

DELIVERABLE

Project Acronym: EAGLE
Grant Agreement number: 325122
Project Title: Europeana network of Ancient Greek and Latin Epigraphy

AIM Infrastructure Specification

D4.1

version: 1.1

Revision: Final

Authors:

**Giuseppe Amato (CNR-ISTI)
Paolo Bolettieri (CNR-ISTI)
Claudio Gennaro (CNR-ISTI)
Paolo Manghi (CNR-ISTI)
Andrea Mannocci (CNR-ISTI)
Franco Zoppi (CNR-ISTI)**

Contributors:

**Vittore Casarosa (CNR-ISTI)
Fabrizio Falchi (CNR-ISTI)**

Reviewers:

**Nicola Alfarano (GOGATE)
Miguel Angel Sicilia (UAH)**

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

Revision History

Revision	Date	Author	Organisation	Description
0.1	28/08/2013	Andrea Mannocci	CNR-ISTI	Structure of the document
0.2	01/09/2013	Paolo Manghi, Andrea Mannocci	CNR-ISTI	Metadata Aggregation System writing started
0.3	01/09/2013	Giuseppe Amato, Paolo Bolettieri, Claudio Gennaro	CNR-ISTI	Image Retrieval System writing started
0.4	17/09/2013	Andrea Mannocci	CNR-ISTI	Merge of the two parts
0.5	20/09/2013	Franco Zoppi	CNR-ISTI	Internal revision
0.6	23/09/2013	Paolo Bolettieri	CNR-ISTI	Writing of missing parts
0.7	24/09/2013	Andrea Mannocci	CNR-ISTI	Writing of missing parts
1.0 Draft	26/09/2013	Vittore Casarosa, Paolo Manghi, Franco Zoppi	CNR-ISTI	Final internal revision
1.0 Final	03/10/2013	Paolo Bolettieri, Andrea Mannocci, Vittore Casarosa, Paolo Manghi, Franco Zoppi	CNR-ISTI	Integration of reviewers' comments
1.1	31/03/2014	Paolo Bolettieri, Andrea Mannocci, Franco Zoppi	CNR-ISTI	Updated for improvements before 1 st Annual Review

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

TABLE OF CONTENTS

EXECUTIVE SUMMARY	4
1 AIM INFRASTRUCTURE ARCHITECTURAL OVERVIEW.....	5
1.1 FUNCTIONAL GOALS	6
1.1.1 <i>Aggregation System (MAS)</i>	6
1.1.2 <i>Image Retrieval System (IRS)</i>	7
1.2 FUNCTIONAL DEFINITION OF THE AIM INFRASTRUCTURE.....	7
1.2.1 <i>High Level Description</i>	7
1.2.2 <i>Content Provider's Perspective of the AIM Infrastructure</i>	8
1.2.3 <i>Data Curator's Perspective of the AIM Infrastructure</i>	9
1.2.4 <i>Administrator's Perspective of the AIM Infrastructure</i>	11
2 METADATA AGGREGATION SYSTEM	12
2.1 INTRODUCTION TO D-NET.....	12
2.1.1 <i>Data Mediation Area</i>	13
2.1.2 <i>Data Storage and Indexing Area</i>	13
2.1.3 <i>Data Conversion Area</i>	14
2.1.4 <i>Data Curation Area</i>	14
2.1.5 <i>Data Provision Area</i>	15
2.2 MAS ARCHITECTURE	16
2.2.1 <i>EAGLE Aggregation Workflow</i>	16
2.2.2 <i>EAGLE Aggregation Kernel</i>	16
2.2.3 <i>Store Services</i>	17
2.2.4 <i>Metadata Cleaning and Curation Services</i>	18
2.2.5 <i>Index Service</i>	18
2.2.6 <i>Export Services</i>	18
3 IMAGE RETRIEVAL SYSTEM	20
3.1 IRS FUNCTIONALITIES.....	20
3.1.1 <i>Image Retrieval Agent</i>	20
3.1.2 <i>Image Feature Extractor</i>	21
3.1.3 <i>Image Indexer and Similarity Search</i>	22
3.1.4 <i>Image Recognizer</i>	23
3.2 IRS ARCHITECTURE	26
3.2.1 <i>Image Retrieval Agent (IRA)</i>	26
3.2.2 <i>Image Feature Extractor Component (IFE)</i>	27
3.2.3 <i>Image Indexer</i>	27
3.2.4 <i>Image Recognition API</i>	28
3.2.5 <i>Similarity Search API</i>	29
4 CONCLUSIONS	32

EXECUTIVE SUMMARY

The purpose of this deliverable is to provide a high level overview of the Aggregation and Image Retrieval system (AIM) in EAGLE. For the sake of readability, this document provides a description of the two subsystems composing the AIM infrastructure, namely the Metadata Aggregation System (MAS) and the Image Retrieval System (IRS), focusing the reader's attention on functional requirements, constraints and architectural aspects, leaving out detailed programming concerns and API specifications.

This deliverable complies with the EAGLE DoW, as described in Work Package 4, EAGLE Aggregation and Image Management Infrastructure, and in particular it fulfills the requirements outlined in tasks T4.1 (Aggregator infrastructure detailed design, implementation and maintenance) and T4.3 (Image recognizer infrastructure detailed design, implementation and maintenance).

It is intended to clarify some key aspects for the benefit of all EAGLE partners and work packages, and it aims to ease the work of the following interrelated tasks: T3.1 Definition of the EAGLE metadata format; T3.2 Metadata mapping; T3.3: Planning and preparing the ingestion to Europeana.

The document is organized as follows: the first section introduces the high-level architectural aspects of the AIM infrastructure, whilst the second and third sections describe in more detail the two subsystems composing the AIM infrastructure, respectively the Metadata Aggregation System (MAS) and the Image Retrieval System (IRS).

1 AIM INFRASTRUCTURE ARCHITECTURAL OVERVIEW

In this chapter, we describe the expected functional requirements to be satisfied by the Aggregation and Image Management infrastructure and introduce the high-level architectural aspects composing the system.

Figure 1-1 shows the overall architecture of the EAGLE system. It is expected that the EAGLE portal and the End-User Services will be hosted in the same physical machine as the AIM Infrastructure, although this does not represent a constraint.

Different types of user will be supported: Content Provider, Data Curator, Administrator, End-User. Whilst the formers will interact with the AIM Infrastructure directly, the latter will interact with the Portal and the supplied Services and Applications. The system will implement those types with distinct roles and permissions and will consequently supply authentication and access functions according to such roles.

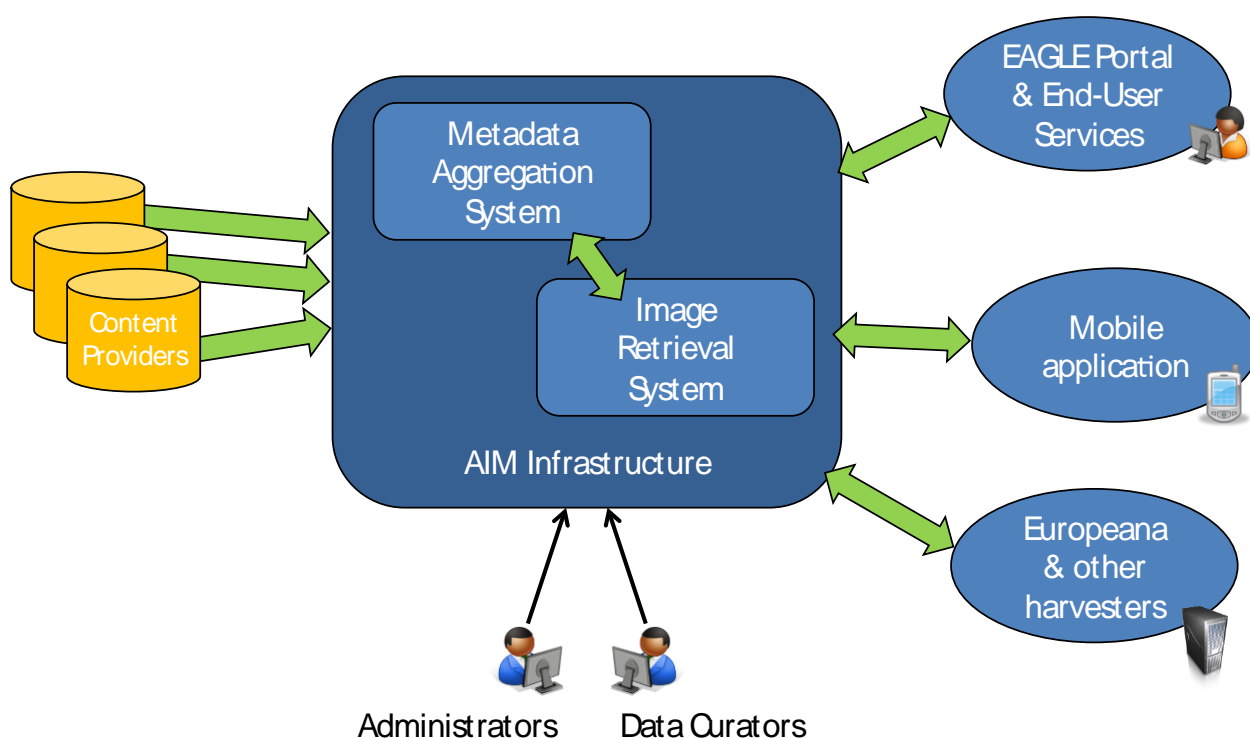


Figure 1-1 The AIM Infrastructure within EAGLE – High Level View

Figure 1-2 shows a detailed view of the overall architecture – decomposed in different modules along with the WPs responsible for them - and its implementation path. Different colors (and patterns) show respectively:

- **Green (no pattern):** components implemented in Release 1.
- **Yellow (dotted):** components implemented in Release 2.
- **Grey (horizontal lines):** components whose implementation is to be confirmed yet.
- **Blue (vertical lines):** components whose responsibility is outside the scope of the AIM Infrastructure.
- **White (dotted grid):** component whose responsibility is outside the scope of the EAGLE project.

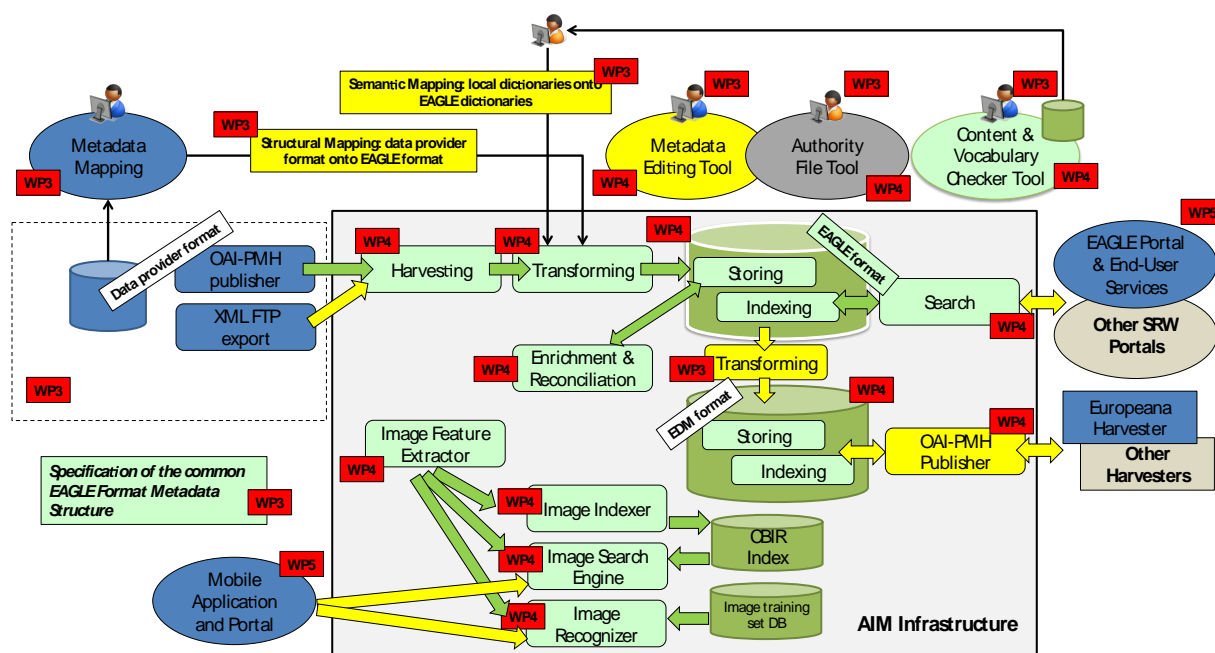


Figure 1-2 The AIM Infrastructure within EAGLE – Detailed View

The different modules belonging to the AIM Infrastructure are described in Section 2 and 3.

1.1 FUNCTIONAL GOALS

The EAGLE AIM infrastructure can be decomposed into two subsystems, namely the Metadata Aggregation System (MAS) and the Image Retrieval System (IRS). These two components will provide the functional requirements depicted in Figure 1-3.

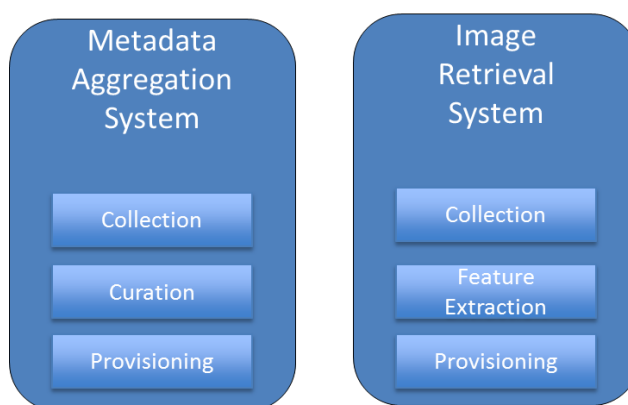


Figure 1-3 AIM Functional goals

1.1.1 Aggregation System (MAS)

The Metadata Aggregation System (MAS) will collect sets of XML-formatted metadata coming from all the Content Providers (CPs) joining EAGLE network. Each collected metadata record needs to be transformed conforming to a homogeneous structural format (EAGLE Metadata Format - EMF) and stored for further elaboration. Digital image resources possibly pointed to by metadata records are fetched following their

URLs and stored as well in a component of the MAS; this information will be the input for IRS system (see section 1.1.2).

Once collected and transformed into EMF, metadata need to be cleaned (i.e. harmonized by applying vocabularies for allowed values) and possibly curated (i.e. selectively edited). The set of cleaned and curated metadata records constitutes the EAGLE Information Space.

Once the information space reaches the desired quality, its content is ready to be indexed, ingested into Europeana and disseminated to the public and to external applications (e.g. web portals and mobile apps). More specifically, the information space is *i)* used as input for the indexing process to make it accessible from portals via web searches, and *ii)* exposed record-by-record toward Europeana or any other third-party system that might be interested in such a content.

1.1.2 Image Retrieval System (IRS)

The Image Retrieval System will receive in input the image of an epigraph and will provide effective and efficient image similarity search and image content recognition of epigraphs.

The images provided by the CPs and collected by MAS will be processed by some IRS components (Image Feature Extractor, Image Indexer, CBIR¹ Index) to build the index that will allow a fast and efficient similarity search during the query phase.

Various search paradigms can be supported by means of the IRS functionality. For example, an image associated with the result of a metadata based search can be used as input for a IRS query to refine the search, or the EAGLE content can be used to get information concerning images provided by the users (for instance: *where is the epigraph contained in this picture located?*).

In addition, for those epigraphs where a set of images is available (training set), each training set will be processed to extract the main features characterizing the epigraph. The training sets and the characterizing features will be the base for building another index, to be used by the image recognizer in order to decide if a received image (during the query phase) can be classified as belonging to one of the existing sets or not. In this way, in many cases, the recognizer is able to properly recognize the content of a query image even if the image given in the query was never stored in the database.

1.2 FUNCTIONAL DEFINITION OF THE AIM INFRASTRUCTURE

1.2.1 High Level Description

Figure 1-4 shows the global functional overview, the interaction and the flow of information between the different functional blocks and components of the AIM.

¹ Content-Based Image Retrieval.

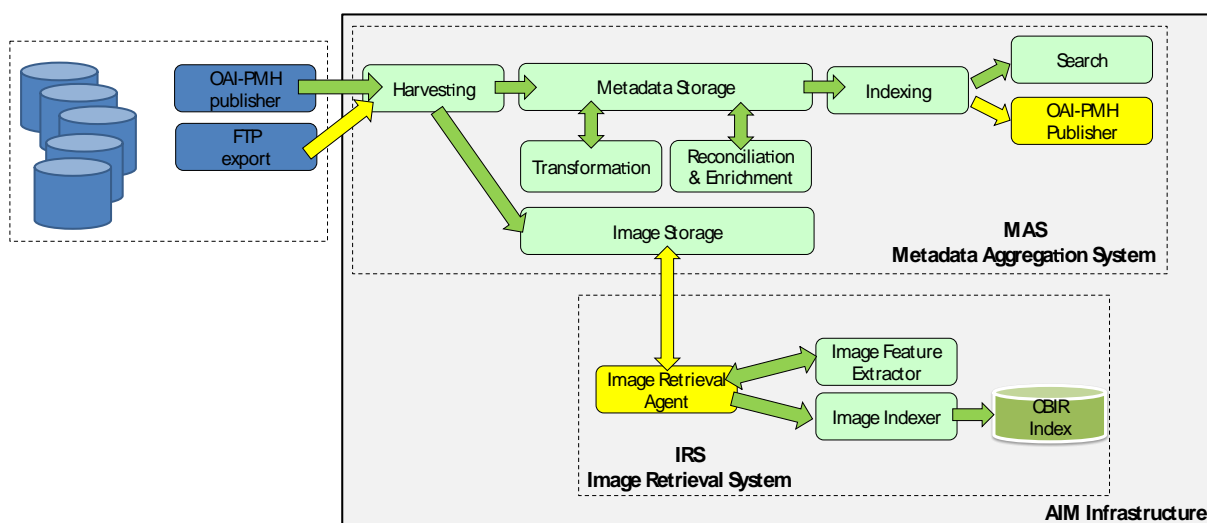


Figure 1-4 Functional AIM Infrastructure - Global picture

The MAS will periodically harvest metadata records (Harvesting Service) made available from CPs and store them within the data infrastructure (Metadata Storage Service). Then, the MAS will start building the EAGLE Information Space step by step. Native metadata records enter a processing flow where they are structurally transformed to comply with a common metadata schema, the EAGLE Metadata Format - EMF (Transformation Service). At the same time, URLs to images pointed by metadata descriptions are visited and the relative images are persisted (Image Storage Service) in the AIM. Once metadata are transformed, metadata records are cleaned and can be further improved by curators via selective editing (Cleaning Service). Eventually they will get into their final revision and will be ready to be indexed (Indexing Service) and disseminated either to external public through the web portal and the mobile application (Search Service), or to third party software agents via OAI-PMH (OAI-PMH Publisher Service).

The IRS will periodically call the Image Storage Service via the Image Retrieval Agent to get new items to be indexed. For each new item collected, the IRS extracts its visual descriptors and encodes the visual descriptors in an efficient textual format (Image Feature Extractor Service), then puts the encoded descriptors into the CBIR Index (Image Indexer Service).

1.2.2 Content Provider's Perspective of the AIM Infrastructure

The Content Providers willing to provide their content to the EAGLE infrastructure have to take into account a few important things:

1. Metadata records will be considered valid and will be imported only if in XML format.
2. Content providers have to provide mappings from the local metadata schema to the EAGLE Metadata Format (defined by WP3 in task T3.1).

In order to get ready and begin to provide content, a compliant Content Provider has to:

1. Register each repository that will provide data to EAGLE with the Metadata Aggregation System.
2. Provide data access protocol information for each repository.
3. Provide the proper mapping from the local scheme to the EAGLE Metadata Format.

Upon providing all the aforementioned information the following ingestion flow will take place:

1. The harvesting process is performed for the first time.
2. Collected metadata records are examined and checked for consistency.
3. The metadata records are transformed to the EAGLE Metadata Format according to the mapping provided.
4. Feedback about the outcome of the preceding steps is given to content provider.
5. Steps 2-3-4- are reiterated until all the problems have been solved.

When the first harvesting from a Content Provider has been proved to work the process can be fired periodically without any other interaction from CPs side.

1.2.3 Data Curator's Perspective of the AIM Infrastructure

Once all metadata records are properly collected and transformed, metadata records enter a cleaning process, which harmonizes their values with the values provided by controlled vocabularies. Content Providers share and edit the vocabularies by means of the Tematres² vocabulary server. A sample screenshot of Tematres frontend editor is shown in Figure 1-5.

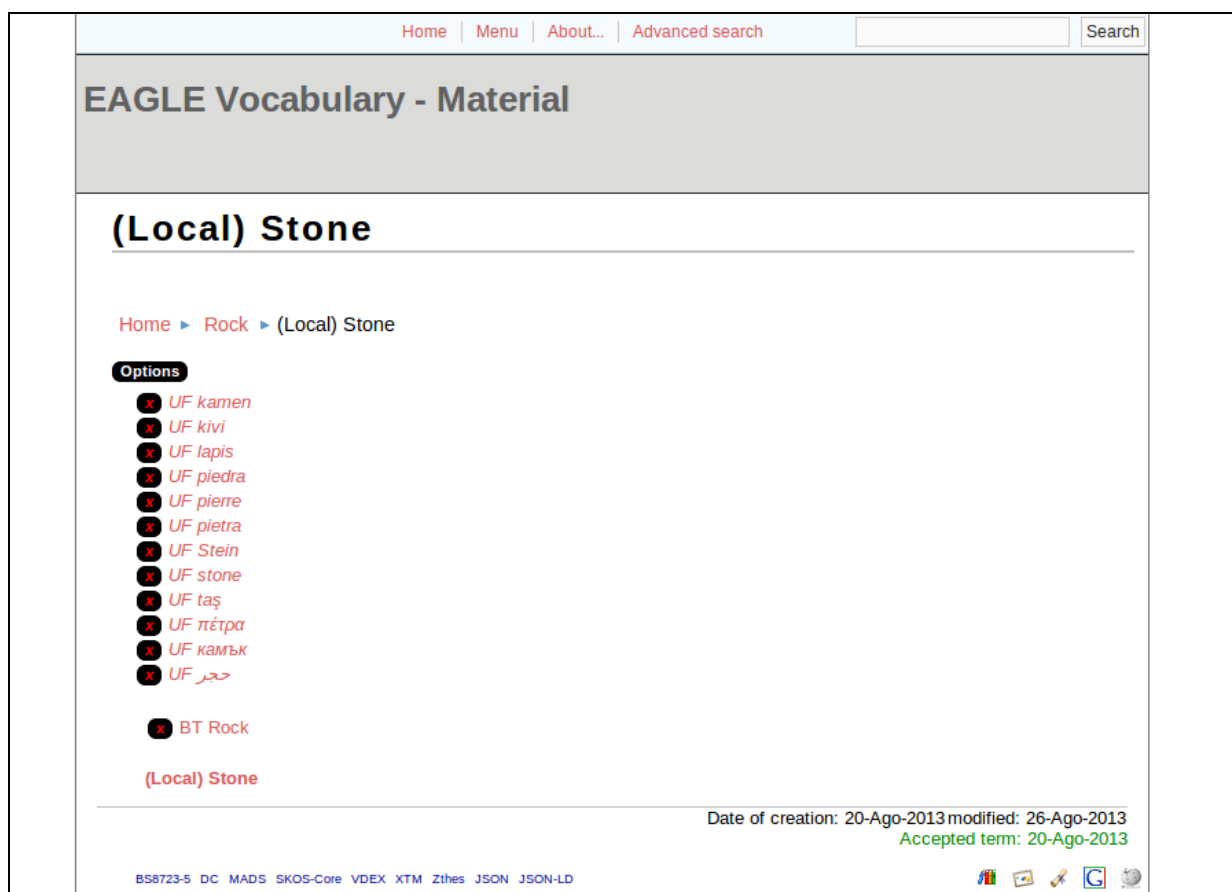
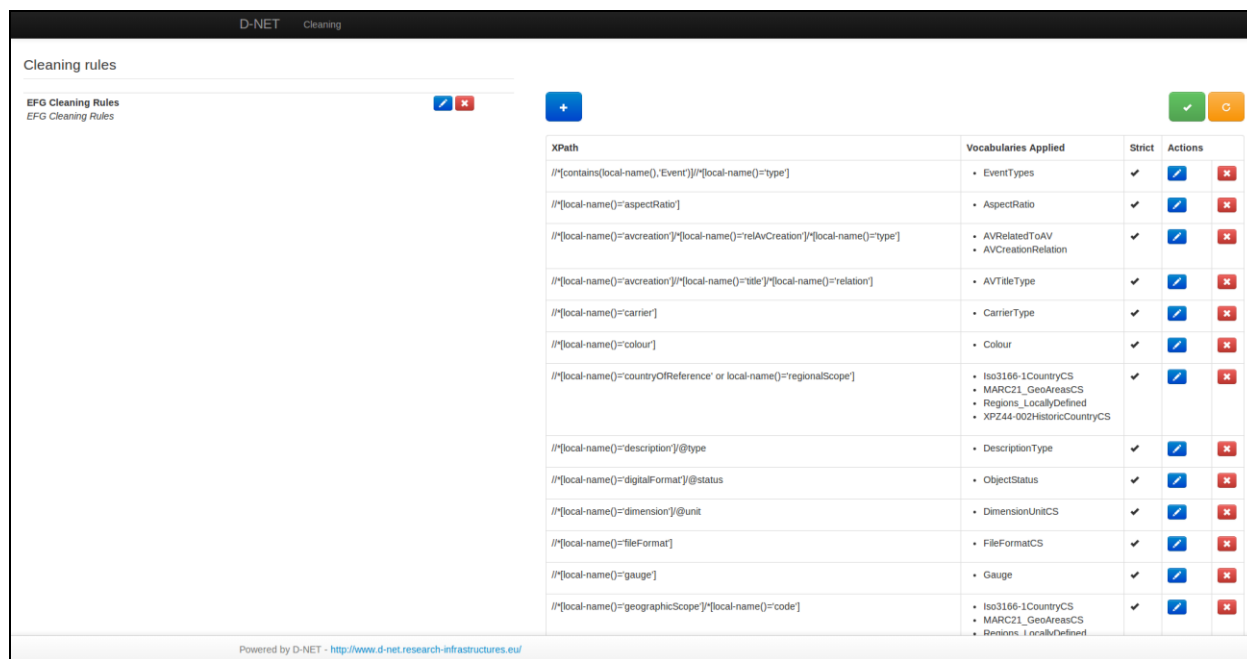


Figure 1-5 Tematres vocabulary server, sample page

² Tematres vocabulary server, <http://www.vocabularyserver.com/>

Vocabularies that have been edited in the Tematres server will be then imported into the AIM for metadata cleaning purposes (more details in 2.2.4).

How to use such vocabularies is to be specified in the Cleaning Rules page, shown in Figure 1-6. Here, it is possible to associate (by properly defining an XPATH) one of the vocabularies imported in the step above with a specific field in the EAGLE Metadata Format.



XPath	Vocabularies Applied	Strict	Actions
//*[contains(local-name(),'Event')]/*[local-name()='type']	• EventTypes	✓	✓ ✕
//*[local-name()='aspectRatio']	• AspectRatio	✓	✓ ✕
//*[local-name()='avcreation']/*[local-name()='relAvCreation']/*[local-name()='type']	• AVRelatedToAV • AVCreationRelation	✓	✓ ✕
//*[local-name()='avcreation']/*[local-name()='title']/*[local-name()='relation']	• AVTitleType	✓	✓ ✕
//*[local-name()='carrier']	• CarrierType	✓	✓ ✕
//*[local-name()='colour']	• Colour	✓	✓ ✕
//*[local-name()='countryOfReference' or local-name()='regionalScope']	• Iso3166-1CountryCS • MARC21_GeoAreasCS • Regions_LocallyDefined • XPZ44-002HistoricCountryCS	✓	✓ ✕
//*[local-name()='description']/@type	• DescriptionType	✓	✓ ✕
//*[local-name()='digitalFormat']/@status	• ObjectStatus	✓	✓ ✕
//*[local-name()='dimensionUnit']/@unit	• DimensionUnitCS	✓	✓ ✕
//*[local-name()='fileFormat']	• FileFormatCS	✓	✓ ✕
//*[local-name()='gauge']	• Gauge	✓	✓ ✕
//*[local-name()='geographicScope']/*[local-name()='code']	• Iso3166-1CountryCS • MARC21_GeoAreasCS • Renions_LocallyDefined	✓	✓ ✕

Figure 1-6 Cleaner service rules

The metadata cleaning process will apply automatically all the rules specified. Since the processing is automatic, it is advisable to check for consistency at the end of the elaboration, especially in early phases of the prototypal implementation. Such validation can be performed thanks to two other customizable tools: the Vocabulary Checker and the Content Checker (see subchapter 2.1.4).

Once metadata are harmonized, a content provider can edit metadata content, fixing some errors or adding missing information via the Metadata Editor tool, whose sample screenshot is shown in Figure 1-6.

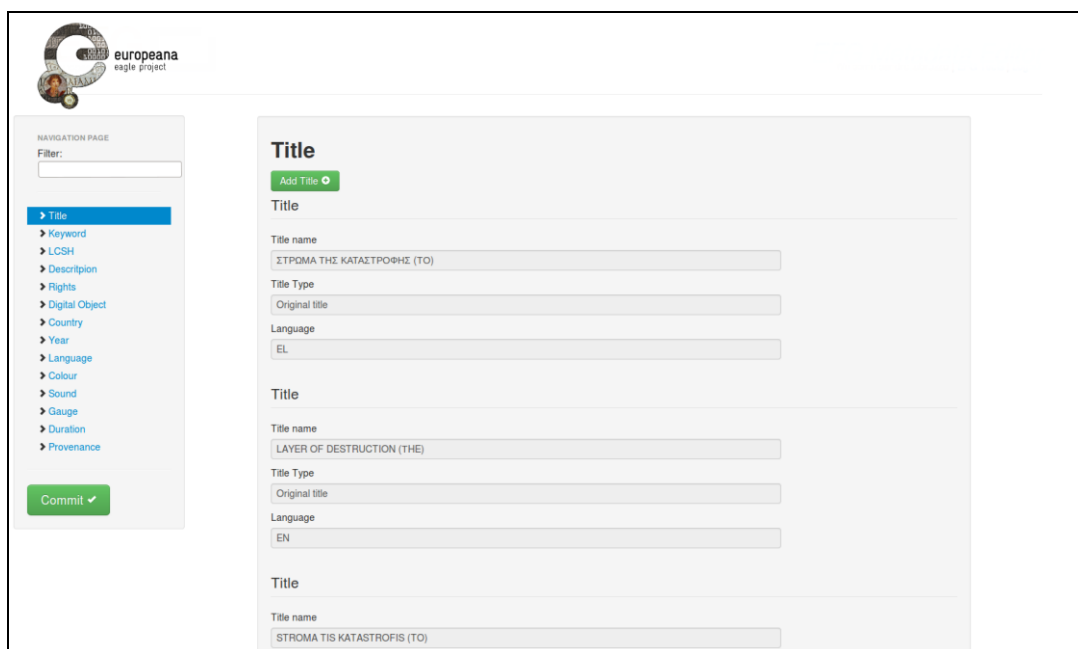


Figure 1-7 Metadata editor

After the final revision, metadata are ready to be ingested into Europeana. The definition of rules for converting metadata to Europeana Data Model (EDM), i.e. T3.2, is presently ongoing. After testing and refining also this second round of mapping, the AIM infrastructure will provide proper mechanism for automatic conversion and ingestion into Europeana platform. Content Providers will not need to be involved in this phase, as the mapping from the EAGLE Metadata Format to EDM and the ingestion to Europeana will be carried on by the AIM.

1.2.4 Administrator's Perspective of the AIM Infrastructure

D-NET exposes a set of configurable interfaces with administrator privilege (known as Inspector), which offers a broad range of low-level functionalities over the entire data infrastructure.

Inspector interfaces are mostly used for diagnosis purposes and low-level back-office interventions by D-NET developers. Thus, access credentials are generally not handed to Content Providers as wrong or misconceived configurations provided in such interfaces may result in serious harm to the data infrastructure.

In EAGLE, a set of high-level administration web tools will be developed in order to provide a correct administration view of D-NET system, avoiding at the same time the chance of severe misconfigurations. Through such tools it will be possible to define workflows and orchestrate the interaction among different services composing the infrastructure as well as monitor the status of past/ongoing jobs and schedule periodic tasks.

2 METADATA AGGREGATION SYSTEM

The EAGLE Metadata Aggregation System will be developed by capitalizing on the D-NET software toolkit, developed within the DRIVER project efforts and extended within the OpenAIRE, OpenAIREplus³, EFG, EFG1914⁴ and HOPE⁵ projects. Within the EAGLE framework, D-NET will be further extended and customized in order to meet new EAGLE-specific requirements and peculiarities.

In this chapter we first outline the D-NET architecture and then describe the customizations to be performed in order to satisfy the EAGLE requirements.

2.1 INTRODUCTION TO D-NET

D-NET is an open source, general-purpose software conceived to enable the realization and operation of Aggregative Data Infrastructures (ADIs) and to facilitate their evolution and maintenance over time.

D-NET implements a service-oriented framework based on standards, namely Web Services with SOAP and REST APIs, where ADIs can be constructed in a LEGO-like approach, by selecting, customizing, and properly combining D-NET services as building blocks. The resulting ADIs are systems, which can be customized, further extended (e.g. by the integration of new services), and scaled (e.g. storage and index replicas can be maintained and deployed on remote nodes to tackle multiple concurrent accesses or very-large data size) at run-time.

D-NET offers a rich and expandable set of services targeting data collection, processing, storage, indexing, curation and data provision aspects. Services can be customized and combined to meet the data workflow requirements of a target user community. D-NET services can be partly or fully replicated and distributed over different machines depending on the level of the QoS required by the specific community. In general, redundant services improve fault tolerance, reduce the overload of each instance, and enable dynamic reorganization of the environment when a server becomes unreachable.

Figure 2-1 presents how several D-NET services implement the high-level architecture and functionalities. The services implemented so far can be grouped into five categories: Data Mediation, Data Storage and Indexing, Data Conversion, Data Curation, Data Provision.

In the following subchapters, we provide a brief description of the main services, focusing our attention on the ones that we foresee will be used within the EAGLE project development.

³ <http://www.openaire.eu/>

⁴ <http://www.europeanfilmgateway.eu/it>

⁵ <http://www.peoplesheritage.eu/>

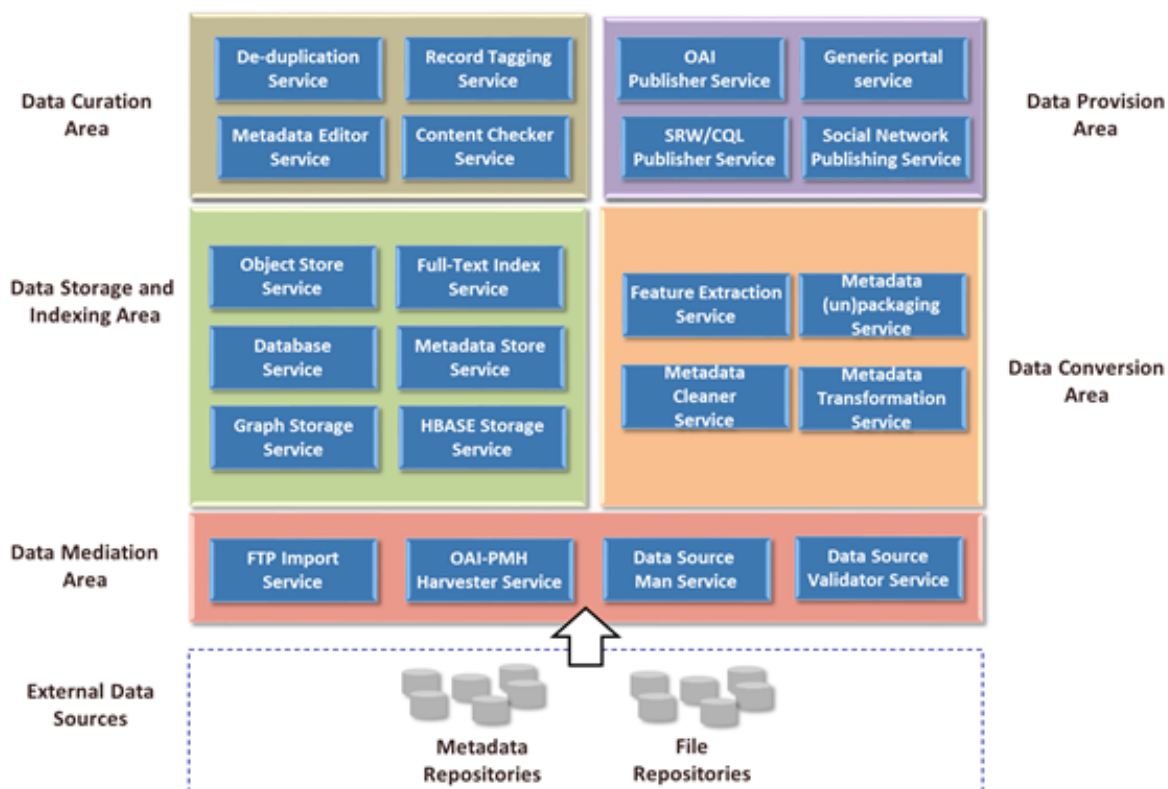


Figure 2-1 D-NET overview

2.1.1 Data Mediation Area

Services in this area are responsible for the interaction with the Content Providers and for collecting the metadata records made available.

Data Source Manager Service

This service manages the set of CPs by providing registration and de-registration services, and monitoring their status and statistics within the MAS. It is also possible to define per-content-provider policies in order to access and collect properly data there exposed.

Harvester and Import Services

D-NET offers a suite of services for on-demand and scheduled data collection based on the following standard protocols: OAI-PMH, FTP, FTPS (FTP over SSL/TSL), SFTP (SSH File Transfer Protocol), HTTP/HTTPS.

Also it is possible to collect metadata records directly from the local file-system in the machine where the D-NET installation resides.

2.1.2 Data Storage and Indexing Area

Services in this area manage storage and access to files and metadata records by providing several data storage supports as backend. Developers can choose and configure the proper underlying storage system according to functional requirements and common data model of the data infrastructure at hand.

Metadata Store Service

The Metadata Store Service is in charge of storing metadata records harvested from Content Providers and making such information persistent into data stores within D-NET.

Object Store Service

The Object Store Service exposes a RESTful API for storing and managing arbitrary “non metadata” objects and retrieving them via unique URIs for later use.

Index Service

The Index Service manages full-text indices to support reliable queries and fast service time. The index implementation chosen as backend is Solr⁶.

2.1.3 Data Conversion Area

Services in this area offer functionalities for the conversion of XML metadata records, regardless of the structure and semantics of their schemas, and the conversion of files, regardless of their storage formats.

Transformation Service

The Transformation Service can be configured to transform metadata records from one schema to another (e.g. from MARC to Dublin Core) by simply providing XSLT mapping or by providing scripts written in Groovy⁷ (an extension of Java). D-NET data managers can create, update, remove and re-use such mappings and configure the service to apply one format to a given input (e.g. a metadata store) at given time intervals.

Cleaner Service

The Cleaner Service harmonizes the metadata records by unifying their values according to controlled vocabularies, with terms-to-terms rules defined by curators.

2.1.4 Data Curation Area

Services in this area offer functionalities for curation and enrichment of metadata records. As to the curation policy to be implemented, each CP might appoint some people to take care of the curation process; as an alternative, the EAGLE consortium could agree upon delegating such task to a centralized role. Both solutions are supported by D-NET.

Metadata Editor

The Metadata Editor Service allows data curators to add, edit and delete metadata records, as well as establish relationships between existing records, even if coming from different sources.

⁶ Apache Solr, <http://lucene.apache.org/solr/>

⁷ Groovy, <http://groovy.codehaus.org>

Content Checker

The Content Checker Service provides tools helping curators to find mapping mistakes and semantic inconsistencies in metadata records. Non-conforming records will not be visible from the outside until they are corrected and validated.

Vocabulary Checker

A customized instance of the Content Checker will validate the value of given fields by comparing them with a list of allowed values coming from a controlled vocabulary registered in the system. Non-conforming values are highlighted for further action and will not be visible from the outside.

A mockup of the interface is depicted in Figure 2-2.

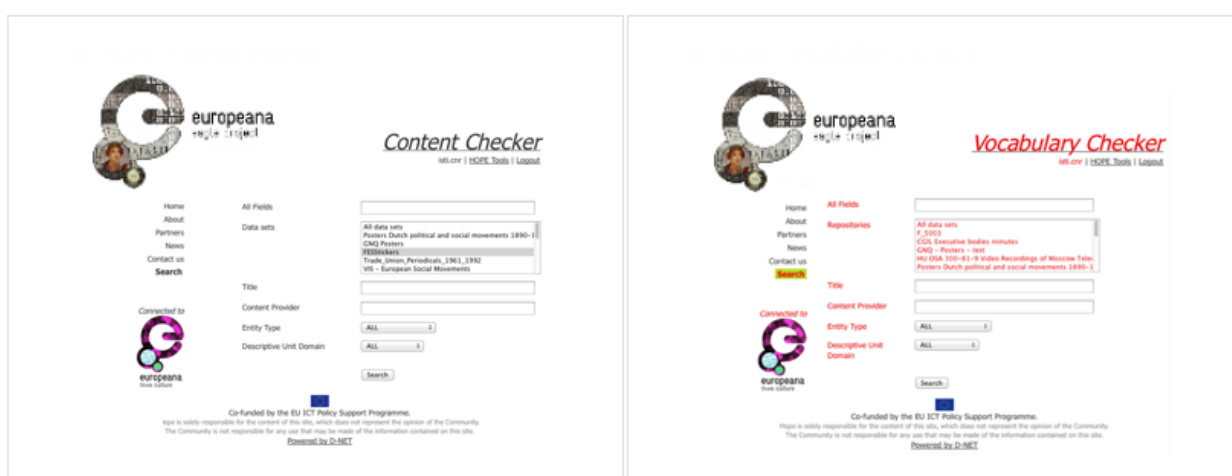


Figure 2-2 Content Checker and Vocabulary Checker

2.1.5 Data Provision Area

Services in this area will provide the content of the information space to external applications via standard APIs (see below).

Search Service

This service provides two interfaces, one accepting queries formulated in the Contextual Query Language (CQL⁸), the other accepting queries formulated according to the Solr syntax⁹, enabling third party applications, such as Web portals, to perform queries over D-NET indices.

OAI-Publisher Service

The OAI-Publisher Service enables third-party harvesters to access metadata records present in the information space via the OAI-PMH 2.0 protocol. The service can provide different metadata formats obtained on-the-fly by applying XSLT mappings at real-time.

⁸ Introduction to CQL, <http://zing.z3950.org/cql/intro.html>

⁹ Apache Solr query syntax, <http://wiki.apache.org/solr/SolrQuerySyntax>

2.2 MAS ARCHITECTURE

In this subchapter, we outline first the workflow satisfying functional requirements of the EAGLE Metadata Aggregation System. Then we define which D-NET services are suitable to be used as building-block in the creation of the infrastructure. Whenever the D-NET actual implementation does not completely satisfy the requirements, we will mention the customizations and extensions to be performed in order to fulfill EAGLE-specific requirements.

2.2.1 EAGLE Aggregation Workflow

The overall workflow of the EAGLE infrastructure is depicted in Figure 2-3, where all the blocks are selected D-NET components.



Figure 2-3 EAGLE Aggregation Workflow

The XML metadata records provided by the Content Providers are gathered (by harvesting, file transfer or other ad-hoc means) and stored into the Native Metadata Store. After that, metadata in native format are transformed into the EAGLE Metadata Format, making them structurally uniform, and stored in a Transformed Metadata Store. At this point the workflow forks into two: one branch deals with pure metadata enhancement, curation and indexing. The other branch is instead in charge of fetching images pointed to by collected metadata and storing them in an Object Store for later usage by the Image Retrieval System (see Chapter 3).

It is crucial to provide a staged metadata storage so that processing and elaborations late in the workflow can take place without repeating all the steps from the beginning. Thus, each Metadata Store can be considered as a “safe-point” in data processing.

2.2.2 EAGLE Aggregation Kernel

Content Provider Manager Service

Content Providers have to register to the Metadata Aggregator System by providing core information that will become the Content Provider's “profile” in the AIM infrastructure. A CP must then register one or more metadata sets containing the data that he/she is willing to provide to EAGLE. The information needed to register a data-set is: access typology (OAI-PMH, FTP, SSH, custom API), entry-point and input metadata format. If the access typology is OAI-PMH, CPs can also specify a particular OAI-Set, telling the Aggregator to harvest content only from that OAI-Set and not from the entire OAI-PMH repository.

Metadata Harvester Service

The core business of the EAGLE project consists in the aggregation of epigraphic material from CPs. The standard Harvester Service present in D-NET (basically OAI-PMH and FTP) should be adequate in the majority of the cases. In some cases, a CP has defined a proprietary API for external access to data. In

such cases, the Harvester Service will be customized for handling each exception to the generic rule. In any case, Content Providers are expected to always export their metadata in XML-format.

Metadata Transformation Service

The EAGLE transformation workflow is composed by one step of transformation which maps records from the native metadata schema into EMF as defined by Task T3.1 in WP3. The aforementioned transformation process is based on a "static mapping" (i.e. a set of rules provided as XSLT mappings or as a Groovy script) from the local format(s) to EMF. This service takes care of the thumbnails creation and store as well.

Image Harvester Service

An Image Harvester (not provided by the actual D-NET implementation) will be developed in order to follow image URLs present in metadata descriptions, and download them either from assets local to CPs, or EAGLE mediawiki instance, or third party image hosting platforms. Images are stored within the AIM for further processing (by the Image Retrieval System). This operation will be done right after the transformation step, as the image URL will be present in a fixed XML path inside the harmonized metadata format. Image resources will be stored inside the Object Store Service, which is described below.

2.2.3 Store Services

Metadata Stores

Several Metadata (MD) Store Services need to be configured in order to host metadata records of every content provider in different stages of processing within the EAGLE AIM infrastructure. MDStores are configured to store each XML record as an XML document in a "blob field" of the underlying database (i.e. MongoDB in our case).

In particular, for each data-set exported from the CP, the following MDStores are needed:

- One MDStore for original metadata records in their native format (local to the CP). This MDStore is fed by the Harvesting Service.
- One MDStore for metadata records transformed into EMF. Such MDStore is used by the transformation Service.
- One MDStore for EAGLE cleaned metadata. This MDStore is fed during the curation processes by applying the vocabulary matching tables and the rules for the content normalization.
- One MDStore for EAGLE final metadata. The MDStore stores curated metadata in their final version and its content is the input of the indexing process.

Image Store

The Image Store Service is the place in which the EAGLE MAS will store images fetched from links present in metadata descriptions.

Such service will be realized as a customization of D-NET Object Store Service. The present Object Store implementation in D-NET is able to collect and store the images provided by the CPs, but does not support the interaction with the EAGLE Image Retrieval System. In fact, the IRS must be able to know which images have been already processed and needs to be aware about the presence of new images queuing for feature extraction. Since both the MAS and the IRS are being developed by CNR-ISTI, the

API defining interactions between the Image Store Service and the IRS will be defined as the collaboration goes on.

2.2.4 Metadata Cleaning and Curation Services

Cleaner Service

The implementation of the Cleaner Service in the present D-NET implementation relies on controlled vocabularies local to D-NET installation. In EAGLE vocabularies will be edited and shared by using the facilities of the Tematres vocabulary server. The Cleaner Service will be extended in order to “convert” Tematres-defined vocabularies into D-NET-compliant vocabularies.

The Cleaner Service is designed to use vocabularies expressed as flat lists of reference terms to which is associated a set of synonyms. Tematres vocabularies and their terms and relative synonyms will be flattened.

Curation tools

EAGLE curators (experts from the Content Provider community) may be willing to perform further automatic and semi-automatic curation activities.

In EAGLE, the Metadata Aggregation System will include EAGLE-specific customizations of Metadata Editor, Content Checker and Vocabulary Checker that will provide user-friendly web interfaces to curators willing to verify the consistency of the aggregated metadata and possibly improve and enrich them.

2.2.5 Index Service

Index services will be configured to store and reply to queries over EAGLE metadata records. The index will be optimized to serve as efficiently as possible the specific queries received by the portal. More specifically, the index will contain EAGLE records relative to descriptive units, digital resources, places, agents, concepts, events, Content Providers information, and all the enrichments provided in the curation phase.

2.2.6 Export Services

Search Service

The Search Service will enable EAGLE portal developers to access the MAS information space via queries either expressed in CQL¹⁰ or in Solr syntax¹¹.

OAI-publisher

The OAI-publisher module is needed in order to implement the EAGLE ingestion plan toward Europeana. The service takes as input the collection of metadata stored in the “right-most” MDStore in the workflow (i.e. the “Final MD Store” containing the most accurate version of metadata in EMF).

¹⁰ Introduction to CQL, <http://zing.z3950.org/cql/intro.html>

¹¹ Apache Solr query syntax, <http://wiki.apache.org/solr/SolrQuerySyntax>



The conversion of the whole corpus of metadata from EMF to Europeana Data Model (EDM) can be performed on-the-fly by the OAI-Publisher itself just by providing an XSLT mapping. All the due structural transformation will be performed at export time without using any further MD Store.

3 IMAGE RETRIEVAL SYSTEM

Content Based Image Retrieval (CBIR) is becoming an effective way for searching digital libraries, as the amount of available image data is constantly increasing. CBIR applications are increasingly becoming useful in accessing cultural heritage information, as a complement to metadata based search. In fact, in some cases metadata associated with images do not describe the content with sufficient details to satisfy the user queries, or metadata are completely missing. For example, images containing reproductions of work of art contain a lot of implicit information that is not generally captured in manually or automatically generated metadata.

Various search paradigms can be supported by means of the IRS functionality. For example, an image associated with the result of a metadata based search can be used as input for a IRS query to refine the search, or the EAGLE content can be used to get information concerning images provided by the users (for instance, where is located the epigraph contained in this picture?).

The Image Retrieval System will provide two modes of recognition of an image provided in input as a query image. In the first mode, called Similarity Search, the query result will be a list of images contained in the data base, ranked in order of similarity to the query image. In the second mode, called Recognition Mode, the result of the query will be the information associated with the recognized image, or a message indicating that the image was not recognized. In this second mode it is possible for an epigraph to appear in the query image in any position, and also as part of a more general picture (e.g. the picture of an archeological site, or the scanned image of the page of a book).

3.1 IRS FUNCTIONALITIES

The image search and recognition service is composed of the following sub-services:

- Image Retrieval Agent
- Image Feature Extractor
- Image Indexer and Similarity Search
- Image Recognizer

In the following sections each component will be analyzed in its functionalities.

3.1.1 Image Retrieval Agent

The Image Retrieval Agent (IRA) is the component in charge of managing the Image ingestion process. As schematically indicated in Figure 3-1, the Image Storage component of the MAS collects the images retrieved during the harvesting process. It stores the images in its own database providing an interface to retrieve and manage them. So the main task of the IRA is to collect images from the Image Storage in order to build and maintain updated the Image Index.

At each preset time interval the IRA will query the Image Storage to check if there are new items to be handled. If any, it will start the image processing flow (features extraction, text encoding of the features, features storing and indexing), and at the end it will flag the item in the Image Storage as processed.

More precisely, at each preset time interval, the IRA:

- checks with the Image Storage if new images to be inserted, to be deleted or to be updated in the index are available;

- for each new item:
 - calls the Image Feature Extractor, to extract the mathematical description of the image;
 - calls the Image Indexer to transform the image visual features into a text encoding;
 - inserts the image visual features in the features storage database;
 - inserts the image text encoding in the index;
 - flags the item in the Image Storage.

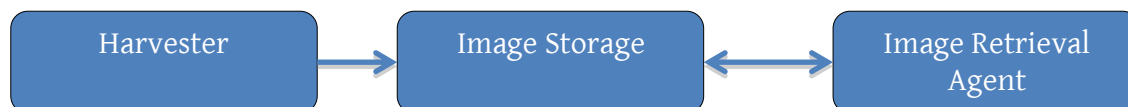


Figure 3-1 Iteration between Harvesting and Image Retrieval System

3.1.2 Image Feature Extractor

Given an image (to be added to the database or provided as a query), the image feature extractor analyzes its visual content to generate a visual descriptor. Visual descriptors are mathematical descriptions of the Features in an image. A feature captures a certain visual property of an image, either globally for the entire image or locally for a small group of pixels.

Global features are computed to capture the overall characteristics of an image. The most commonly used global features include those reflecting color, texture, shapes, and interest (or salient) points in an image.

Local features are low level descriptions of Keypoints in an image. Keypoints are interest points in an image that are invariant to scale and orientation, and therefore are very useful for determining the similarity of two images in all those cases where the second image has a different orientation or scale.

The result of global and local extraction of visual features is a mathematical description of the image visual content that can be used to compare different images, judge their similarity, and identify common content. The IRS will use global visual descriptors for the Similarity Search task, and local visual descriptors for the Recognition task.

As stated before, global features are based on the overall properties of an image, such as the histogram of colors present in the image, the relative position of shapes identified in the image, the texture (information about the spatial arrangements of colors) of the image or of selected sub-regions. A number of mathematical libraries already exist for extracting global features from an image.

Figure 3-2 shows the local features extraction process. First a number of keypoints (usually in the order of one thousand) are identified and selected in the image (colored dots in the second image), and then for each point its coordinates, scale and orientation are computed. Also in this case there are already existing mathematical libraries that can be customized to the specific aspects of epigraph images.

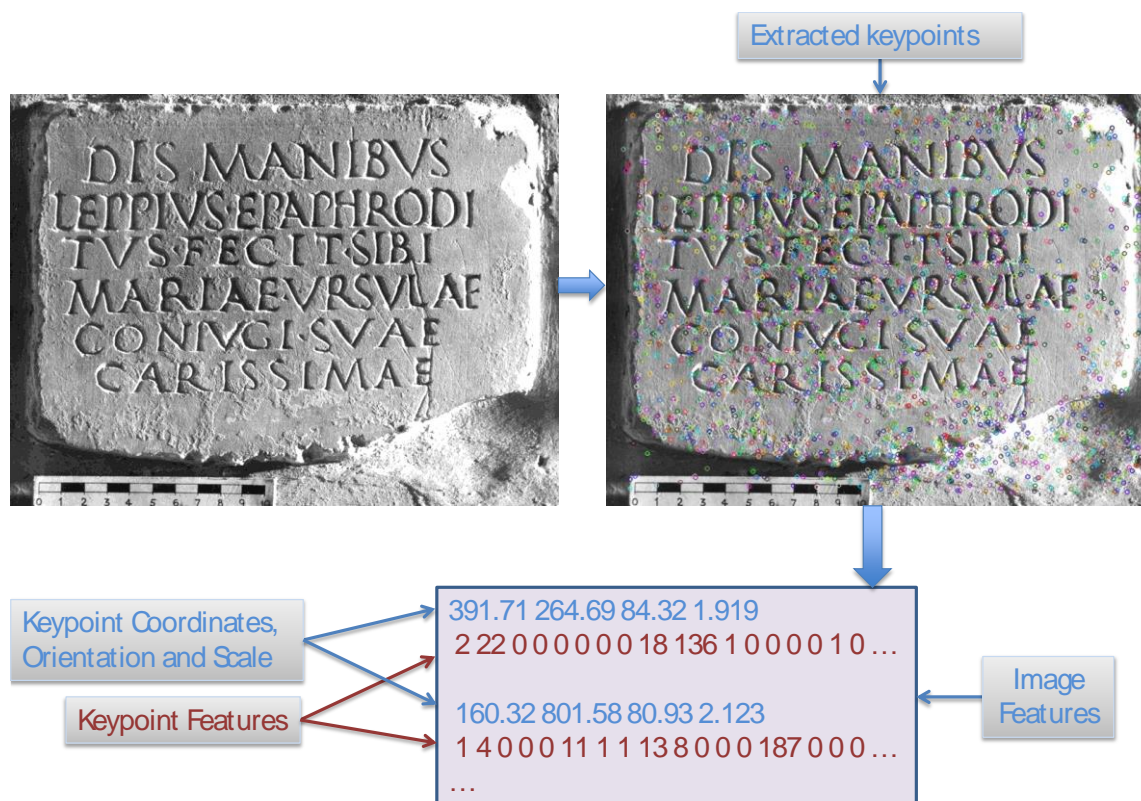


Figure 3-2 Image Features extraction

3.1.3 Image Indexer and Similarity Search

In order to efficiently and effectively execute the retrieval and recognition process, visual descriptors extracted from images have to be inserted in an index for efficient similarity search and ranking.

The Image Indexer component analyzes the visual descriptors and processes them for insertion in the CBIR index. State of the art technology in CBIR expects that the visual features of an image (global or local) are encoded into a text string, so that in the search and retrieval phase the mature technology of the text search engines can be exploited. The IRS will use two different approaches to generate a convenient text encoding for local and global visual features.

Local features will be encoded using an approach called "Bag of Features", where a textual vocabulary of visual words will be created starting from all the local descriptors of the whole dataset. The set of all the local descriptors of all the images is divided into a number of clusters (usually in the order of 100 thousand clusters) and to each cluster is assigned a (usually random) textual tag. The set of all tags becomes the "vocabulary" of visual words related to the whole set of images. At this point each image can be described by a set of "words" in the vocabulary (the textual tags), corresponding to the clusters containing the visual features of the image.

Global Features will be encoded using an approach called "perspective based space transformation". The idea at the basis of this technique is that when two global descriptors (GDs) are very similar (with respect to a given similarity function), they "view the world" around them in a similar way. To capture this "view" a set of distinguished images, called anchor descriptors (usually in the order of a few thousands), is defined (usually in a random fashion) and the similarity function is defined to represent a "distance" between two images. In this way it is possible to encode an image by first defining the order in which the

image “sees” the anchor descriptors, ranked by their distance to the image itself and then encoding with a textual string the specific permutation of the anchor descriptors.

In both cases it is then possible to build an index of “words” using the consolidated technologies of the Image Retrieval field, and to use also IR technologies for searching and ranking the results of a query. The Similarity Search will be based on the well-known “tf-idf” formula for weighting the “words” (visual features) within an image and on the cosine formula for ranking similar images. The tf-idf formula assigns a weight to each word in an image, and this weight is proportional to the number of occurrences of the word in the image (term frequency) and inversely proportional to the number of images in which this word appears (document frequency).

In this way each image is represented by a vector of weights and the similarity between an image in the data base and the query image (to which the same processing has been applied at query time) will be measured by the cosine of the angle between the two vector representations.

3.1.4 Image Recognizer

The Image Recognizer (IR) will be built starting from a collection of image sets provided to the IRS outside of the harvesting process. For each epigraph to be included in the recognizer what is needed is a set of images of the epigraph (training set) and an associated textual tag.

For each training set the recognizer will build a classifier that will be used to compute the probability that a query image belong to that set. The higher the number of images in a training set of a given epigraph, the best the performance of the classifier. The image training database will contain the visual descriptors of the images contained in the training sets of the selected epigraphs and will provide efficient retrieval functionality based on those descriptors.

The image recognizer, given a query image, uses the image training database to decide if the query image contains an epigraph that can be (reliably) associated with one of the training sets. The IR will use a recognition technology called “Single-label Distance-weighted kNN”¹². In other words, the recognizer will determine the k images that are closest to the query image (Nearest Neighbors), then will determine the training sets to which the nearest neighbors belong and finally, with mathematical algorithms based on weighted distances between the query image and the training sets, will determine the single label (the textual tag associated with a training set) to be returned as result of the query, provided that the confidence level is above a certain threshold.

Figure 3-3 shows an example of two image recognition training sets, which are basically a collection of labeled images of the same subject with different perspectives.

¹² See paper <http://dl.acm.org/citation.cfm?id=1862360>

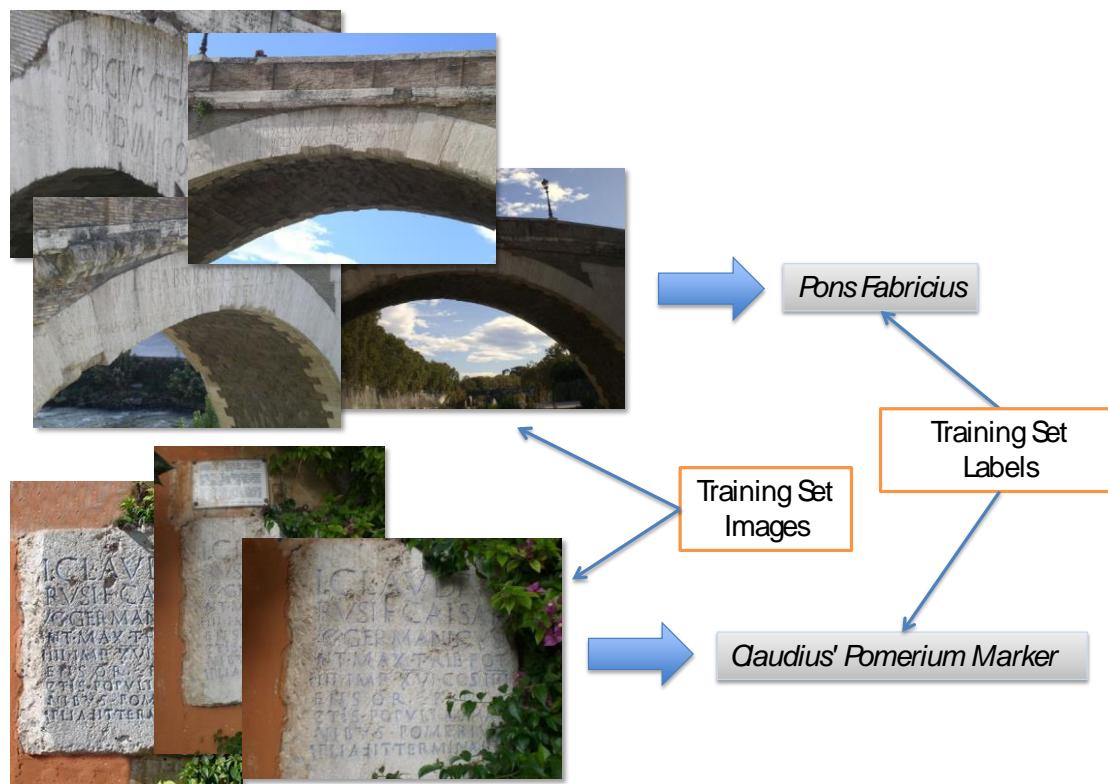


Figure 3-3 Image Recognizer training set example

Figure 3-4 shows a typical image recognition scenario, supporting the Flagship Mobile Application.

- The user takes a picture of an epigraph with the EAGLE Flagship Mobile Application.
- The mobile device sends the picture to the image recognition system.
- The system recognizes the picture as belonging to a training set and returns the label associated with that set.
- The mobile application, with the assistance of the EAGLE server (not shown in the picture) gets all the information associated with the recognized epigraph.

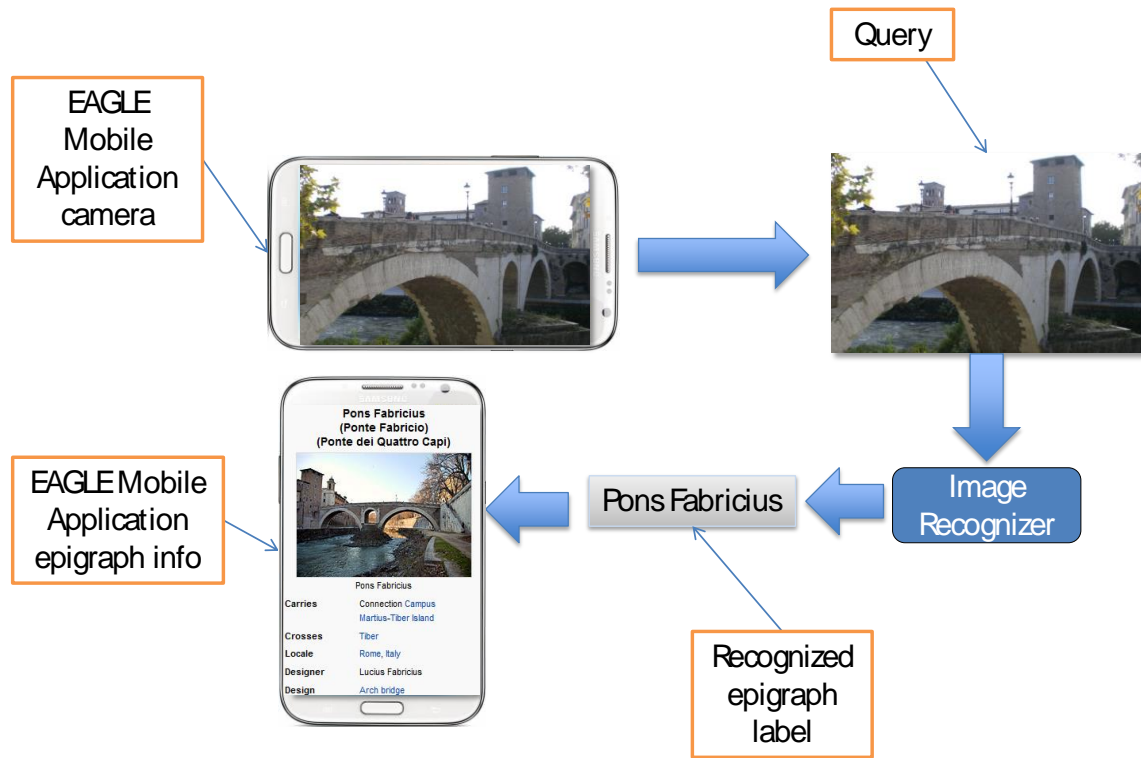


Figure 3-4 Image recognition typical scenario

3.2 IRS ARCHITECTURE

3.2.1 Image Retrieval Agent (IRA)

The Image Retrieval Agent is the component that manages the image ingestion process into the IRS. Functionally, The IRA is a daemon, i.e. a program that runs continuously as a background process. At each preset time interval it queries the Image Storage to check if new items are available and, if there are images available, it starts the image processing flow, depicted in Figure 3-5.

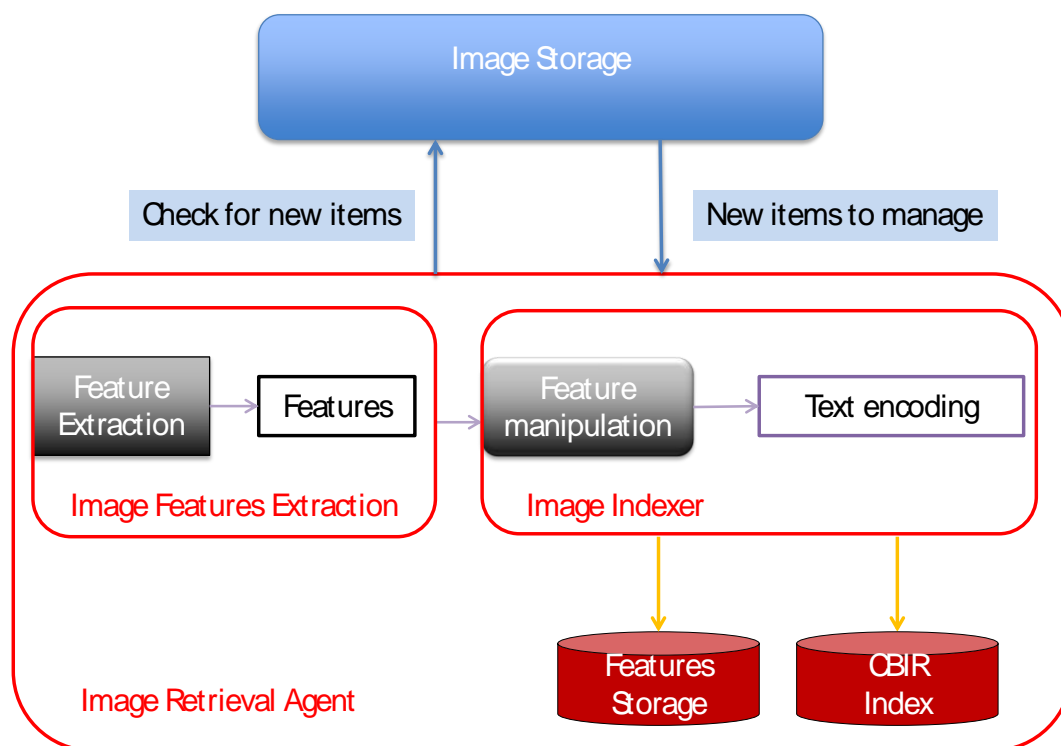


Figure 3-5 Image Retrieval Agent

The Image Storage component of the MAS provides API to retrieve the EAGLE images collected by the harvesting process.

During the insertion or updating process the IRA executes the following steps:

- calls the Image Storage API to retrieve new images to be inserted or updated;
- calls the Image Feature Extractor to get the mathematical description of the images (the image features);
- calls the Image Indexer module, that processes the features to an efficient indexing format (textual representation);
- call the indexing process to insert them in the image index and in the features storage database;
- flags the image on the Image Storage as inserted or updated.

During the deletion process the IRA executes the following steps:

- calls the Image Storage API to retrieve new images to be deleted;
- call the indexing process to delete the image features in the image index and in the features storage database;
- flags the image on the Image Storage as deleted.

The Image Retrieval Agent does not provide any external API to invoke its functions, as it is really an internal component of the EAGLE system and no external applications are allowed to use it.

3.2.2 Image Feature Extractor Component (IFE)

This is the component that given an image (to be added to the database or being provided as a query), analyzes its visual content to generate a visual descriptor. As described before, visual descriptors are mathematical description of the image. IFE can be thought of as a stand-alone application providing an API to receive an image and return its visual description in XML format. This API is not made visible from the outside, as the IFE will be called either by the Image Retrieval Agent for building and updating the index or by the Search and Recognition module that will provide a query image in order to get its visual features extracted.

The IFE is designed with a multi-threaded architecture for fast extraction of features and to take advantage of multicore processors. It has a plug-in architecture, so that it is easy to add or delete the mathematical libraries supporting the extraction algorithms of many different features, such as MPEG-7, CEDD¹³, SIFT¹⁴, SURF¹⁵, ORB¹⁶, etc..

3.2.3 Image Indexer

The Image indexer module will leverage the functionality of the Melampo CBIR System¹⁷. Melampo stands for *Multimedia enhancement for Lucene to advanced metric pivOting*. It is an open source CBIR library developed at CNR-ISTI that allows efficient searching of images by visual similarity through the use of Local and Global features. The visual features are transformed into strings of text suitable to be indexed and searched by a standard full-text search engine. The search engine used in the present implementation is the open-source Apache Lucene. The Melampo library also allows searching by combining visual similarity with the textual metadata associated with the images.

As described before, Local Features will be encoded using the Bag of Features approach. In the EAGLE project a specific vocabulary will be defined, based on the characteristics of the epigraph collections.

Global Features, as already stated, will be encoded based on a perspective space transformation. Figure 3-6 shows schematically the transformation process. Blue points represent anchor descriptors in a metric space (reference features), and the other colored points represent feature descriptors of images in the dataset. The figure also shows the encoding of the visual features in the transformed space and their representation in textual form (Surrogate Text Representation - STR). As can be seen intuitively, strings corresponding to images X and Y are more "similar" between them than those corresponding to images X and Z, as the textual strings capture in some way the distance between each image point and the reference features.

¹³ Color and Edge Directivity Descriptor.

¹⁴ Scale-Invariant Feature Transform.

¹⁵ Speeded Up Robust Features.

¹⁶ Oriented FAST and Rotated BRIEF

¹⁷ <https://github.com/claudiogennaro/Melampo>

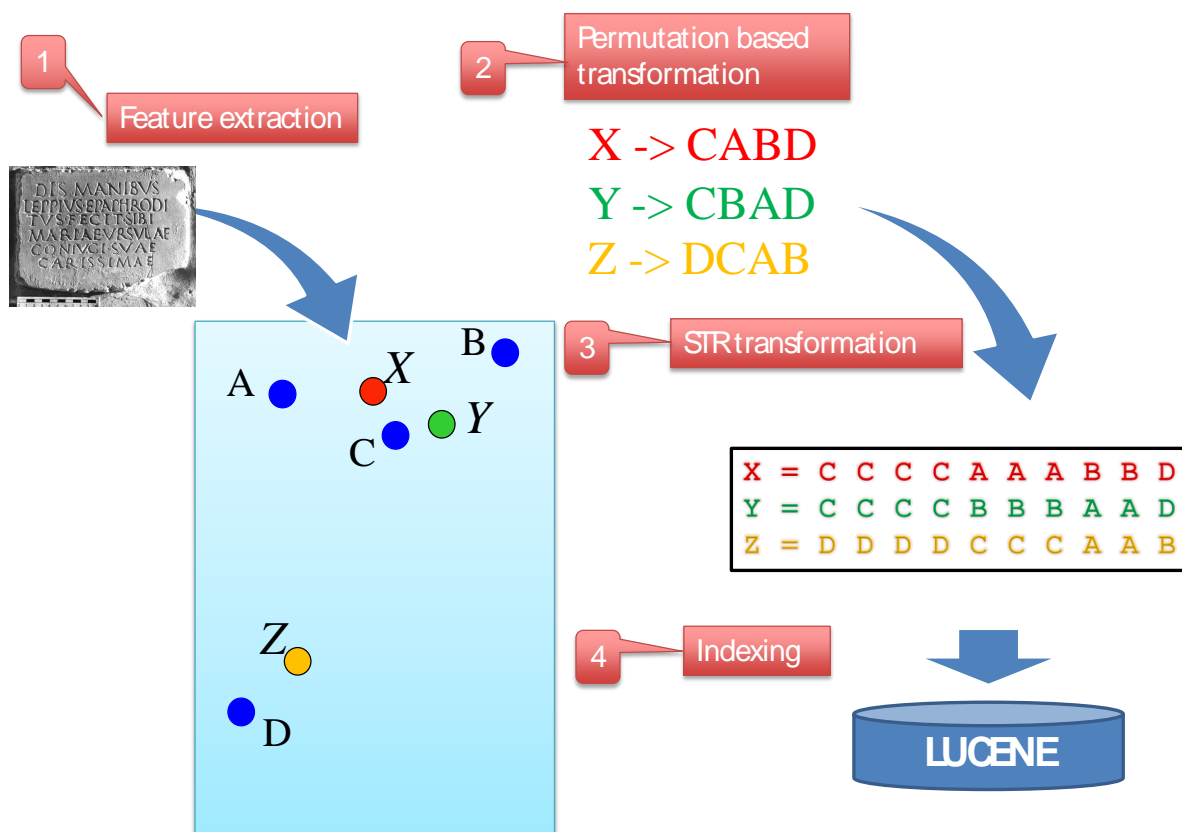


Figure 3-6 Example of perspective based space transformation and Surrogate Text Representation.

Figure 3.6. Legend

1. From the images we extract the features represented by points in a metric space. Blue points are reference features and colored points are data features.
2. The points are transformed into permutations of the references.
3. The permutations are transformed into text documents.
4. The text documents associated with the images are indexed.

3.2.4 Image Recognition API

As described in the preceding sections, the image recognizer, given a query image, uses the image training database to decide if the epigraph contained in the query image belongs to one of the stored training sets.

Briefly, given a collection of labeled training sets containing various images for each epigraph, the image recognizer assigns a label to a query in two steps. It first executes a kNN search among all the objects of all the training set. The result of such operation is a ranked list of labeled images belonging (possibly) to different training sets. By using the classifier built during the construction of the index, the label assigned to the query (the recognized epigraph) will be the one of the training set that maximizes the sum of the similarity between the query and the images in the kNN result list.

The Image Recognition Service will provide a REST API that will accept in input an image and will return either the label of the recognized epigraph or a "NOT IDENTIFIED" message. This API will be used by the

EAGLE user interface whenever it receives from a user (either through a browser or through the mobile application) an image to be recognized.

Figure 3-7 shows schematically the process described.

Starting from the query image, the recognizer service:

- Invokes the Image Feature Extractor to extract the visual features of the image;
- Invokes the Image Indexer providing the results of the preceding step; the indexer will transform the features into an effective text encoding and will perform an image similarity search on the index;
- Re-ranks the result of the similarity search from the preceding step to improve the result precision;
- Calls the image Classifier to evaluate the results and returns the identified epigraph, if the level of confidence determined by the classifier is above a certain threshold.

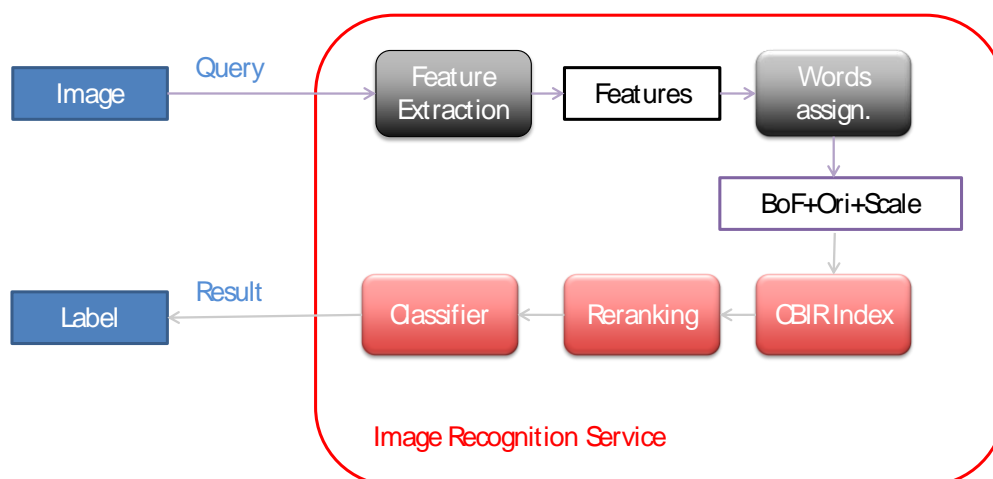


Figure 3-7 Image recognizer scheme

3.2.5 Similarity Search API

As described in the preceding sections, the Similarity Search components, given in input a query image, will return a list of images "similar" to the query image, ranked in decreasing order of similarity. This functionality will be provided through a REST API that will accept in input either an image or the ID of an image already in the CBIR index.

In the first case, the user will search similar images by uploading an image or by providing an URL to an image (this is known as Query by Example); in the second case the user will search similar images by providing the ID of an epigraph already in the index; this feature can be useful to find epigraphs similar to the one already obtained as the result of a metadata search. In either case the result will be a list of image IDs, ordered with respect to the decreasing value of the similarity.

Figure 3-8 and Figure 3-9 show the actions that will be performed during an image content-based search both in a Query by Example case and in an internal ID case.

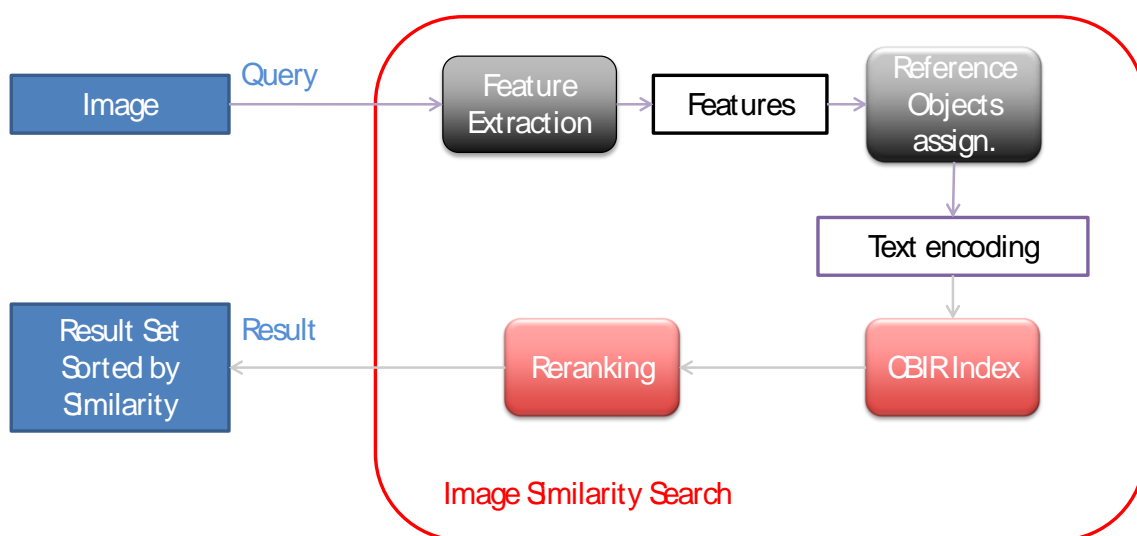


Figure 3-8 Query by Example scheme

The Image Similarity Service, after receiving a query image,

- calls the Feature Extraction component to get the image features
- converts the image features in a text encoding;
- uses the text encoding to query the CBIR index;
- to improve the search precision, reorders the index query results by a re-ranking tool;
- returns the re-ordered result set to the caller.

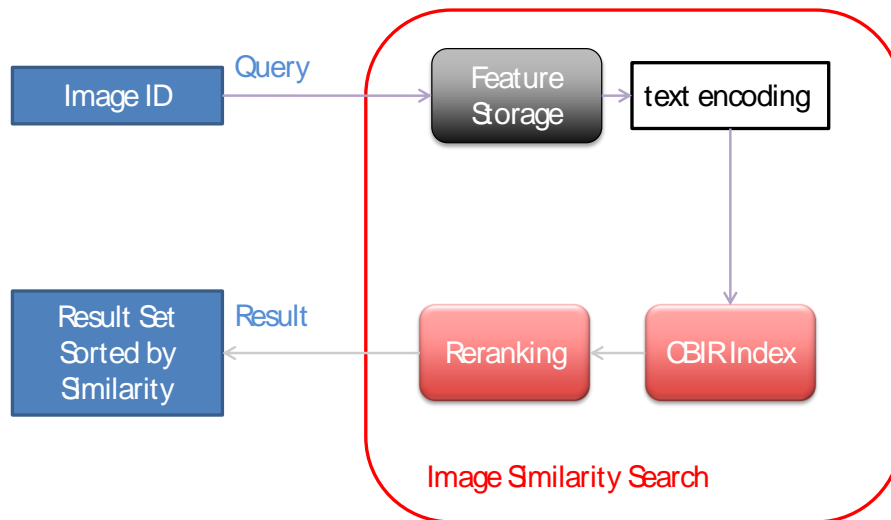


Figure 3-9 Query by internal ID scheme

In the Search by Image ID case, the steps taken are similar, except the first.

- calls the Feature Storage providing the image ID, in order to get the visual features of the image;
- converts the image features in a text encoding;
- uses the text encoding to query the CBIR index;
- to improve the search precision, reorders the index query results by a re-ranking tool;
- returns the reordered result set to the user.

4 CONCLUSIONS

The high-level architectural aspects of the Aggregation and Image Retrieval (AIM) system have been introduced first, highlighting the main functional components and the different types of users which are foreseen to interact with the AIM in the wider scope of the whole EAGLE project.

The system will support different types of user with distinct roles and permissions. The types currently foreseen to be supported are i) Content Provider, ii) Data Curator, iii) Administrator, and iv) End-User. The system will consequently supply authentication and access functions according to such roles.

The Metadata Aggregation System (MAS) and the Image Retrieval System (IRS) - the two complementary components of the AIM - are then described in detail, showing both the basic functionalities and the specific architecture and customisation that will be adopted to implement the AIM in the context of the EAGLE project.

The description of the MAS has been undertaken from both a Content Provider's Perspective, a Data Curator's perspective, and a Administrator's Perspective.

The description of the IRS has provided first an overview of the functionality that it supports, and then an architectural description showing how the different components of the IRS interact with each other, with the MAS and with the user interface of the EAGLE portal. It has to be noted that the only external APIs of the IRS are the ones related to the Similarity Search and the Image Recognition. .