. The recognize service returns to the IREnrichment a list of pairs

(*reference*, *number*)

[5] It should be noted that the images in WikiArt are not freely available and the images were only used for the purposes of this test.
[6] We are in the process of evaluating other sources, e.g. Wikidata.
[7] The use of an RDF repository is planned for the future. For the moment, records are saved on MongoDB since this is the DBMS used in the Europeana Enrichment Framework. The idea is to make the IREnrichment application compatible with this enrichment Framework. The IRFramework is independent from the storage implementation.
[8] A testing web page with a form is here: http://vemir.visualengines.com/miraserver/testingPage.html

where the reference is the URL to the WikiArt web page describing the target image and the number represents the 'trust level' of the result. If there are several matching images the service returns a list of pairs, if no match occurs the result-set is empty.

To test the application we created (using the Europeana 'search.json' API) a local database of links to images (see Annex 1 for details). In the database there are about 3,543 links to artworks taken from the Europeana dataset. Using the IREnrichment with this database as source, we have found 165 matches[9] [8]. This test has helped us to fix bugs and to understand which are the main problems of such an approach. Main outcomes obtained and issues are described in the following sections.[10]

6.1.Enrichment example

As first example, consider the Europeana resource[11] shown in Fig 3:

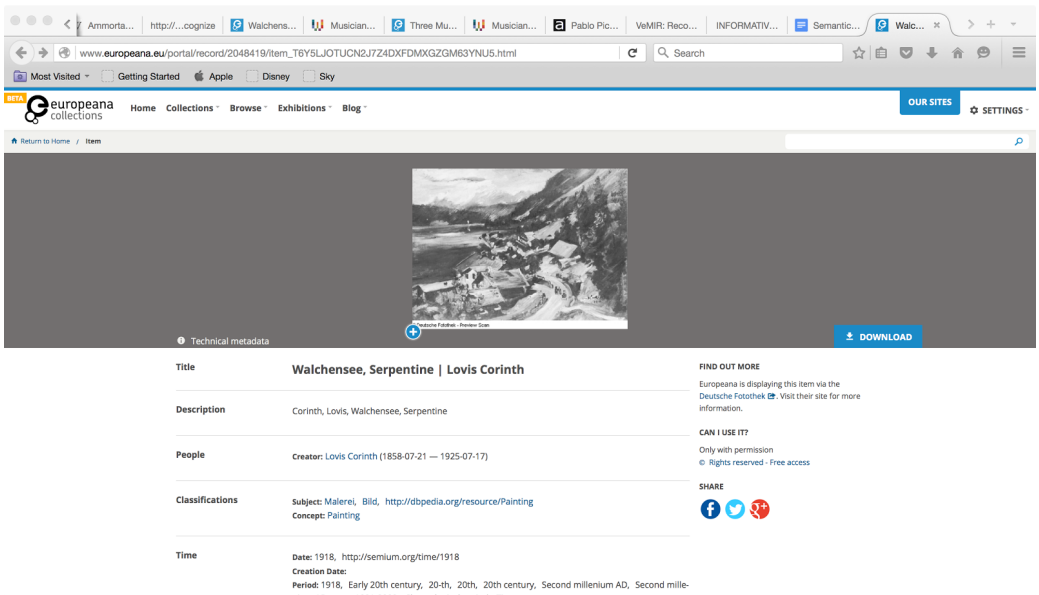


Figure 3 Europeana: Walchensee, Serpentine, Lovis Corinth

The resource describes a painting by the German painter Lovis Corinth.
The IREnrichment tool recognizes the image of the painting and discovers as target resource the WikiArt web page[12] in Fig. 4:

[9] This is a surprising low number. Further communication with the Visual Engines developers, revealed that the VeMIR recognizer has a bug and a number of images were actually recognized but no answer were sent to IRFramework, developers are working to fix it. Additionally, there is an undocumented 'threshold' number that can be set in the POST request in order to release service recognition parameters. Work continues after the completion of this report.

[10] The .xml file reporting current test results is available here:
https://dl.dropboxusercontent.com/u/630356/WikiArtEnrichment.xml

[11] http://www.europeana.eu/portal/record/2048419/item_T6Y5LJOTUCN2J7Z4DXFDMXGZGM63YNU5.html

[12] http://www.wikiart.org/en/lovis-corinth/the-walchensee-serpentine-1920?utm_source=returned&utm_medium=referral&utm_campaign=referral

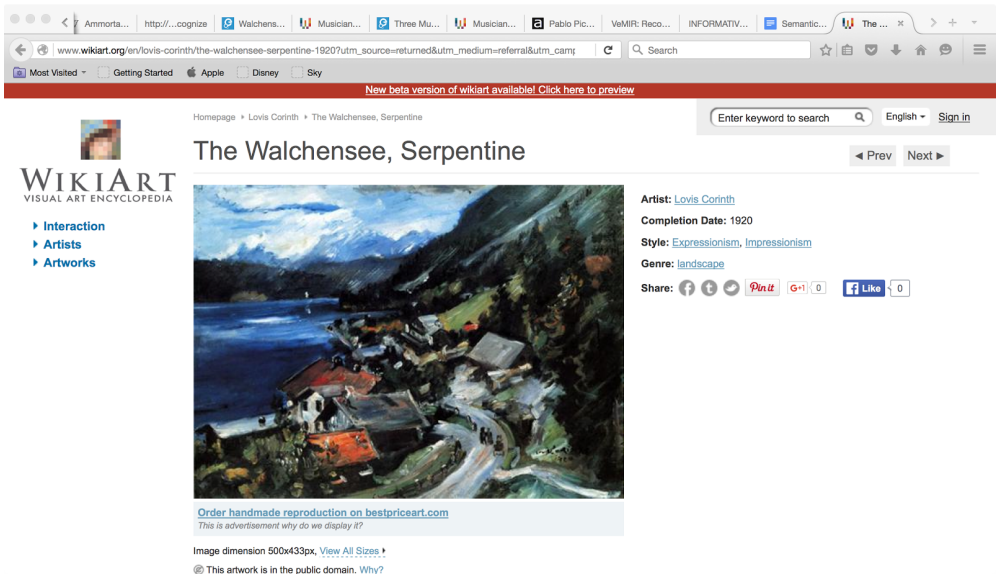


Figure 4 WikiArt, The Walchensee Serpentine, Lovis Corinth

The WikiArt metadata record has more properties than the record we have in Europeana, in particular it reports a 'genre' and the 'style' of the painting, these properties could be used to enrich the Europeana record.

The enrichment record extracted from the WikiArt database is the following:

```
<edm:ProvidedCHO>
    <dc:identifier xml:lang="en">http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/
0130000/df_hauptkatalog_0130903.jpg</dc:identifier>
    <edm:hasView xml:lang="en">http://uploads5.wikiart.org/images/lovis-corinth/the-
walchensee-serpentine-1920.jpg!Blog.jpg</edm:hasView>
    <dc:creator xml:lang="en">Lovis Corinth</dc:creator>
    <dcterms:created xml:lang="">1920</dcterms:created>
    <skos:prefLabel xml:lang="en">Impressionism</skos:prefLabel>
    <skos:prefLabel xml:lang="en">landscape</skos:prefLabel>
    <dcterms:format xml:lang="en">http://schema.org/Painting</dcterms:format>
    <dcterms:title xml:lang="en">The Walchensee, Serpentine - Lovis Corinth</dcterms:title>
    <edm:isShownAt xml:lang="en">http://www.wikipaintings.org/en/lovis-corinth/the-walchensee-
serpentine-1920</edm:isShownAt>
  </edm:ProvidedCHO>
```

This example is interesting because if we check the properties of the two records carefully, we notice that the values of the "Title" and "Date" are different: the title in the Europeana record is written without initial "The" and the creation date is 1918 in Europeana and 1920 in WikiArt. If we would have based the discovery process on the values of those properties we could have missed this match, which, according to image recognition is correct.

6.2.WikiArt to EDM mapping

We want to state in advance that mapping has been defined for testing purposes only. The main goal here is to try to identify cases where *string matching* based rules and *image recognition* based rules could give ambiguous results (see the example described in the section above) in order to investigate if the approach we propose can improve enrichment precision. Additionally at the moment we do not know whether the WikiArt portal can be considered a *reliable* target.

In case the Europeana team decides that WikiArt can be considered a trusted target for enrichment, an appropriate mapping will be defined.

As far as we know, the WikiArt portal does not have a document model describing the format used for metadata. However, checking the source code of WikiArt web pages it is easy to discover that metadata is embedded in HTML using the Microdata Tags[7]. Therefore we implemented an html parser[13] to get the properties needed. The table below reports the current mapping rules.

WikiArt	Europeana
Image	edm:hasView
Author	dc:creator
dateCreated	dcterms:created
Style	dc:subject
Genre	dc:subject
Title	dc:title
Description	dcterms:description
???	dcterms:format

An XML representation of the resulting records is the following:

```
<edm:ProvidedCHO>
  <dc:identifier>http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0753000/df_hauptkatalog_0753797.jpg</dc:identifier>
  <edm:hasView>http://uploads0.wikiart.org/images/pablo-picasso/musicians-with-masks-1921.jpg!Blog.jpg</edm:hasView>
  <dc:creator>Pablo Picasso</dc:creator>
  <dcterms:created>1921</dcterms:created>
  <skos:prefLabel>Cubism</skos:prefLabel>
  <skos:prefLabel>genre painting</skos:prefLabel>
  <dcterms:format>oil</dcterms:format>
  <dcterms:format>http://schema.org/Painting</dcterms:format>
  <dc:subject>allegories-and-symbols, music-and-dancing</dc:subject>
  <dcterms:title>Musicians with masks - Pablo Picasso</dcterms:title>
  <dcterms:title xml:dataset="europeana">Three Musicians I</dcterms:title>
  <edm:isShownAt xml:dataset="wikiart">http://www.wikipaintings.org/en/pablo-picasso/musicians-with-masks-1921</
edm:isShownAt>
  <edm:isShownAt xml:dataset="europeana">http://www.europeana.eu/portal/record/2048418/
item_FILYG5SJQBLIWVQURZB5K2WBHJDXVKYL.html?utm_source=api&amp;utm_medium=api&amp;utm_campaign=ctPEdiu7D</
edm:isShownAt>
 </edm:ProvidedCHO>
```

The identifier (<dc:identifier>) of the record is the URI of the image in the Europeana dataset. Beside the values listed in the table above, also other useful information are reported: the Europeana web page (<edm:isShownAt xml:dataset="europeana">), the WikiArt matching image, the *title* in both datasets etc.

6.3. Discovering duplicated metadata records

The identification of duplicated metadata records in a large digital library dataset such as the Europeana dataset is a complex problem. Over time, many de-duplication algorithms and tools have been developed [3]: there are algorithms based on string matching, others using string similarity measures and there are also algorithms where metadata records are mapped into graphs and graph analysis techniques are used to find duplicated items.

[13] based on the jsoup package

In our tests on a subset of Europeana records, we discovered that the approach based on image recognition could help to individuate possible duplicated records. As described above, in our approach when two images are recognized as being the same image, the records are considered as describing the same resource.

In the following examples we will see how suspected duplicates can be detected with our approach.

6.4.A second interesting example, duplicate records discovered?

Checking the xml file obtained by processing the local dataset we noticed the two records shown in the picture:

```
<edm:ProvidedCHO>
    <dc:identifier xml:lang="en">http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/
0753000/df_hauptkatalog_0753797.jpg</dc:identifier>
    <edm:hasView xml:lang="en">http://uploads0.wikiart.org/images/pablo-picasso/musicians-
with-masks-1921.jpg!Blog.jpg</edm:hasView>
    <dc:creator xml:lang="en">Pablo Picasso</dc:creator>
    <dcterms:created xml:lang="">1921</dcterms:created>
    <skos:prefLabel xml:lang="en">Cubism</skos:prefLabel>
    <skos:prefLabel xml:lang="en">genre painting</skos:prefLabel>
    <dcterms:format xml:lang="en">oil</dcterms:format>
    <dcterms:format xml:lang="en">http://schema.org/Painting</dcterms:format>
    <dc:subject xml:lang="en">allegories-and-symbols, music-and-dancing</dc:subject>
    <dcterms:title xml:lang="en">Musicians with masks - Pablo Picasso</dcterms:title>
    <edm:isShownAt xml:lang="en">http://www.wikipaintings.org/en/pablo-picasso/musicians-
with-masks-1921</edm:isShownAt>
  </edm:ProvidedCHO>
  <edm:ProvidedCHO>
    <dc:identifier xml:lang="en">http://fotothek.slub-dresden.de/fotos/df/ld/0021000/
df_ld_0021846.jpg</dc:identifier>
    <edm:hasView xml:lang="en">http://uploads0.wikiart.org/images/pablo-picasso/musicians-
with-masks-1921.jpg!Blog.jpg</edm:hasView>
    <dc:creator xml:lang="en">Pablo Picasso</dc:creator>
    <dcterms:created xml:lang="">1921</dcterms:created>
    <skos:prefLabel xml:lang="en">Cubism</skos:prefLabel>
    <skos:prefLabel xml:lang="en">genre painting</skos:prefLabel>
    <dcterms:format xml:lang="en">oil</dcterms:format>
    <dcterms:format xml:lang="en">http://schema.org/Painting</dcterms:format>
    <dc:subject xml:lang="en">allegories-and-symbols, music-and-dancing</dc:subject>
    <dcterms:title xml:lang="en">Musicians with masks - Pablo Picasso</dcterms:title>
    <edm:isShownAt xml:lang="en">http://www.wikipaintings.org/en/pablo-picasso/musicians-
with-masks-1921</edm:isShownAt>
  </edm:ProvidedCHO>
```

Figure 5 Duplication detected?

According to IREnrichment two different images (both shown in Fig. 6), are considered as similar to the painting described in the WikiArt page[14] shown in Fig. 7:

[14] http://www.wikiart.org/en/pablo-picasso/musicians-with-masks-1921

Figure 6 two images of a painting from Picasso found on Europeana

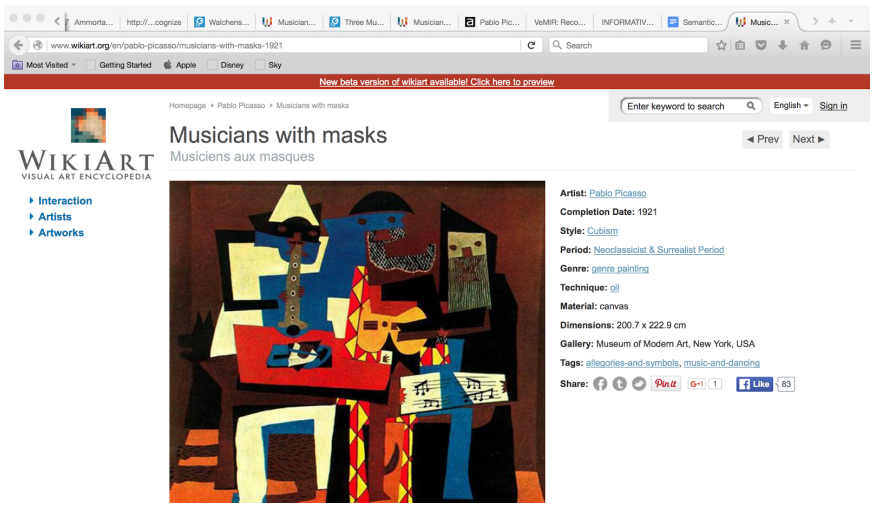


Figure 7 The WikiArt page describing the painting of Picasso

The images are stored in Europeana in two separate records[15] [16] and both records have the same property values except for the record and dataset identifiers. Apparently these two records describe the same object, and then we could have discovered a case of record duplication in Europeana. We

15 http://www.europeana.eu/portal/record/2048418/item_F7IFW3SI5NUOO5EAJHPPGDF7727JXX6P.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

16 http://www.europeana.eu/portal/record/2048416/item_VVOWNNOMTFKCEJM6BAQFTMPGUCZ3LXWV.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

will need to investigate if this is a duplication case, if not it would be useful to add further values to both records to distinguish each image.

6.5.Third example: duplicate detected and inaccuracy in title?

Checking the following xml elements makes another interesting discovery:

```
<edm:ProvidedCHO>
    <dc:identifier xml:lang="en">http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0192000/
df_hauptkatalog_0192717.jpg</dc:identifier>
    <edm:hasView xml:lang="en">http://uploads2.wikiart.org/images/raphael/putti-detail-from-the-sistine-
madonna-1513.jpg!Blog.jpg</edm:hasView>
    <dc:creator xml:lang="en">Raphael</dc:creator>
    <dcterms:created xml:lang="">1513</dcterms:created>
    <skos:prefLabel xml:lang="en">High Renaissance</skos:prefLabel>
    <skos:prefLabel xml:lang="en">religious painting</skos:prefLabel>
    <dcterms:format xml:lang="en">oil</dcterms:format>
    <dcterms:format xml:lang="en">http://schema.org/Painting</dcterms:format>
    <dcterms:title xml:lang="en">Putti, detail from The Sistine Madonna - Raphael</dcterms:title>
    <edm:isShownAt xml:lang="en">http://www.wikipaintings.org/en/raphael/putti-detail-from-the-sistine-
madonna-1513</edm:isShownAt>
    <edm:isShownAt xml:lang="en">http://www.europeana.eu/portal/record/2048413/
item_755CC3YZQGXKNXX5VPZEADZ6VHTZZGCR.html?utm_source=api&amp;utm_medium=api&amp;utm_campaign=ctPEdiu7D</
edm:isShownAt>
  </edm:ProvidedCHO>
  <edm:ProvidedCHO>
    <dc:identifier xml:lang="en">http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0130000/
df_hauptkatalog_0130047.jpg</dc:identifier>
    <edm:hasView xml:lang="en">http://uploads2.wikiart.org/images/raphael/putti-detail-from-the-sistine-
madonna-1513.jpg!Blog.jpg</edm:hasView>
    <dc:creator xml:lang="en">Raphael</dc:creator>
    <dcterms:created xml:lang="">1513</dcterms:created>
    <skos:prefLabel xml:lang="en">High Renaissance</skos:prefLabel>
    <skos:prefLabel xml:lang="en">religious painting</skos:prefLabel>
    <dcterms:format xml:lang="en">oil</dcterms:format>
    <dcterms:format xml:lang="en">http://schema.org/Painting</dcterms:format>
    <dcterms:title xml:lang="en">Putti, detail from The Sistine Madonna - Raphael</dcterms:title>
    <edm:isShownAt xml:lang="en">http://www.wikipaintings.org/en/raphael/putti-detail-from-the-sistine-
madonna-1513</edm:isShownAt>
    <edm:isShownAt xml:lang="en">http://www.europeana.eu/portal/record/2048411/
item_62IOOX6Z44HCOATRU57SPOZSUGJWHXDV.html?utm_source=api&amp;utm_medium=api&amp;utm_campaign=ctPEdiu7D</
edm:isShownAt>
  </edm:ProvidedCHO>
```

Figure 8. Duplication detected

The two Europeana pages[17] [18] seem to refer the same CHO, and then probably we have again detected a duplicate (Fig. 9).

http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0130000/df_hauptkatalog_0130047.jpg

http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0192000/df_hauptkatalog_0192717.jpg

Figure 9 A likely case of duplicated resources

Additionally the title: "**Die Sixtinische Madonna | Raphael**", on the Europeana pages**,** is actually the title of the painting shown here:

http://www.europeana.eu/portal/record/2048411/item_2TGIA6VCCWBMKNL3UHZP7NXQ4PL6CRWI.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

17 http://www.europeana.eu/portal/record/2048413/item_755CC3YZQGXKNXX5VPZEADZ6VHTZZGCR.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

18 http://www.europeana.eu/portal/record/2048411/item_62IOOX6Z44HCOATRU57SPOZSUGJWHXDV.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

The matching WikiArt page reports instead, correctly, that the resource described is a *detail* of the painting[19] (Fig. 10):

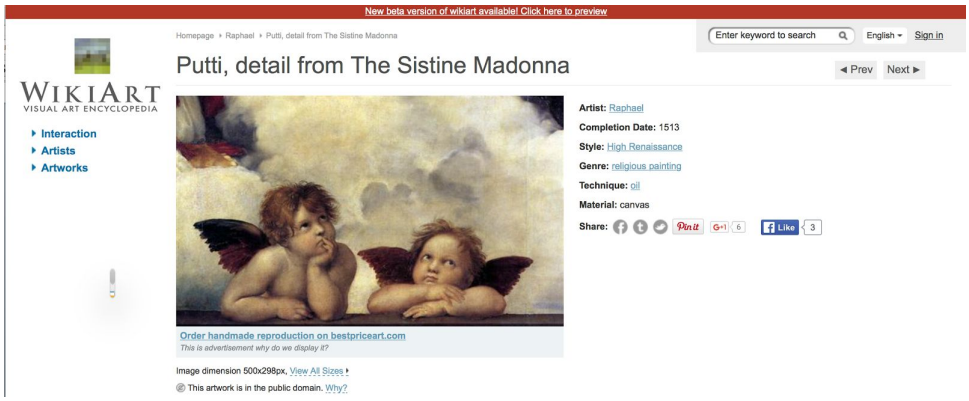


Figure 10 The WikiART recognized target resource

6.6.One last example: duplicate detected

The analysis of the file containing the output of the IREnrichment [8] allowed us to discover what appears to be an actual case of duplicate detection.

```
 <edm:ProvidedCHO>
    <dc:identifier>http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0753000/
df_hauptkatalog_0753797.jpg</dc:identifier>
    <edm:hasView>http://uploads0.wikiart.org/images/pablo-picasso/musicians-with-masks-1921.jpg!Blog.jpg</
edm:hasView>
    <dc:creator>Pablo Picasso</dc:creator>
    <dcterms:created>1921</dcterms:created>
    <skos:prefLabel>Cubism</skos:prefLabel>
    <skos:prefLabel>genre painting</skos:prefLabel>
    <dcterms:format>oil</dcterms:format>
    <dcterms:format>http://schema.org/Painting</dcterms:format>
    <dc:subject>allegories-and-symbols, music-and-dancing</dc:subject>
    <dcterms:title>Musicians with masks - Pablo Picasso</dcterms:title>
    <dcterms:title xml:dataset="europeana">Three Musicians I</dcterms:title>
    <edm:isShownAt xml:dataset="wikiart">http://www.wikipaintings.org/en/pablo-picasso/musicians-with-
masks-1921</edm:isShownAt>
    <edm:isShownAt xml:dataset="europeana">http://www.europeana.eu/portal/record/2048417/
item_FILYG5SJQBLIWVQURZB5K2WBHJDXVKYL.html?utm_source=api&amp;utm_medium=api&amp;utm_campaign=ctPEdiu7D</
edm:isShownAt>
  </edm:ProvidedCHO>
 <edm:ProvidedCHO>
    <dc:identifier>http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0753000/
df_hauptkatalog_0753797.jpg</dc:identifier>
    <edm:hasView>http://uploads0.wikiart.org/images/pablo-picasso/musicians-with-masks-1921.jpg!Blog.jpg</
edm:hasView>
    <dc:creator>Pablo Picasso</dc:creator>
    <dcterms:created>1921</dcterms:created>
    <skos:prefLabel>Cubism</skos:prefLabel>
    <skos:prefLabel>genre painting</skos:prefLabel>
    <dcterms:format>oil</dcterms:format>
    <dcterms:format>http://schema.org/Painting</dcterms:format>
    <dc:subject>allegories-and-symbols, music-and-dancing</dc:subject>
    <dcterms:title>Musicians with masks - Pablo Picasso</dcterms:title>
    <dcterms:title xml:dataset="europeana">Three Musicians I</dcterms:title>
    <edm:isShownAt xml:dataset="wikiart">http://www.wikipaintings.org/en/pablo-picasso/musicians-with-
masks-1921</edm:isShownAt>
    <edm:isShownAt xml:dataset="europeana">http://www.europeana.eu/portal/record/2048418/
item_FILYG5SJQBLIWVQURZB5K2WBHJDXVKYL.html?utm_source=api&amp;utm_medium=api&amp;utm_campaign=ctPEdiu7D</
edm:isShownAt>
  </edm:ProvidedCHO>
```

Figure 11: same resource images for different web pages

The same resource image:

[19] http://www.wikiart.org/en/raphael/putti-detail-from-the-sistine-madonna-1513

http://fotothek.slub-dresden.de/fotos/df/hauptkatalog/0753000/df_hauptkatalog_0753797.jpg

is used as property value in two separate Europeana web pages[20] & [21], apparently this means that two metadata records in Europeana actually describe the same resource.

7. Current status and future work

The goal of our activity so far has been to check if an approach based on Image Recognition (new for metadata enrichment) could help in improving the enrichment quality of Europeana metadata records and the level of duplicates detected within the Europeana dataset. To do this we have designed and implemented an application and performed tests on a small source database of 3,543 Europeana records while the target was nearly 30,000 WikiArt images, randomly downloaded.
The java code of the IREnrichment application is on the Europeana svn repository.

There are 165 matching metadata records [8], but in the current implementation it is not possible to understand how many images in the Europeana database we use actually refer to paintings and whether some painting images in the source database are among the WikiArt images not yet indexed (i.e. they could be recognized when the index is completed).
Even if limited, these results seem significant enough to plan a second test phase, aiming to understand if a more consistent Europeana dataset could be enriched with our approach.

Additionally both the Image Recognizer and the WikiArt target dataset in the current implementation have been chosen in order to minimize efforts for application setup: VeMIR is ready to use and WikiPaintings only contains paintings, each one described by a minimal but significant set of metadata. This has enabled us to focus on fixing bugs, tuning the interaction between IREnrichment and the recognizer, and defining a mapping strategy for the creation of enrichment records.
In the next phase of our activity we plan to investigate other solutions for these two components.

Essentially next activity steps will be:

1. Work with the Europeana team to build a significantly larger source database of Europeana images, if possible containing only images referring to paintings.
2. Complete the indexing of images on WikiArt, thereby almost doubling the size of the index, to complete tests.[22] In parallel we will start indexing images on Wikidata [9] in order to have a more reliable target dataset.
3. Try to involve people at VeMIR. So far we have used the recognizer as it is, and in our logs there are a small number of recognition errors that we cannot fix without cooperating with the VeMIR team. We need more details on how images are indexed, mainly to understand if something can be done to increase both precision in matching and performances and also to understand how we can use the trust level number returned in case of matching. Note that at the moment it is not clear if the VeMIR service could be used in production, which means we will have to start to investigate the use of other frameworks such as Pastec or Google Images.

[20] http://www.europeana.eu/portal/record/2048417/item_FILYG5SJQBLIWVQURZB5K2WBHJDXVKYL.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

[21] http://www.europeana.eu/portal/record/2048418/item_FILYG5SJQBLIWVQURZB5K2WBHJDXVKYL.html?utm_source=api&utm_medium=api&utm_campaign=ctPEdiu7D

[22] As noted, this is a proof of concept test. WikiArt copy right restrictions would have to be taken into account in the implementation phase.

4. Possibly build or adopt a tool to formally evaluate the result of our enrichment approach, following guidelines defined in [1].

We plan to complete the step 1 by the second week of February and step 2 by the end of February; contacts are in place for step 3 and its timeline depends from the feedback, but is not blocking. The results of the second test phase are expected by the end of March 2016.

After these steps will be completed we plan to carefully check the results involving experts and plan the next steps.

8. Annex 1: Use Europeana API to obtain source images

The images are retrieved from the Europeana dataset invoking Europeana API services.
The invoked call is the following:

http://www.europeana.eu/api/v2/search.json?wskey=key&query=who:*painternam*e&media=true&qf=IMAGE_SIZE:small&qf=IMAGE_SIZE:medium&cursor=%s&rows=n

where *painternam*e is the name of a painter. In this first phase we searched for paintings of the following artists:

Albrecht Durer
Angelica Kauffman
Annibale Carracci
Bernardo Bellotto
Canaletto
Caravaggio
Lovis Corinth
Cranach the Elder
Picasso
Raffaello

9. References

[1] http://pro.europeana.eu/files/Europeana_Professional/EuropeanaTech/EuropeanaTech_taskforces/Enrichment_Evaluation//FinalReport_EnrichmentEvaluation_102015.pdf
[2] https://docs.google.com/document/d/1JvjrWMTpMIH7WnuieNqcT0zpJAXUPo6x4uMBj1pEx0Y
[3] http://onto.dm2e.eu/edm#Agent
[4] Min-Yen Kan and Yee Fan Tan. Record Matching in Digital Library Metadata, https://www.comp.nus.edu.sg/~kanmy/papers/2008-cacm.pdf
[5] http://vemir.visualengines.it
[6] http://www.wikiart.org/en/About
[7] https://en.wikipedia.org/wiki/Microdata_(HTML)
[8] https://dl.dropboxusercontent.com/u/630356/WikiArtEnrichment.xml
[9] https://www.wikidata.org/wiki/Wikidata:Main_Page

Report on task 4.3.2. - Discovering Semantic Similarity

1. Introduction

In this task the Open University investigated different models for connecting content uploaded to the Europeana Cloud platform and providing a service for discovering semantic similarity between content of different providers. Three different methods were examined and evaluated in terms of accuracy and performance. A selected method is exposed as a live service using a simple application programmable interface (API)

2. Motivation

Europeana Cloud content is the aggregation of content uploaded by multiple providers. This consists of various media formats, descriptive metadata, and is organised in various distinct collections (datasets).OU wanted to be able to discover semantically similar records (given a reference record). This can be useful to detect duplicate documents (deposited by different providers) or simply to provide a content-based recommendation to the viewer of Europeana content. The method should be capable to detect same objects with minor alterations, or provide a list of objects with similar full text.

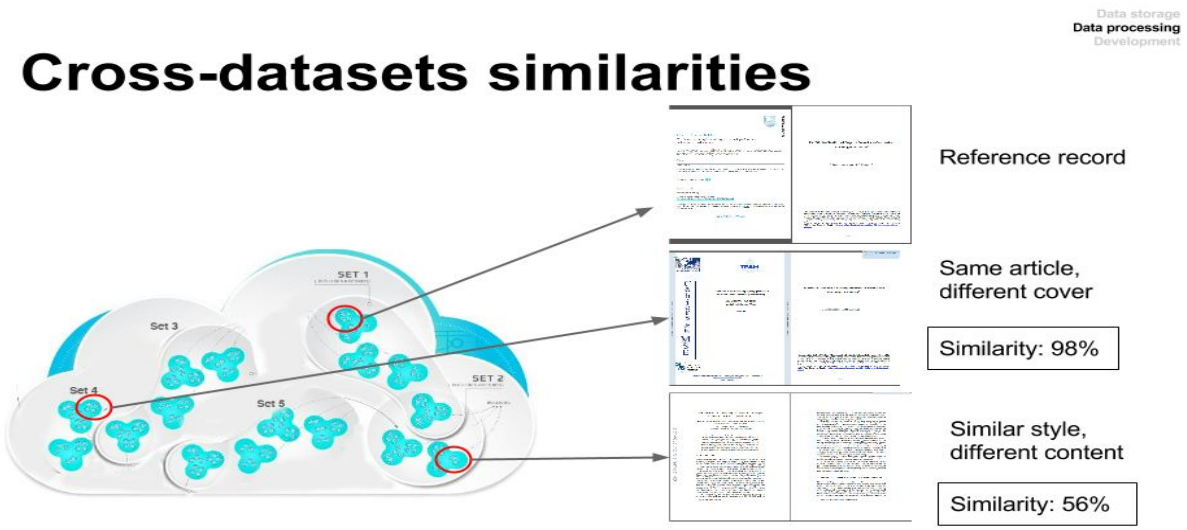


Figure 1 - Similar records discovery in Europeana Cloud

3. Method used

We can interpret content uploaded to Europeana Cloud as a large document database. In order to be able to discover textual similarities we need to represent this content in a vector based representation. Having represented documents as a set of term vectors we shall be able to apply content-based queries on top of it. Examples of such queries can be a simple term vector comparison (naive or with simple heuristics applied, such as tf-idf) [1].

In the early stages of this task, Locality Sensitive Hashing (LSH) techniques were considered (Simhash [2] and Minhash [3]) as robust and highly-scalable methods to detect near-duplicates. As a matter of fact, a variation of MinHash was used in the past by Europeana to hierarchically structure large aggregations of metadata [4]. However, this approach was abandoned as it was capable to detect only very similar (almost identical) documents. LSH techniques are capable to accurately detect differences in small portions of documents, essential for applications like crawling, in which a small piece of added advertisement in a page should not affect web search. However, to support the needs in working with a broader corpus such as that in Europeana Cloud, we chose the more flexible and sound method of term vector comparison using the tf-idf heuristic [5]. Term vector comparison with tf-idf comprises of a simple model to assist ranked queries and has been battle proven in the industry for years. It well suits the case of Europeana Cloud, in terms of the expected enormous scale of the dataset and the flexibility to further extend or fine-tune the approach in the future.

To assist in the complex task of indexing documents we make use of the Apache Lucene [6] library: an open source information retrieval library that in turn is the basis for the most popular enterprise search platforms: Apache Solr[23] and ElasticSearch[24]. For the purposes of this task we created appropriate identical interfaces for both engines, allowing flexibility to Europeana Cloud clients to select the solution that best fits their purposes.

4. Schema used

We expected a broad, heterogeneous corpus of content in Europeana Cloud. As we wanted to capture all metadata formats expected to be uploaded in Europeana Cloud, we could not specify a narrow and specific schema. We settled for using a simple minimalistic schema with just one field to hold the concatenation of text from different files of a record.

Concatenation follows a simple strategy to merge textual content coming from various representations of a record (textual content extracted from binary files or stripped text from metadata content). It is able to handle multiple representations, multiple versions per representation, it even handles the extreme cases of having multiple files per representation version. Stripped content is not ordered in any particular order (as it will later be organised in an inverted index, order is not important in this case) but special care is taken not to include twice the same file.

Setting up the schema used for both solutions (Solr/Elasticsearch) is fully described in the setup script hosted in
https://github.com/europeana/Europeana-Cloud/tree/develop/utils/scripts/similarity_search
It is important to explicitly state that the single-field used in the above schema is stored as a term vector with positions offsets (and not simply as an analyzed string which is the default operation).

5. Indexing process

Indexing Europeana Cloud ingested content is done in a controlled manner using the Data Processing Service (DPS). The process of reading from Europeana Cloud, transforming a document into indexable document and pushing to the external index is executed inside a DPS plugin.

The plugin is responsible for reading files from the Metadata and Content Service (MCS), applying appropriate stripping over them (in the case of pdf files for example), transforming content in an

[23] Apache Solr - http://lucene.apache.org/solr/
[24] ElasticSearch - https://www.elastic.co/

indexable document, potentially merging this to other files of the same record and finally pushing it to an external endpoint. The current plugin can handle only textual content. This includes pdf (using Apache Tika to extract text) and txt files. Metadata files are stripped (of xml tags) and concatenated to the rest of content, therefore indexed as part of the whole content. In some cases only a metadata representation exists, so the indexed content is exclusively comprised of stripped metadata xml record.

Europeana Cloud DPS plugins are implemented as Apache Storm[25] topologies. Storm is a distributed event-based computation framework that encloses its applications in topologies of data-processing nodes called spouts and bolts. Those are connected in a directed acyclic graph to represent the pipeline of data propagation. The topology used for the purposes of extracting and indexing is shown in figure 2:

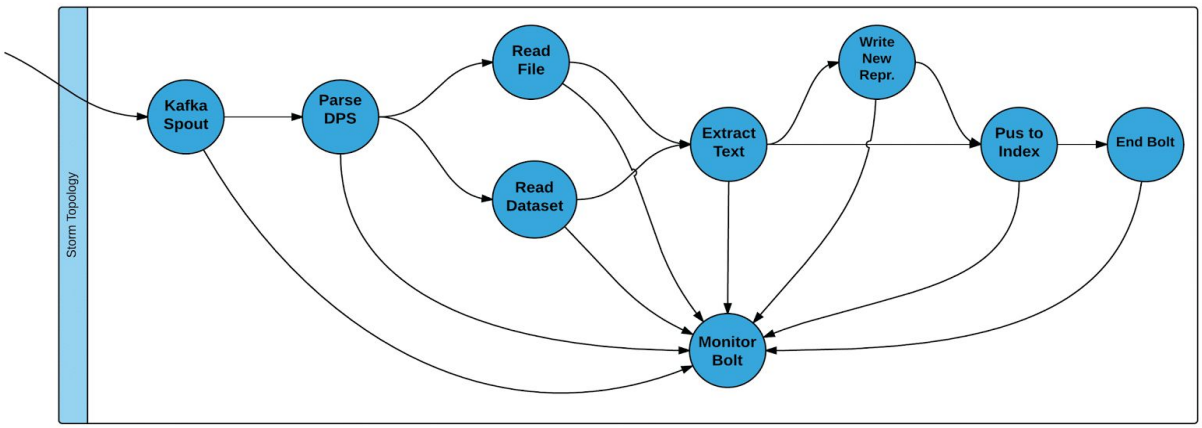


Figure 2 - DPS extract and index topology

From a top-level view and without going into extensive detail, the above topology starts (as all DPS topologies) with a Kafka spout connected to a Parse DPS task bolt. The spout is responsible for reading the incoming DPS task from a Kafka topic (the Notification Service - NS) and passing it to the next bolt which in turns parses data included in the DPS task and decides the mode in which the topology shall operate. That is identifying the input data (which can be a list of file URIs or/and a list of dataset URIs) and parsing all the parameters passed by the user. It then propagates the list of input resources to either the ReadFile or the ReadDataset bolts which are responsible for reading from the MCS the content of these resources and pass each resource's content to the next bolt "ExtractText" . The ExtractText bolt is responsible for stripping textual content from the binary input stream and transforming it to something compatible with the indexer's schema document. If the task was invoked with the parameter STORE_REPRESENTATION=true, this transformed document shall be stored as a new representation (named INDEXABLE_REPRESENTATION) in the same record of the file. Finally the index-friendly document is posted to the external index and the file processing comes to an end. Notice that each node in the topology is connected to a monitor bolt which receives notifications of the success of the operation in each node, for each step of each file. This is persistently stored in a notification database which can be queried at any time to retrieve progress status of a DPS task execution.

[25] Apache Storm - http://storm.apache.org/

5.1. Indexing workflow

To invoke the execution of the above plugin we need to submit a new DPS task. From unix command line:

```
$> curl -u admin:admin -XPOST
http://cloud.europeana.eu/topologies/extract_and_index/tasks -d
'
{
     "inputData":{

"FILE_URIS":["https://cloud.europeana.eu/api/records/6OMRCVUE5C
WPVV2KRZ5I2PODPTEPY67VCJDDROQ5HMIOT2HZZV5Q/representations/oai-
pmh/versions/bc14a2c0-bdfe-11e5-9f0d-fa163e2dd531/files/07f225c
c-ff58-44b6-9e79-4ee2494049e4"  ],
          "DATASET_URIS":[]
     },
     "parameters":{
          "INDEXER" : "ElasticSearch",
          "INDEXER_ENDPOINT" : "http://localhost:9200/",
          "STORE_REPRESENTATION":false
     }
}'

$> {"taskId":"12345", "time":123456789010}
```

Listing 3 - DPS Task invocation

In the above example we denote that we want to process and index the file in the URI : https://cloud.europeana.eu/api/records/6OMRCVUE5CWPVV2KRZ5I2PODPTEPY67VCJDDROQ5HMIOT2HZZV5Q/representations/oai-pmh/versions/bc14a2c0-bdfe-11e5-9f0d-fa163e2dd531/files/07f225cc-ff58-44b6-9e79-4ee2494049e4 in an ElasticSearch instance running in `localhost:9200`.

We get as a response the id of the created task and the time when it was created as a UNIX timestamp.

The Indexer parameter expects values of Solr or ElasticSearch. Multiple files are allowed in the `FILE_URIS` or `DATASET_URIS` field.

5.2. Monitoring progress

To monitor progress of the execution of the above DPS task we make use again of the DPS REST API; so from a unix shell:

```
$> curl -u <user>:<password> -XGET
http://cloud.europeana.eu/topologies/extract_and_index/tasks/12
```

```
345/notifications

$> {
     "taskId":"12345",
     "processed_tuples":[
          {

"resource_URI":"https://cloud.europeana.eu/api/records/6OMRCVUE
5CWPVV2KRZ5I2PODPTEPY67VCJDDROQ5HMIOT2HZZV5Q/representations/oa
i-pmh/versions/bc14a2c0-bdfe-11e5-9f0d-fa163e2dd531/files/07f22
5cc-ff58-44b6-9e79-4ee2494049e4",
               "time":123456789000,
               "state":"SUCCESS",
               "information":"",
               "additional_information":""
          },
          {

"resource_URI":"https://cloud.europeana.eu/api/records/2ZBXXCMW
34LN4FO75FDAOJ5FPBENQPPLRFZ4EFDMFLTJNHYKAAEA/representations/oa
i-pmh/versions/b8d07440-bdfe-11e5-9f0d-fa163e2dd531/files/a9c1c
b08-ed4e-40e3-abf7-4c83d15e8955",
               "time":123456789012,
               "state":"ERROR",
               "information":"Error while creating
representation in MCS",
               "additional_information":"Exception in thread
\"main\" javax.ws.rs.ProcessingException:
java.net.ConnectException: Connection timed out
     at
org.glassfish.jersey.client.HttpUrlConnector.apply(HttpUrlConne
ctor.java:205)
     at
org.glassfish.jersey.client.ClientRuntime.invoke(ClientRuntime.
java:217)
     at
org.glassfish.jersey.client.JerseyInvocation$1.call(JerseyInvoc
ation.java:655)

                              • • •


     at
org.glassfish.jersey.client.JerseyInvocation$Builder.post(Jerse
yInvocation.java:321)
     at
eu.europeana.cloud.mcs.driver.RecordServiceClient.createReprese
ntation(RecordServiceClient.java:234)
```

```
Caused by: java.net.ConnectException: Connection timed out
     at java.net.PlainSocketImpl.socketConnect(Native Method)
     at
java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketI
mpl.java:350)

                                • • •
     at
org.glassfish.jersey.client.ClientRequest.writeEntity(ClientReq
uest.java:433)
     at
org.glassfish.jersey.client.HttpUrlConnector._apply(HttpUrlConn
ector.java:290)
     at
org.glassfish.jersey.client.HttpUrlConnector.apply(HttpUrlConne
ctor.java:203)
     ... 16 more"
          },
          {
          ...
          }.
          ...
     ]
     }
```

Listing 4 - Progress monitoring example

In the above example we can monitor for the taskid 12345 that 2 resources were processed, one with success and one with an error caused by miscommunication with the MCS. The operator can observe the Storm java exception and proceed to debug the error (or notify the DPS admin team of this exception).

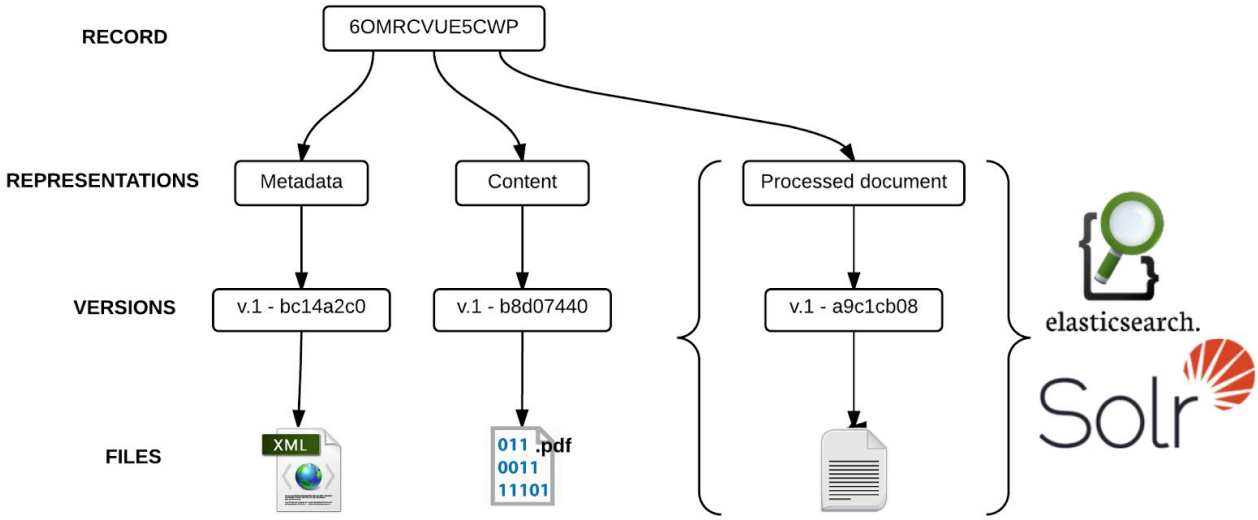


Figure 3 - Example record

6. Similarity service API specification

The similarity service is exposed to the end-user by means of a simple minimal interface. It makes use of Lucene's more_like_this query which is a convenience function for a Boolean weighted (based on the tf-idf heuristic) query.

```
/**
     * Retrieve documents with similar content.
     * @param documentId index of reference document
     * @param fields array of fields to search
     * @param maxQueryTerms maximum number of query terms that will be selected
     * @param minTermFreq minimum term frequency below which the terms will be
ignored from the input document
     * @param minDocFreq minimum document frequency below which the terms will
be ignored from the input document
     * @param maxDocFreq maximum document frequency above which the terms will
be ignored from the input document
     * @param minWordLength minimum word length frequency below which the terms
will be ignored
     * @param maxWordLength maximum word length frequency above which the terms
will be ignored
     * @param size number of results on one page
     * @param timeout tells how long it should keep the search context alive.
(ms)
     * @param includeItself specifies whether the input document should also be
included in the search result
     * @return instance of SearchResult
     * @throws IndexerException
     */
    public SearchResult getMoreLikeThis(String documentId, String[] fields, int
```

```
maxQueryTerms, int minTermFreq,
            int minDocFreq, int maxDocFreq, int minWordLength, int
maxWordLength,
            int size, int timeout, Boolean includeItself) throws
IndexerException;
```

Listing 5 - Similarity interface

The function effectively requires only one parameter: the cloud_id of the reference document. The rest parameters (maxQueryTerms, minTermFreq, minDocFreq, maxDocFreq, minWordLength, maxWordLength) are simply parameters to further optimise and fine-tune search results. Detailed documentation can be found here:
https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-mlt-query.html

Finally we added 3 complementary parameters (optional): size (the number of search results to return), timeout (how long to keep the request alive) and includeItself (whether we want to include the reference record in the search results).

7. Similarity service REST API

The above similarity interface is further exposed using a REST API which is described below (using Swagger[26] specifications)

```
swagger: '2.0'

info:
  version: "0.5.Snapshot"
  title: europeana cloud similarity service REST API

# Describe your paths here
paths:
  /similar/{cloud_id}:
    get:
      description: |
        Gets a list of content similar records to the record provided
      # This is array of GET operation parameters:
      parameters:
      - name: cloud_id
        in: path
        description: The cloud id of the input record
        required: true
        type: string
      responses:
        200:
          description: Successful response
          schema:
            type: object
```

[26] Swagger (http://swagger.io/) is a framework to describe RESTful web services.

```
            $ref: "#/definitions/SearchResponse"
        400:
          description: Invalid input data
        404:
          description: No document with input cloud_id found

definitions:
  SearchResponse:
    type: object
    properties:
      totalHits:
        type: integer
      maxScore:
        type: number
      tookTime:
        type: integer
      hits:
        type: array
        items:
          $ref: '#/definitions/hit'
  hit:
    type: object
    properties:
      ecloud_id:
        type: string
      lucas:
        type: number
      data:
        type: string
```

Listing 6 - Similarity REST service Swagger specification

An example of invoking the similarity service is shown in the listing below:

```
$> curl -u admin:admin -XPOST
'http://<ecloud_host>/similarity/6OMRCVUE5CWPVV2KRZ5I2PODPTEPY6
7VCJDDROQ5HMIOT2HZZV5Q' -d '
{
     "fields":["raw_text"],
     "maxQueryTerms":25,
     "minTermFreq":2,
     "minDocFreq":5,
     "maxDocFreq":0,
     "minWordLength":0,
     "maxWordLength":0,
     "size":10
     "timeout":1000
     "includeItself":false
}'
```

```
$> {
     "totalHits" :11,
     "maxScore": 1,
     "tookTime": 123
     "hits":[
          {

"ecloud_id":"7JC6K66FSPPQCDTGBFELTSTSSPDBLEN6UUS7WSHTA4VYA4J5L7
LA",
               "score":0.98,
               "data":"{"raw_text":"Social Research of the
University La Sapienza \npublished a report entitled
FuoriLuogo. L'immigrazione e i media italiani in 2004 as part
of the national \ninitiative 'Etnequal Social Communication'
organized by the Minstry of Employment and Social
               ...
               Italian media can be \nfound the 2009 report
Ricerca nazionale su immigrazione e asilo nei media italiani
(National research \nreport on  Security \n \nCurrently Italian
society is facing a future headed by economic and welfare
uncertainty, but the \nattention of Italians"}"
          },
          {

"ecloud_id":"4ZDUCIBMNO63CGRI4VUH2NDORYJRLOV5M74JFR2AZ6S6CPBJUR
QQ",
               "score":0.85,
               "data":"{"raw_text":"Challenging social
anti-immigration perceptions in Lampedusa. Masters thesis,
School of Advanced Study.<\/dc:identifier><dc:relation>\r\n
http:\/\/sas-space.sas.ac.uk\/4775\/<\/dc:relation><\/oai_dc:dc
><\/metadata><\/record>text\r\nhe country is in the process of
transition towards a cosmopolitan and multi-ethnic society, but
is\r\nrevealing growing tensions and
               ...
               crisis that originally was not one. It will aim
to highlight the essential traits that have stood\r\nout during
the study phase.\r\nThe second section will then turn to the
data collected whilst interviewing the local population
of\r\nLampedusa in order to understand first whether the
picture of the island that has been created by\r\nItalian news
is a truthful reflection of the reality or just a factious and
false portrayal."}"
          },
          {...},
          ...
     ]
```

```
}
```

Listing 7 - Similarity REST example

8. Similarity discovery example

In listing 8 below we present a solid example that discovers semantic similarity between two Europeana Cloud records . The records in this specific example comprise of an xml representation and were uploaded to the Europeana platform from different collections (similarity score returned by similarity service : **0.985746**).

We carried the same experiment in 1,196,155 (a small subset of) records uploaded in Europeana Cloud. Setting up a high threshold of the similarity score (>0.985) we managed to detect in this subset 3,587 **near-duplicate** records (i.e. 0.0029998 % of records are duplicates).

Clearly, this demonstrates one of the use cases of this similarity service: to discover near-duplicates (or near identical records) deposited in Europeana Cloud. The similarity score expressed in the results of this service offers different gradations of how this service can be used, so that it may, for example, provide recommendations on related (but not identical) content that exists in the Europeana Cloud platform.

In addition, the end-user can select the sensitivity of the service not using defaults, but having the option to pass term selection parameters (e.g. max_query_terms, min_term_freq, etc.). In the body of the above REST request, he can manipulate the behaviour of the ranking formula, therefore the behaviour of the similarity service.

ecloud_id	Q7KM3L54HKIDS4DU3ATOV4PXAVW5NANHNQCKL6OBFDKKNARA7NOA	676EPBQS77U2TIFB457PRXJTTTXCERPKAPJWDA7SXUCSJFAPC7BQ
local id	29018906	99785
ecloud OAI-PMH representation file URI	https://cloud.europeana.eu/api/records/Q7KM3L54HKIDS4DU3ATOV4PXAVW5NANHNQCKL6OBFDKKNARA7NOA/representations/oai-pmh/versions/54a2ddc0-c383-11e5-a91e-fa163e60dd72/files/e130f13f-20ec-463f-9353-e393d7300817	https://cloud.europeana.eu/api/records/676EPBQS77U2TIFB457PRXJTTTXCERPKAPJWDA7SXUCSJFAPC7BQ/representations/oai-pmh/versions/2e88bdf0-c68d-11e5-888d-fa163e60dd72/files/db932106-ea46-435b-af4a-0a0acd9ce255
OAI-PMH representation content	<record><header><identifier> oai:eprints.leedsbeckett.ac.uk:792</identifier><datestamp> 2015-01-08T01:30:15Z</datestamp><setSpec> 74797065733D636F6E666572656E63655F6974656D</setSpec></head	<record><header><identifier> oai:eresearch.qmu.ac.uk:2227</identifier><datestamp> 2014-03-19T12:58:49Z</datestamp><setSpec> 7374617475733D707562</setSpec><setSpec>

	er><metadata><oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd" ><dc:relation> http://eprints.leedsbeckett.ac.uk/792/</dc:relation><dc:title> Gender and Health: The case for gender-sensitive health policy and health care delivery</dc:title><dc:creator> O'Brien, O</dc:creator><dc:creator> White, AK</dc:creator><dc:description> There is growing national and international recognition that gender is an important indicator of health differences. The United Kingdom is in danger of falling behind many other countries that are beginning to recognise the crucial importance of gender to the development of effective health policy and practice. This briefing paper sets out some of the reasons why those involved in the gender and health partnership (GAHP) believe that it is time to place gender at the heart of the equalities agenda in health, and why it is important to ensure gender sensitivity in health policy, medical research and services. This briefing paper refers to the United Kingdom; however, Scotland is in the process of producing guidelines specifically on mainstreaming gender in health policy and service delivery.</dc:description><dc:date> 2003-11-14</dc:date><dc:type> Conference or Workshop Item</dc:type><dc:type> PeerReviewed</dc:type><dc:format>	74797065733D636F6E666572656E63655F6974656D</setSpec></header><metadata><oai_dc:dc xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd" ><dc:relation> http://eresearch.qmu.ac.uk/2227/</dc:relation><dc:title> Gender and Health: The case for gender-sensitive health policy and health care delivery</dc:title><dc:creator> O'Brien, Oonagh</dc:creator><dc:creator> White, Alan</dc:creator><dc:description> There is growing national and international recognition that gender is an important indicator of health differences. The United Kingdom is in danger of falling behind many other countries that are beginning to recognise the crucial importance of gender to the development of effective health policy and practice. This briefing paper sets out some of the reasons why those involved in the gender and health partnership (GAHP) believe that it is time to place gender at the heart of the equalities agenda in health, and why it is important to ensure gender sensitivity in health policy, medical research and services. This briefing paper refers to the United Kingdom; however, Scotland is in the process of producing guidelines specifically on

	text</dc:format><dc:language> en</dc:language><dc:identifier> http://eprints.leedsbeckett.ac.uk/792/1/Gender%20and%20Health%20The%20case%20for%20gender-sensitive%20health%20policy%20and%20health%20care%20delivery.pdf</dc:identifier><dc:identifier> O'Brien, O and White, AK (2003) Gender and Health: The case for gender-sensitive health policy and health care delivery. In: First UK Gender and Health Summit Promoting Health Equality for Men and Women, The King's Fund, London. </dc:identifier></oai_dc:dc></metadata></record>	mainstreaming gender in health policy and service delivery.</dc:description><dc:date> 2003</dc:date><dc:type> Conference or Workshop Item</dc:type><dc:type> PeerReviewed</dc:type><dc:format> application/pdf</dc:format><dc:language> en</dc:language><dc:rights> </dc:rights><dc:identifier> http://eresearch.qmu.ac.uk/2227/1/eResearch_2227.pdf</dc:identifier><dc:identifier> O'Brien, Oonagh and White, Alan (2003) Gender and Health: The case for gender-sensitive health policy and health care delivery. In: First UK Gender and Health Summit Promoting Health Equality for Men and Women, 14th November 2003, King's Fund, London. </dc:identifier><dc:relation> http://www.emhf.org/resource_images/GaHP_Briefing_Paper.pdf</dc:relation></oai_dc:dc></metadata></record>

Listing 8 – Real similar records example

9. Code hosting

The current deployment model implies a remote instance of Solr or Elasticsearch to be managed by the concerned party. Both engines are supported to avoid lock-in in a specific solution (both solutions though similar in functionality offer different interfaces of interaction) and to facilitate future decision on the solution chosen. The workflow dictates that he then selects the collections of interest from the content residing in Europeana Cloud and then uses the DPS plugin to create an index which can be searched using the similarity REST API.

Though not currently a component of the infrastructure services described in WP2, the service described in this section offers the option to be deployed as a vital component of ECloud platform, with minimal effort, if this is decided as part of Europeana Cloud services extension in the future.

The implementation of the above solutions is hosted among other Europeana Cloud services in https://github.com/europeana/Europeana-Cloud . Among others, you can find:

- The DPS topology (for extracting and indexing)
- A full example demonstrating the workflow carried out

- The REST web application that implements the similarity service
- The example schemas for Solr and Elasticsearch, and documentation of how to setup either service.

10. Conclusion

As part of this task we developed a search solution for Europeana Cloud content that can be used to discover (semantic) similarities between different records (even those uploaded by different providers). Scope of this task was to provide the mechanism to index uploaded ECloud content (which by nature is highly heterogeneous; comes from various providers and in various formats).
The solution proposed utilises celebrated industry solutions (Solr/ElasticSearch) to increase compatibility and ease of integration with external to eCloud platforms. The service is exposed through a REST API following the service-oriented mentality of all the rest of ECLoud components, avoiding a monolithic solution.

11. References

[1] Zobel, Justin, and Alistair Moffat. "Exploring the similarity space." *ACM SIGIR Forum*. Vol. 32. No. 1. ACM, 1998.

[2] Sadowski, Caitlin, and Greg Levin. "Simhash: Hash-based similarity detection." (2007): 13-22.

[3] Chum, Ondrej, James Philbin, and Andrew Zisserman. "Near Duplicate Image Detection: min-Hash and tf-idf Weighting." BMVC. Vol. 810. 2008.

[4] Wang, Shenghui, et al. "Hierarchical structuring of cultural heritage objects within large aggregations." *Research and Advanced Technology for Digital Libraries*. Springer Berlin Heidelberg, 2013. 247-259.

[5] McCandless, Michael, Erik Hatcher, and Otis Gospodnetic. *Lucene in Action: Covers Apache Lucene 3.0*. Manning Publications Co., 2010.

Report on task 4.3.3 - Evaluation of Europeana Cloud for use with the Heritage of the Printed Book database (HPB)

1. CERL and the Heritage of the Printed Book database

The Consortium of European Research Libraries (CERL) has been founded in 1994 as a membership organisation of currently over 250 libraries each with a keen focus on cultural heritage in the form of early printed books and manuscripts. CERL's aim is to develop and provide tools and services to both the library community and researchers in the field of the history of the book. At the time of its foundation, CERL's first objective was to bring together all information on books printed in Europe during the hand-press period (c.1450 - c.1830) in the Heritage of the Printed Book Database (then the Hand Press Book Database (HPB)): CERL was one of the pioneers of using web-based interfaces to make integrated bibliographical sources available to researchers. All of CERL's products and services, which include the HPB the CERL Thesaurus, the Material Evidence in Incunabula database and the CERL Portal for Manuscripts are created as a joint effort of special collections libraries across Europe and North-America.

The Heritage of the Printed Book is a steadily growing aggregation of catalogue records from major European and North American research libraries covering items of European printing of the hand-press period integrated into one database. It currently holds a little more than 6 million records. New files are added to the HPB Database on a regular basis. The majority of these files consist of high-level bibliographical records created by book-in-hand cataloguing. Some files contain records created as a result of retro-conversion projects, and these are gradually being replaced by enhanced records. It may be accessed for information retrieval and downloading by CERL member institutions, their staff and users.

The Description of Work suggested that this task would explore whether editing of data in the Cloud would be possible, either manually or via batch-editing. It became apparent that this is not an appropriate use case for the Cloud infrastructure. Data correction typically takes place on the side of the institutions, and Europeana benefits from these improvements after the data is (re-) uploaded in the Cloud. This echoes the current HPB workflow at CERL and it is for this reason that in this task we wanted to explore whether the Europeana Cloud could be a viable alternative to our current hosting for the HPB (the Gemeinsamer Bibliotheksverbund in Göttingen, Germany).[27]

2. HPB specifics and workflow

The HPB accepts records from many different suppliers (libraries and cultural heritage institutions). All of these providers deliver files in a format that they can handle. The preferred delivery format used to be UNIMARC but has changed to MARC21 during the last years. Data providers may also deliver files in other formats.

All delivered files are converted to Pica+, the bibliographic format used by HPB. The original formats are currently not displayed in HPB (a generated reverse MARC21 mapping can be displayed

[27] https://www.gbv.de/?set_language=en

but this is not identical to the version in the delivered file). Even though HPB links to the source system record wherever possible, hosting the original record in Europeana Cloud along with the converted record would be interesting. The record displayed in the source system may change over time. Displaying how the versions of the source record evolve would improve confidence and provenance tracking, and facilitate direct user access to the records.

All suppliers to the HPB send updates at different intervals that contain new records, modified records and delete records that were present in a previous version of the dataset. The HPB converts the update file to Pica+ and replaces the previous file in HPB. Version control is currently achieved via archiving the different versions of the delivered file and each conversion on our file system. Version control on a record level within the database would be interesting. Users of HPB would be able to access past versions. The database might even implement a notification facility for new versions of a record. Direct comparisons between record versions would be possible.

3. Requirements

For using the cloud infrastructure with CERL's HPB data, the following requirements must be met:

a) Records belong to a file that represents the current collection of one of our suppliers.
b) Records exist in different data formats. One is the data delivery format (MARC21, UNIMARC, others), another is the converted data format (PICA+) in order to provide a shared display and retrieval interface. The existing data formats vary among different providers and may even change for a single provider over time.
c) Records exist in different versions. Every time a supplier sends a new update, the individual records might have been modified or deleted, or kept the same. We need to keep a convenient overview of the current (and each previous) version not only within one format but among different formats. This means:
    - We need to easily select all records that belong to one file delivery.
    - We need to select all records of a certain delivery in one data format and easily switch to the same delivery in another format.
    - We need to have the order of available versions of a file be listed on the cloud storage side as opposed to maintaining local mapping tables ourselves.
    - Ideally, we need to influence the order of versions, i.e. the order we would prefer might differ from the upload order.
    - We need to list all records of one provider, hierarchically grouped by A. version and B. format. We need to navigate this data structure easily.
    - We need to access all records of the latest version within one format as well as within other existing formats with a good performance for display and indexing.
d) Local identifiers are present in each record but in different locations inside the record structure depending on the format. We need to access the local record identifier automatically in these different data structures for use with the cloud.
e) We need to index all versions and all representations of all records grouped by provider, representation, file and version as well as easily access the specific file from index search result.
f) The HPB gives access to users with a valid account only. We need to ensure that rights management for accessing the data in the cloud can be handled accordingly.
g) Ideally, the cloud would allow for OAI (or Z39.50) harvesting of the data with appropriate user rights management.

4. Approaches to using the Europeana Cloud data model for HPB

4.1. General

The HPB can make use of the record-based approach of the Europeana Cloud model. One record in the cloud would consist of one bibliographic record. On the level of files that are uploaded and stored in the cloud, each file is a bibliographical file that contains one record in a bibliographic format (MARC21, Pica+, UNIMARC, etc.).

Different versions of this file can then be uploaded using the same Cloud Identifier. Each record would correspond to exactly one uploaded file.

Each existing bibliographic data format (MARC21, Pica+, UNIMARC, etc.) would be modelled as one representation. As a result, records that belong to the same collection would be stored in different representations using the same Cloud Identifier. This allows for independent versioning of records in different bibliographic data formats.

4.2. Files

For the HPB, in this context, "file" designates the delivery entity. Suppliers deliver a set of all the records that they contribute in the current version and in one format. This entity is different from the files within Europeana Cloud where the file level designates the actual uploaded entity, e.g. a text file of one record.

The data delivery as received from a HPB data provider (i.e. a CERL member library) would need to be modelled in Europeana Cloud as it is an important grouping criterion for HPB. This could be achieved by using datasets. There would be many datasets, each of which would represent a complete file as received from an HPB data provider. One dataset "file" (in the HPB sense) would be linked to from all the versions of records that were present in this file.

The delivered file and the converted file could link to the same dataset because the differentiation of formats can be made based on the representation. The individual file could then be filtered via dataset *and* representation.

4.3. Suppliers

Suppliers could be modelled as data providers. This would require the HPB to work with different independent data providers in Europeana Cloud as there is no way to link data providers or structure them hierarchically. The ability of a customer to keep a separate set of fundamental data such as providers list, have separate customizations, and have different views of its data from other customers, is referred to as Multi-tenancy. Therefore the HPB case could drive the analysis of Multi-tenancy system requirements for Europeana Cloud.

4.4. Updates

New records would be assigned to the dataset that corresponds to the update file ("file" in the HPB sense). Deleted records would not hold a version that points to this new dataset.

There are different options for handling records which are unchanged. The unchanged records could be uploaded again and this version would point to the new dataset. Even though this option may simplify the workflow, it is not desirable as it would result in redundant data. To avoid duplicate data storage, the unchanged records could be represented by the already existing versions from the previous update and could be assigned to the new dataset. The existing versions would then point to more than one dataset that represents an HPB-file (collection), i.e. be part of more than one file/collection. This second option however will keep all metadata unchanged. If the existing record is required to display the current date (last modification) this option cannot be used.

5. Evaluation

The DCG team used the Europeana Cloud API to explore which HPB requirements are currently met. (Referring to the requirements in section 2.)

a) The concept of a data delivery that contains all records provided by one provider in one format (representing the then current version of all these records) can be modelled as a combination of dataset and representation.

b) The same content in different bibliographic formats can be represented by different representations.

c) On the level of the individual bibliographic record, versioning is handled automatically by the cloud, i.e. uploading a new record creates a new version of this record. The versioning of a bibliographic file (containing many records) can be modelled by assigning all the records of the current data delivery to a new dataset.

Although it is possible to organize the data in the cloud that belong to the HPB similarly to the way it is done in the HPB itself, it is not very convenient: Within the cloud structure, records are hosted individually. Any connection between them is represented by the assignment of those records to one or more datasets. This means that requesting the same bibliographic record in a different format would require the request URI to be changed in order to retrieve the actual file name of a different representation. Representation *and* file name would change in the requested URI. An API that would allow for changing only the representation part of the URI in order to get the same version of a record in a different bibliographic format would be more accommodating to the needs of the HPB.

Example request of a MARC21 record:
https://195.216.97.95/api/records/JP77YTUECZLZKHR55T6LDQXQFEYHDNFBB54FL5QYNSET3QDC6JGQ/representations/MARC21/versions/081b5d90-babb-11e5-9e07-fa163e60dd72/files/a2e074fb-68b6-459d-9167-9deb0770b7aa

Example request of the same record in UNIMARC format:
https://195.216.97.95/api/records/JP77YTUECZLZKHR55T6LDQXQFEYHDNFBB54FL5QYNSET3QDC6JGQ/representations/UNIMARC/versions/363277c0-bacc-11e5-9e07-fa163e60dd72/files/7f07cfa1-a3ff-491a-821d-2e3df991c716

The examples show that only the Cloud Identifier remains the same (JP77Y…), while the representation (MARC21 / UNIMARC) as well as version ID and file ID change.

We did not find a way to change the metadata of versions in retrospect.

d) Local identifiers can be mapped to Cloud Identifiers. The local identifiers will, however, need to be extracted from the data itself for the sake of this mapping. As the cloud does not allow accessing the data in the bibliographic record it cannot be automatically extracted from the record via, e.g., format-specific rules and therefore needs to be done on the client side.

   The Cloud Identifier would represent all versions of a record in all representations. Representation, version ID and file ID need to be added in order to retrieve the record. Directing the retrieval to a certain representation needs to be implemented on the client side.

e) Possible workflows for indexing have not yet been evaluated.

f) Rights management in Europeana Cloud is currently implemented on the level of uploaded files. Users can be granted access to a particular file (i.e. specific version of a bibliographic record in a certain format), but not to a record including all its representations and in all their versions.

   Extended permission management is planned.

g) Since Europeana Cloud offers an HTTP API, harvesting records would be possible. Rights management for access via HTTP would be the same as general rights management. This does, however, not represent the level of harvesting common in the library community because the bibliographic records are stored as atomic file entities in the cloud and cannot be searched or filtered on the level of *content* of the record on the API level. This is a downside as harvesting records is a common way of exchanging metadata between cultural heritage institutions.

5.1. Further observations

These are issues not listed in section 2 above.

h) The SSL certificate of the API is not valid and causes errors in all clients. Evaluation and use of the API will be more convenient once the SSL certificates provided will be valid which will be the case shortly.
i) Different Post requests sent to the database require data to be sent in various formats. Sometimes data is sent as JSON (create a new provider), sometimes as URL-encoded form data (create a new representation), sometimes as multipart form data (upload an image). Some of the form data is sent via the URI (data-provider, version ID, cloud ID). It would be desirable to be able to, e.g., always send data as JSON or always as URL-encoded form data, where possible.
j) Version control on other levels than the individual record, for example version control on the level of datasets, is not possible. If this is required by an application it needs to be implemented on the client side.
k) Data providers cannot be structured hierarchically either. This makes it inconvenient to handle HPB suppliers as providers, since they cannot be linked to the data provider “CERL HPB”. The supplier level needs to also be modelled as a dataset which increases the described complexity of handling datasets on the client side.

5.2. Summary

The main challenge for HPB data in the Europeana Cloud are the various grouping requirements of records. This can in part be achieved via datasets in Europeana Cloud but is limited by the fact that hierarchical data structures are not well supported. We would need to assign every record to various datasets (which is possible) and keep track of the hierarchical structure on the client side, which is not convenient.

6. Further Evaluation

If we could perform further evaluation of the cloud service, we would explore the following questions in more detail:

- Performance of the cloud service compared to local server storage
- Different approaches to modelling our data structures via datasets

7. Final remarks and recommendations

a) The core functionality of Europeana Cloud is storing data. All other aspects must currently be dealt with on the client side. Purely metadata-based applications, such as the HPB would not benefit from data storage that does not also offer means to access the content of a record itself.

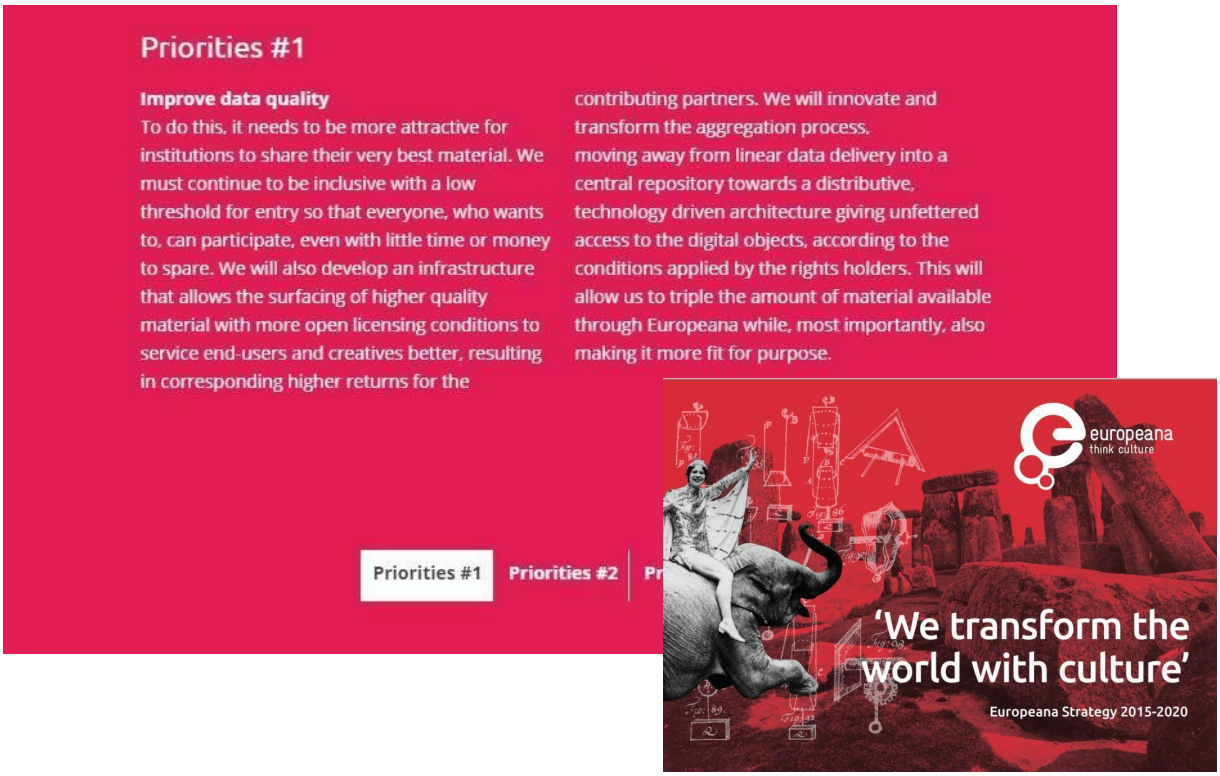
b) Functionality that would require client-side implementation includes version control on other levels than the on the individual record level, particularly versions of datasets. Adding this functionality would be recommended.

c) Handling of hierarchically structured metadata and more extensive metadata management options (as described above) might be useful.

d) In the context of the HPB, user rights management would be far easier if it could also be applied on the level of datasets, which would free client implementations from this task.

Conclusions

Improving data quality is the main priority presented in the Europeana Strategy 2015-2020. The excerpt from from the Strategy outlines a number of approaches to this task. Europeana will endeavour to source better quality data from its providers. But in addition Europeana will explore improving the infrastrucutre in which the data they have already aggregated is presented, and will explore improving the data they already hold. This can be done by improving the context in which the existing data is presented, by deduplication of the dataset, and possibly by improving the data itself. The tasks executed in this Work Package go towards the contextualisation and data improvement referenced in the Strategy.



The exploration of image recognition techniques to identify overlap between Europeana and an external resource like WikiArt, as well as duplication within the Europeana dataset, which was explored in the Europeana Cloud project, complements the extensive work on semantic enrichment that already takes place in Europeana,[28] and fits in with the move towards the distributive, technology driven architecture described in the Europeana Strategic Plan above.

The Europeana Cloud similarity service created by OU goes towards an improved navigation through a huge quantity of data of varying quality. Varying the settings in the similarity tool will give the end user control over how similar the elements in the data set are that is returned as his

[28] For a description of the current state of Europeana enrichment see https://docs.google.com/document/d/1JvjrWMTpMIH7WnuieNqcT0zpJAXUPo6x4uMBj1pEx0Y. Recently a Task Force on enrichment and evaluation presented its report (http://pro.europeana.eu/taskforce/evaluation-and-enrichments). This report will act as a guideline for Europeana to determine the focus of its future efforts in this area.

search result. This will allow the user to set the parameters so that the set that he is provided with is fit for purpose.

The tests performed by the Data Conversion Group, Göttingen, on behalf of CERL, went towards exploring the Europeana stated aim to move from Portal to Platform. Now that the Europeana Cloud infrastructure is available, it is important to explore what a potential customer, in this case CERL, might need to execute its services on top of data held in the Cloud. This resulted in a set of recommendations for dealing with (hierarchical) datasets in the Europeana Cloud.

The outcomes of this work: a contribution to the set of rules used in the Europeana Semantic Enrichment Framework (now expanded beyond textual matching), a REST API for the discovery of textually similar items, and recommendations for dealing with (hierarchical) datasets, each landed in a different place in the organisation of the Europeana Foundation, but together provided small building blocks towards the delivery of the Europeana Strategy 2015-2020.

References

Task 4.3.1

- http://pro.europeana.eu/files/Europeana_Professional/EuropeanaTech/EuropeanaTech_taskforces/Enrichment_Evaluation//FinalReport_EnrichmentEvaluation_102015.pdf
- https://docs.google.com/document/d/1JvjrWMTpMIH7WnuieNqcT0zpJAXUPo6x4uMBj1pEx0Y
- http://onto.dm2e.eu/edm#Agent
- Min-Yen Kan and Yee Fan Tan. Record Matching in Digital Library Metadata, https://www.comp.nus.edu.sg/~kanmy/papers/2008-cacm.pdf
- http://vemir.visualengines.it
- http://www.wikiart.org/en/About
- https://en.wikipedia.org/wiki/Microdata_(HTML)
- https://dl.dropboxusercontent.com/u/630356/WikiArtEnrichment.xml

Task 4.3.2

Chum, Ondrej, James Philbin, and Andrew Zisserman. "Near Duplicate Image Detection: min-Hash and tf-idf Weighting." BMVC. Vol. 810. 2008.

McCandless, Michael, Erik Hatcher, and Otis Gospodnetic. *Lucene in Action: Covers Apache Lucene 3.0*. Manning Publications Co., 2010.

Sadowski, Caitlin, and Greg Levin. "Simhash: Hash-based similarity detection." (2007): 13-22.

Wang, Shenghui, et al. "Hierarchical structuring of cultural heritage objects within large aggregations." *Research and Advanced Technology for Digital Libraries*. Springer Berlin Heidelberg, 2013. 247-259.

Zobel, Justin, and Alistair Moffat. "Exploring the similarity space." *ACM SIGIR Forum*. Vol. 32. No. 1. ACM, 1998.

Describing Europeana Enrichment activities

- https://docs.google.com/document/d/1JvjrWMTpMIH7WnuieNqcT0zpJAXUPo6x4uMBj1pEx0Y
- http://pro.europeana.eu/taskforce/evaluation-and-enrichments