

## Project

<b>Project Acronym:</b>	AthenaPlus
<b>Grant Agreement number:</b>	325098
<b>Project Title:</b>	Access to cultural heritage networks for Europeana

## Deliverable

<b>Deliverable name:</b>	<b>Final report on the platform for metadata aggregation and delivery to Europeana</b>
<b>Deliverable number:</b>	<b>D3.3</b>
<b>Delivery date:</b>	October 2015
<b>Dissemination level:</b>	Public
<b>Status</b>	Final version
<b>Authors (organisation)</b>	Nikolaos Simou (NTUA)
<b>Contributors (organisation)</b>	Regine Stein (UNIMAR)
<b>Reviewers (organisation)</b>	Marzia Piccininno (ICCU) Erik Buelinckx (KIK-IRPA)

## Revision History

Revision	Date	Author	Organisation	Description
V0.1	2015-10-05	Nikolaos Simou	NTUA	Table of contents
V0.2	2015-10-20	Nikolaos Simou	NTUA	First Draft
V0.3	2015-10-25	Nikolaos Simou	NTUA	Minor Corrections - Sent for Internal Review
V0.4	2015-10-30	Nikolaos Simou	NTUA	Final version
V0.5	2015-10-30	Maria Teresa Natale	ICCU	Formal check

### 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This deliverable contains original unpublished work except where clearly indicated otherwise.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

1	EXECUTIVE SUMMARY .....	2
2	INTRODUCTION .....	4
2.1	Background.....	4
2.2	Role of this Deliverable in the Project.....	5
2.3	Approach .....	6
2.4	Structure of the document .....	6
3	COMPLIANCE WITH CH METADATA MODELLING REQUIREMENTS .....	7
3.1	LIDO to EDM transformation mechanism.....	7
3.2	An EDM profile based on LIDO .....	11
4	HELPDESK SUPPORT SYSTEM.....	18
5	METADATA INGESTION.....	20
5.1	<i>MINT instance for producing high quality LIDO metadata</i> .....	20
6	DELIVERY TO EUROPEANA.....	23
7	QUALITY EVALUATION ON LIDO METADATA.....	27
7.1	Selection of LIDO metadata for the evaluations .....	27
7.2	Quality Evaluation.....	27
8	CONCLUSION .....	32
8.1	Results.....	32
8.2	Impact.....	32
9	APPENDIX 1: DEFINITION OF TERMS AND ABBREVIATIONS.....	33
10	APPENDIX 2: LIDO TO EDM XSLT .....	34

## 1 EXECUTIVE SUMMARY

The main objective of this deliverable is the presentation of the platform for metadata aggregation and delivery to Europeana as this has been evolved within the AthenaPlus project. In detail this deliverable outlines all the activities performed that are related to the WP3 tasks that are:

- Task 3.1: Compliance with CH metadata modelling requirements.
- Task 3.2: Help desk support system.
- Task 3.3: Metadata Ingestion.
- Task 3.4: Delivery to Europeana.

There are two main activities related to Task 3.1 “Compliance with CH metadata modelling requirements”. First the transformation of valid LIDO records produced through MINT by the content providers to valid EDM records that are delivered to Europeana. This transformation is performed through MINT by the use of a crosswalk – an XSLT file – that appropriately maps the LIDO elements to the corresponding EDM elements of the same semantics. Since this transformation has been vital for one of the project’s main objectives – the publication on Europeana – a first version of this XSLT has been implemented from the quite early days of the project. However, this version was not fully exploiting the expressive power of the less expressive data model between the two – that is EDM – thus the first version was not using all the EDM elements. This limitation has been corrected and the current version – that is fully reported in section 3 – supports all the EDM elements offering in that way a high quality LIDO to EDM transformation mechanism exploiting the full expressiveness of both models.

The second activity that falls in this task that has been performed towards the compliance with CH metadata modelling requirements is the modelling of an EDM profile based on LIDO. The need for this activity was born within the project after the mapping of LIDO to EDM and wasn’t among the original project objectives. In short an analysis has been made on the EDM properties and classes that can be directly used, identifying the properties and classes missing and those that need to be defined from scratch or to be discovered among existing ontologies.

Section 4 refers to the activities performed for the second task of this WP that is the help desk support system. The main objective of this task was the provision of technical assistance to the content providers through a mailing list. A short discussion of the main issues tackled through the list is given, including mainly import and mapping difficulties as well as comments Europeana made on delivered metadata. A short statistical analysis of the issues sent and resolved through the list is also given, outlining a very good overall operation that made provider’s lives much easier.

The section that follows (5) is dedicated to the ingestion platform describing in detail the core developments done fulfilling the project requirements that namely are:

- Set up of MINT Instances based on the AthenaPlus metadata quality requirements together with their interconnected OAI-PMH repositories that have been mainly used by the AthenaPlus content providers for updating metadata previously published for the Athena and Linked Heritage projects.
- Set up a MINT Instance based on the requirements for visualization and indexing of museum content using LIDO.

Section 6 presents the updated publication method employed in MINT. The main limitation of the previous method was the fact it only allowed the publication of metadata coming from one schema through MINT. This schema was EDM – since this is the schema supported by Europeana for publication – and for each EDM record a corresponding OAI-DC record was produced. The updated publication method allows:

- The OAI publication of more than one schema through MINT. More specifically LIDO and EDM metadata are now sent to the OAI server, using different OAI namespaces making in that way the LIDO records produced by the content providers consumable through the OAI-PMH protocol.
- The definition of the publication schema that will be used for the creation of the OAI\_DC record.
- The definition of the OAI namespace will be used for different kinds of metadata.

Finally section 7 provides an analysis of the LIDO metadata produced through the MINT tool in terms of expressivity. More specifically, the analysis examines the LIDO elements and values used, putting special

effort on the Events and vocabularies. In addition, since the need of high quality and expressive metadata has been recognised by Europeana by the creation of the metadata quality taskforce, a task group with similar objectives to the one created by Europeana but focused on the LIDO metadata standard instead of the EDM has been established within the project consortium. This task group has authored guidelines – through the use of specific LIDO elements together with specific values – for the production, visualization and indexing of museum content using LIDO. These guidelines have been used for the configuration of a different MINT instance that has been set up especially for this purpose and several providers volunteered to follow them in producing 100 high quality metadata records. A statistical analysis is provided for these records together with a comparison between the two sets.

## 2 INTRODUCTION

This deliverable describes all the activities performed within WP3 for fulfilling the project's main objectives that were:

- The implementation and integration of the technical components necessary for the ingestion of content into Europeana.
- The delivery of a web-application with a user-friendly interface on top of the integrated technical components that would allow content providers to produce, validate and publish metadata on Europeana.

These components constitute the so-called Metadata INTERoperability services (MINT). The development of which started in the ATHENA<sup>1</sup> project when the NTUA team integrated all the necessary components for publishing metadata to Europeana into a common technology platform, while it evolved through its use in other Europeana-feeder projects like Linked Heritage<sup>2</sup>, EuScreen<sup>3</sup>, ECLAP<sup>4</sup>, Fashion<sup>5</sup>, EuropeanaPhotography<sup>6</sup>. The main reason why MINT has become so successful and popular is the provision of a user friendly interface that acts as a visualization for the XSLT language. In that way MINT makes the involvement of the content providers – that in most cases do not have any technical background and only know well their in-house metadata model – into the mapping process possible. Moreover, MINT is a powerful tool to monitor the progresses of each provider and the overall content aggregation for Europeana.

Every of the afore-mentioned projects was composed by a different consortium that was mainly based on the content that it had to publish on Europeana. This resulted into different metadata requirements and thus different customizations on MINT built around them. This deliverable focus on the customizations that have been performed for accomplishing the targets and needs of the Museum community represented through the AthenaPlus project.

### 2.1 Background

The main role of the MINT ingestion platform in the AthenaPlus project is to enable users to

- Provide metadata records in a range of “source” formats.
- Convert metadata to LIDO (that is used as an intermediate standard).
- Transform LIDO metadata to the latest version of EDM.
- Publish metadata to Europeana.
- Monitor the progresses of content provision.

Its key functionalities include:

- Organization and user level access rights and role assignment.
- Collection and record management (XML serialisation).
- Direct import and validation according to registered schemas (XSD).
- OAI-PMH based harvesting and publishing.
- Visual mapping editor for the XSLT language.
- Transformation and previewing (XML and HTML).
- Repository deployment and remediation interfaces.

The typical workflow employed through MINT in the AthenaPlus is shown in the figure below.

---

<sup>1</sup> <http://www.athenaeurope.org/>

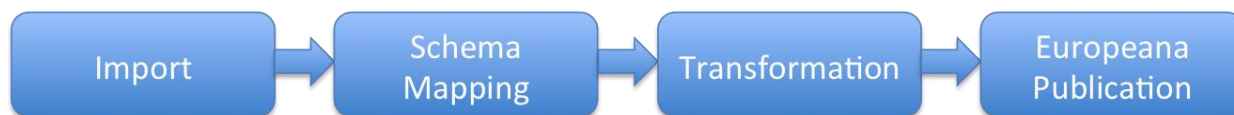
<sup>2</sup> <http://www.linkedheritage.eu/>

<sup>3</sup> <http://euscreen.eu/>

<sup>4</sup> <http://www.eclap.eu/drupal/?q=en-US>

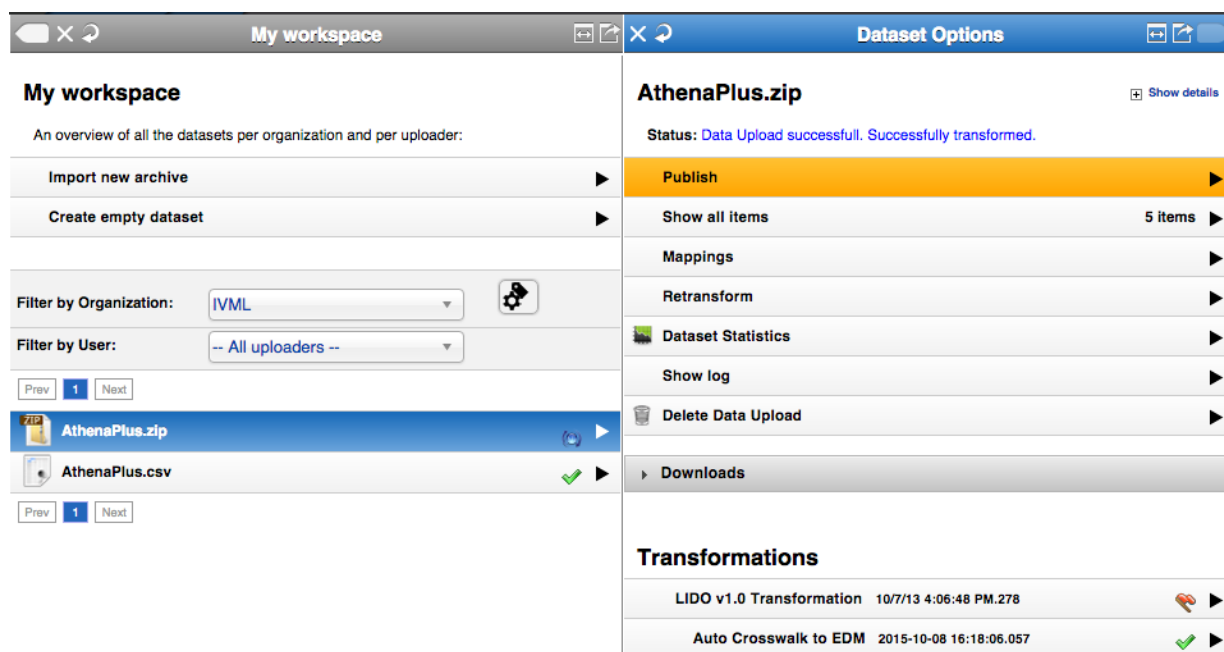
<sup>5</sup> <http://www.europeanafashion.eu/>

<sup>6</sup> <http://www.europeana-photography.eu/>



**Figure 1: Overall workflow**

A content provider performs an export of his metadata from his own content management system. Then this set of records – called dataset – is imported into MINT. After that using the user-friendly interface of MINT he performs a mapping – i.e. an alignment of the information held in his own content management system to the corresponding elements of LIDO that acts as the intermediate schema in the project. Once this is done he can check the validity of the mapping through a set of preview interfaces that give him inline schema validation and also a view of how his records would look like when published on Europeana. Once these checks are performed he transforms his metadata to LIDO creating a new dataset. After that, an additional transformation is performed by the MINT tool; the transformation of LIDO records to EDM. Again this process results into a new EDM dataset and once this is ready the content provider can sent his metadata to NTUA’s OAI-PMH server (see figure 2). This action is called publication (Publish button) within the MINT environment, because is the last action a content provider should perform before seeing his metadata published on Europeana’s portal. Typically after this action WP2 takes over by informing Europeana to harvest from NTUA’s repository and once the metadata are thoughtfully examined, they appear on its portal.



**Figure 2: A transformed dataset with the derived LIDO and EDM datasets**

An extended description of the MINT mapping tool, its functionalities and the customizations made for the AthenaPlus version is given in D 3.1 “The MINT ingestion platform”. In addition D 3.2 “Description of the LIDO to EDM” provides the technical details of the first version of the XSLT file used for the transformation of valid LIDO records to semantically aligned and valid EDM files.

## 2.2 Role of this Deliverable in the Project

This deliverable reports on all the activities performed towards the project objectives that correspond to WP3 tasks:

- Task 3.1: Compliance with CH metadata modelling requirements.
- Task 3.2: Help desk support system.
- Task 3.3: Metadata Ingestion.
- Task 3.4: Delivery to Europeana.

More specifically this deliverable covers the gap between the previous reports of this WP (D 3.1 and D 3.2) and the developments performed from the period these have been delivered till the end of the project. All the activities reported in this deliverable contributed to the achievement of the following milestones:

- MS4 10-20% content uploaded in MINT
- MS5 60-70% content uploaded in MINT
- MS6 100% content uploaded in MINT
- MS7 MINT/AthenaPlus ready for delivering content to Europeana

## 2.3 Approach

The approach employed by NTUA for each of the activities performed within WP3 is described in detail in the sections that follow. In general the main priority of NTUA in all the activities performed was to carefully analyze all the parameters of each action, especially when these had an impact on the work performed by the content providers and the Europeana publication. Additionally the technologies used for the development of MINT have been selected having as main criteria the stability and scalability that were required for the needs of probably the biggest Europeana feeder project that successfully contributed **3.5 millions** records to Europeana.

## 2.4 Structure of the document

This deliverable presents in detail the activities performed for all the WP3 tasks. Section 3 outlines task 3.1. "Compliance with CH metadata modelling requirements" including separate sub sections for each of the main activities related to it that namely are the LIDO to EDM crosswalk and the EDM profile based on LIDO. Similarly Sections 4 to 6 report on "Help desk support system", "Metadata Ingestion" and "Delivery to Europeana" respectively. Section 7 details the quality evaluation of LIDO metadata. Finally, Appendix 1 contains the abbreviations used in this document while the final version of the LIDO to EDM XSLT is provided in Appendix 2.

### 3 COMPLIANCE WITH CH METADATA MODELLING REQUIREMENTS

This section describes in detail the activities performed for the first task of WP3 the main objective of which was the compliance with the Europeana Data Model (EDM) – the standard developed by Europeana for aggregating rich metadata from a wide range of data providers including libraries, museums, archives and audio-visual sectors. Since EDM is complex and a standard that constantly evolves this task ensured the compliant of MINT tool with EDM and its updates. Towards this objective a mapping of LIDO to EDM exploiting the full expressiveness of both models has been implemented that is the main outcome for this task.

As also presented in D 3.2 – that is dedicated to this activity – the mapping from one model to the other is not a trivial task. The main difficulty is that EDM is an ontology – despite the fact that a harvesting schema for it exists – the instances of which have to fulfil certain criteria, like being consistent to domain and range restrictions while LIDO is an XSD schema having different validation rules for the instances. Therefore, the main difficulty encountered was the transition from a tree structured data model such as LIDO to a graph – RDF structured model such as EDM. Additionally, the expressiveness of the two data models is not the same. LIDO can be considered as a more expressive schema since it uses some complex types/elements having semantics that either cannot be found in EDM (e.g. actorInRole) or are not currently supported (e.g. Event).

There are two approaches that can be followed for mapping LIDO records to EDM. The first one employed into the MINT mapping tool makes use of an XSLT file that transforms a valid LIDO instance to a valid EDM instance. The second and most flexible – expressive approach, that has been explored but not fully developed is the creation of an EDM – LIDO profile.

#### 3.1 LIDO to EDM transformation mechanism

MINT has been built around XML and it mainly uses XML technologies. The imports are either directly XML files or CSV files that are automatically converted to XML, the mapping made through MINT’s user interface is compiled as an XSLT file that is used for the transformation, while the validation is performed according to the XSD constraints (e.g. cardinality, ordering and value restriction constraints). Therefore, since the LIDO to EDM transformation mechanism had to be integrated into the overall metadata production workflow the implementation of an XSLT file for this process has been a rational solution and it has been implemented from the early days of the project.

Section 4.1 of D 3.2 presented the mapping approach and the mapping choices made that can shortly summarized to:

1. The use of the `lido:recordID` element of LIDO for the creation of the identifiers required for the `edm:ProvidedCHO` and `ore:Aggregation` classes as well as for the relation between them as shown in the figure below.



Figure 3: EDM basic structure

So the identifier of the ProvidedCHO is

```
http://mint-projects.image.ntua.gr/Athena_Plus/ProvidedCHO/+dataProvider+/
lido:recordID
```

that is also the value of the EDM xpath,



rdf:RDF/ore:Aggregation/edm:aggregatedCHO@rdf:resource

while the identifier of the respective instance of type ore:Aggregation is

[http://mint-projects.image.ntua.gr/Athena\\_Plus/Aggregation/+dataProvider+/lido:recordID](http://mint-projects.image.ntua.gr/Athena_Plus/Aggregation/+dataProvider+/lido:recordID)

2. The LIDO bookmarks used in the MINT mapping tool to assist providers during the mapping process.
3. The use of language tags and the exploitation of multilingual information when this is possible.

Although this approach covered most of the mapping cases, it didn't include all the EDM elements of the Europeana harvesting schema<sup>7</sup>. More specifically the properties of EDM that are used for expressing relations between objects such as `edm:isNextInSequence`, `edm:isRelated` and others have not been originally used. LIDO uses the XML complex type `lido:objectRelationWrap` for the relations and the type of the relation – that can vary – is specified in the `lido:relatedWorkRelType`. The following table presents the EDM properties used for the description of an `edm:ProvidedCHO` that were not included in the initial XSLT version presented in D 3.2 together with the corresponding LIDO xpaths. What is highlighted in bold is the main condition that needs to be satisfied in the LIDO record for the mapping to be performed.

**Table 1: Mapping of LIDO elements to EDM properties**

EDM Xpath	LIDO Xpath
<code>edm:ProvidedCHO/dc:coverage</code>	From the LIDO Xpath it can always be distinguished into the more specific subproperties <code>dcterms:spatial</code> or <code>dcterms:temporal</code> , see therefore the EDM Xpaths <code>edm:ProvidedCHO/dcterms:spatial</code> and <code>edm:ProvidedCHO/dcterms:temporal</code>
<code>edm:ProvidedCHO/dc:relation</code>	<code>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</code> <b>Condition</b> <code>lido:relatedWorkRelType/lido:conceptID='http://purl.org/dc/elements/1.1/relation'</code> and ( <code>lido:relatedWork/lido:object/lido:objectWebResource</code> or <code>lido:relatedWork/lido:object/lido:objectID</code> or <code>lido:relatedWork/lido:object/lido:objectNote</code> )
<code>edm:ProvidedCHO/dc:source</code>	<code>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</code> <b>Condition</b> <code>lido:relatedWorkRelType/lido:conceptID='http://purl.org/dc/elements/1.1/source'</code> and ( <code>lido:relatedWork/lido:object/lido:objectWebResource</code> or <code>lido:relatedWork/lido:object/lido:objectID</code> or <code>lido:relatedWork/lido:object/lido:objectNote</code> )
<code>edm:ProvidedCHO/dcterms:conformsTo</code>	<code>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</code> <b>Condition</b> <code>lido:relatedWorkRelType/lido:conceptID='http://purl.org/dc/terms/conformsTo'</code> and ( <code>lido:relatedWork/lido:object/lido:objectWebResource</code> or

<sup>7</sup> The reason why we explicitly mention the elements of the EDM harvesting schema at this case is because the majority of the EDM properties as mentioned in the EDM specification are not used in practice for the aggregation process since they are not included in the EDM harvesting schema.

	<p>lido:relatedWork/lido:object/lido:objectID lido:relatedWork/lido:object/lido:objectNote)</p> <p style="text-align: right;">or</p>
edm:ProvidedC HO/dcterms:hasFormat	<p>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</p> <p><b>Condition</b></p> <p>lido:relatedWorkRelType/lido:conceptID= <a href="http://purl.org/dc/terms/hasFormat">http://purl.org/dc/terms/hasFormat</a> ' and (lido:relatedWork/lido:object/lido:objectWebResource or lido:relatedWork/lido:object/lido:objectID or lido:relatedWork/lido:object/lido:objectNote)</p>
edm:ProvidedC HO/dcterms:hasPart	<p>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</p> <p><b>Condition</b></p> <p>lido:relatedWorkRelType/lido:conceptID= <a href="http://purl.org/dc/terms/hasPart">http://purl.org/dc/terms/hasPart</a> ' and (lido:relatedWork/lido:object/lido:objectWebResource or lido:relatedWork/lido:object/lido:objectID or lido:relatedWork/lido:object/lido:objectNote)</p>
edm:ProvidedC HO/dcterms:references	<p>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</p> <p><b>Condition</b></p> <p>lido:relatedWorkRelType/lido:conceptID= <a href="http://purl.org/dc/terms/references">http://purl.org/dc/terms/references</a> ' and (lido:relatedWork/lido:object/lido:objectWebResource or lido:relatedWork/lido:object/lido:objectID or lido:relatedWork/lido:object/lido:objectNote)</p>
edm:ProvidedC HO/dcterms:tableOfContents	<p>lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:objectDescriptionWrap/lido:objectDescriptionSet [<b>@lido:type='tableOfContents'</b>]/lido:descriptiveNoteValue</p>
edm:ProvidedC HO/dcterms:temporal	<p>lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:subject/lido:subjectDate</p> <p><b>Condition</b></p> <p>lido:displayDate or (lido:date/lido:earliestDate and lido:date/lido:earliestDate)</p>
edm:ProvidedC HO/edm:currentLocation	<p>lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:repositoryWrap/lido:repositorySet [<b>@lido:type='current'</b>]/lido:repositoryLocation/lido:placeID [<b>@lido:type='URI'</b>]</p>
edm:ProvidedC HO/edm:hasMet	<p>From the LIDO Xpath it can always be distinguished into the more specific subproperties of edm:hasMet, e.g. dc:contributor, dc:creator, dc:date and their subproperties.</p>
edm:ProvidedC HO/edm:hasType	<p>From the LIDO Xpath it can always be distinguished into the more specific subproperties dc:format and dc:type.</p>
edm:ProvidedC HO/edm:incorporates	<p>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</p> <p><b>Condition</b></p> <p>lido:relatedWorkRelType/lido:conceptID= <a href="http://www.europeana.eu/schemas/edm/incorporates">http://www.europeana.eu/schemas/edm/incorporates</a> ' and (lido:relatedWork/lido:object/lido:objectWebResource or</p>

	<pre>lido:relatedWork/lido:object/lido:objectID lido:relatedWork/lido:object/lido:objectNote)</pre>	or
edm:ProvidedCHO/edm:isDerivativeOf	<pre>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</pre> <p><b>Condition</b></p> <pre>lido:relatedWorkRelType/lido:conceptID= http://www.europeana.eu/schemas/edm/isNextInSequence'</pre>	and or or
edm:ProvidedCHO/edm:isNextInSequence	<pre>lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet</pre> <p><b>Condition</b></p> <pre>lido:relatedWorkRelType/lido:conceptID= http://www.europeana.eu/schemas/edm/isNextInSequence'</pre>	and or or

Apart from the mapping of the EDM properties that were not there in the initial version, short corrections and updates have been also performed that include the following:

- Spaces have been normalized from the `dc:subject` and `dc:description` values.
- The occurrence of `edm:object` has been set to 1. The problem was the cardinality of the LIDO element mapped to `edm:object`. In detail, its cardinality is 1 or many, resulting to 1 or many values for the `edm:object`. However, the cardinality of `edm:object` in the EDM schema is 1, hence this XSLT had to be updated accordingly for not producing invalid EDM files.
- The Europeana Photography vocabulary has been included into the XSL as a variable. In that way the dereferencing of the vocabulary terms – that was a bit problematic in Europeana – is handled on MINT's side.
- The mapping of the `lido:subjectPlace` has been updated. The values of this element ended up as values of the `dc:subject` property while they are now mapped to `dcterms:spatial` property.
- The XSLT has been updated to accept ftp urls for the properties `edm:isShownBy` and `edm:hasView`.
- The mapping of multilingual values for elements `dcterms:medium`, `dc:creator`, `dc:publisher` has been corrected.
- The `edm:isNextInSequence` property can be used for the `WebResource` instances. This allows modeling the data in a way that enables an end user to browse cultural objects in the appropriate context (in this case from one entry point that is the record the user has access to all digital resources available for that given object) something that Europeana recommends and considers as an effort for providing the best data quality possible. An example of such a record is shown below

[http://www.europeana.eu/portal/record/9200231/BibliographicResource\\_2000092036280.html](http://www.europeana.eu/portal/record/9200231/BibliographicResource_2000092036280.html)

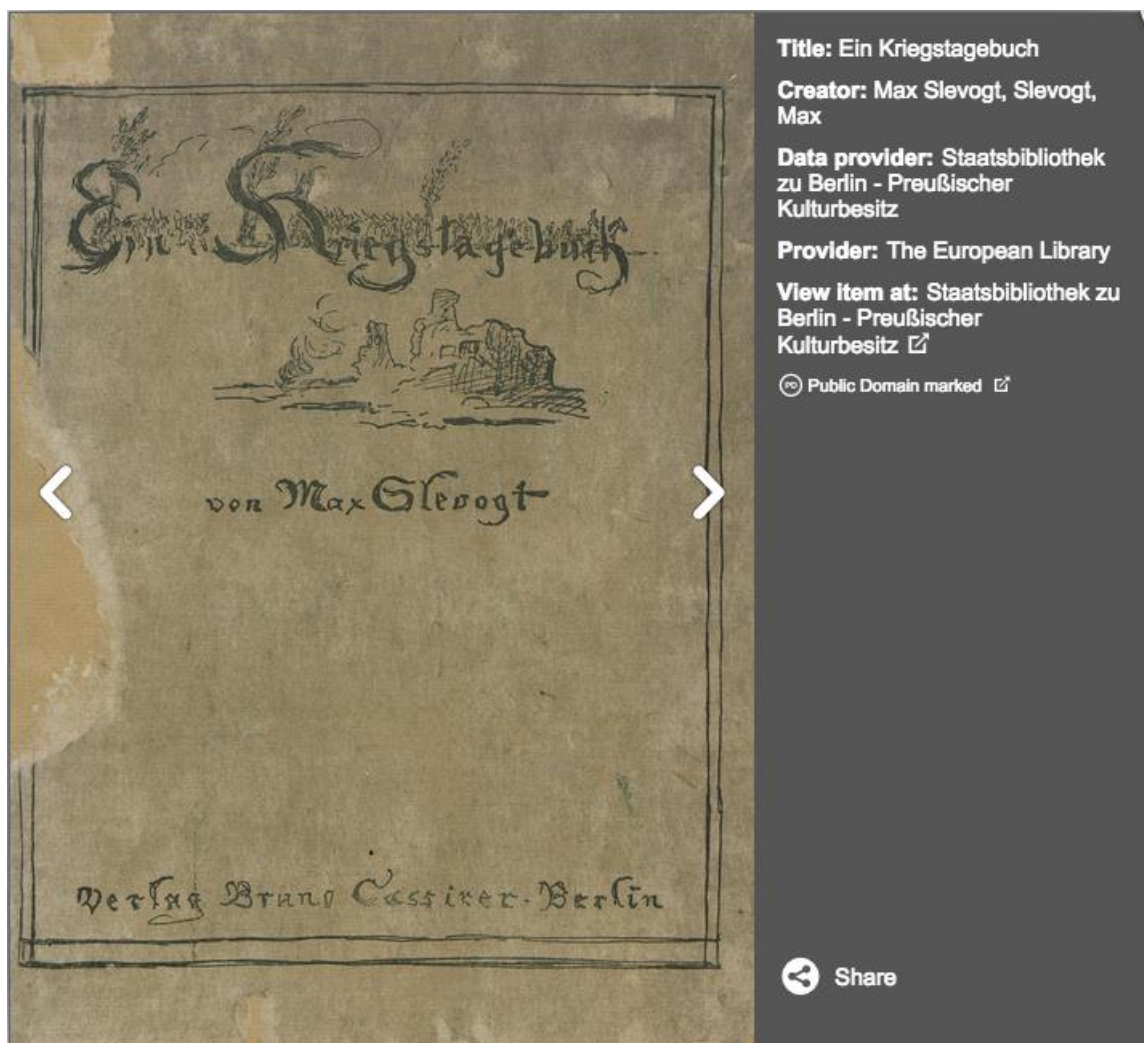


Figure 4: An object with many digital resources as shown in Europeana

The final XSLT for the conversion of valid LIDO files to valid EDM files is available at

<https://svn.image.ece.ntua.gr/repos/lido2edm/>

while an excerpt including the core mappings can be found at Appendix 2.

### 3.2 An EDM profile based on LIDO

A better way of tackling the expressiveness limitations of EDM and representing LIDO information about a digitised cultural object in EDM is the use of an EDM profile based on LIDO. This implies that the basic classes and properties of EDM remain in place and whenever they are not sufficient for holding the information a LIDO record can hold, a new class or property is either discovered among existing namespaces or created in a new namespace. In other words the basic relation of an `edm:ProvidedCHO` and an `ore:Aggregation` remains, but more specialized classes of `edm:ProvidedCHO` like may be implemented that will be described by specific properties like e.g. `lido:culture` instead of the abstract `dc` and `dcterms` properties that are mainly offered by EDM.

Section 4.2 of D 3.2 presented the main reasons why an EDM LIDO profile should be implemented that are shortly summarized below:

- Representation of a relation's type between a pair of objects in EDM. Any information about the relation's type between objects or works can hardly be modelled with the use of EDM. Particularly,

EDM defines specific kinds of relations, like “is successor of”, “is similar to”, “has part”, “incorporates”, between the described objects/works, or the general relation “ens:isRelatedTo”, omitting specific types of relation like “larger context for”, “model of”, “model for”, “study of”, “study for”, “rendering of”, “copy of” that LIDO anticipates as possible types.

- Actor’s role participating in an event. Unless the actor is a creator of the work at hand, or its publisher, his/her role cannot be specified, but all different kinds of contribution go to the contributor. Similarly different dates related to an `edm:ProvidedCHO` cannot be qualified.

Since then a close examination of the EDM classes and properties has been performed. In detail the EDM classes that will be used in the EDM-LIDO profile have been analysed together with the properties that are used to describe them. Our examination for the classes used in the EDM-LIDO profile is shown in the following table. As it can be seen there are classes like the `edm:Event` and the `edm:PhysicalThing` that despite the fact that exist in the EDM specification they are not implemented in the EDM harvesting schema and thus they are not supported by Europeana yet.

**Table 2: EDM classes to be used in EDM-LIDO profile**

Class Name	URI	EDM Class / as of 30/10/2015 implemented in Europeana	Used for delivery to Europeana	Used for LIDO profile	Mandatory
ore:Aggregation	<a href="http://www.openarchives.org/ore/terms/Aggregation">http://www.openarchives.org/ore/terms/Aggregation</a>	yes / yes	yes	yes	yes
edm:WebResource	<a href="http://www.europeana.eu/schemas/edm/WebResource">http://www.europeana.eu/schemas/edm/WebResource</a>	yes / yes	yes	yes	yes
edm:ProvidedCHO	<a href="http://www.europeana.eu/schemas/edm/ProvidedCHO">http://www.europeana.eu/schemas/edm/ProvidedCHO</a>	yes / yes	yes	yes	yes
edm:Agent	<a href="http://www.europeana.eu/schemas/edm/Agent">http://www.europeana.eu/schemas/edm/Agent</a>	yes / yes	yes	yes	no
edm:Place	<a href="http://www.europeana.eu/schemas/edm/Place">http://www.europeana.eu/schemas/edm/Place</a>	yes / yes	yes	yes	no
edm:TimeSpan	<a href="http://www.europeana.eu/schemas/edm/TimeSpan">http://www.europeana.eu/schemas/edm/TimeSpan</a>	yes / yes	no	yes	no
skos:Concept	<a href="http://www.w3.org/2004/02/skos/core#Concept">http://www.w3.org/2004/02/skos/core#Concept</a>	yes / yes	yes	yes	no
edm:Event	<a href="http://www.europeana.eu/schemas/edm/Event">http://www.europeana.eu/schemas/edm/Event</a>	yes / no	no	yes	no
edm:PhysicalThing	<a href="http://www.europeana.eu/schemas/edm/PhysicalThing">http://www.europeana.eu/schemas/edm/PhysicalThing</a>	yes / no	no	yes	no

Then the properties that are used to describe these classes have been thoughtfully examined, this time examining not only their support status by Europeana but also the corresponding property that could be used in the EDM-LIDO profile. The two tables below present our analysis for the two basic EDM classes that are the edm:ProvidedCHO and ore:Aggregation. It is important at this point to mention that our observation is that many of the edm properties that are used for describing an edm:ProvidedCHO should be event-mediated in the EDM-LIDO profile and therefore would require the implementation of the edm:Event class.

**Table 3: Properties of edm:ProvidedCHO**

Property Name	EDM Class / as of 30/10/2015 implemented in Europeana	Used for LIDO to Europeana xsl	Used for LIDO Profile	Mandatory	Range	Value
dc:contributor	yes / yes	yes	no -> event-mediated	no		literal or ref
dc:creator	yes / yes	yes	no -> event-mediated	no		literal or ref
dc:publisher	yes / yes	yes	no -> event-mediated	no		literal or ref
dc:date	yes / yes	yes	no -> event-mediated	no		literal or ref
dcterms:created	yes / yes	yes	no -> event-mediated	no		literal or ref
dcterms:issued	yes / yes	yes	no -> event-mediated	no		literal or ref
dc:format	yes / yes	yes	?	no		literal or ref
dcterms:extent	yes / yes	yes	yes	no		literal or ref
dcterms:medium	yes / yes	yes	no -> event-mediated	no		literal or ref
dc:identifier	yes / yes	yes	yes	no		literal
dc:title	yes / yes	yes	yes	Either dc:title or dc:description must be provided		literal
dcterms:alternative	yes / yes	yes	yes	no		literal
dcterms:provenance	yes / yes	yes		no		literal or ref
dc:language	yes / yes	yes	yes	mandatory in case that edm:type = TEXT		literal
edm:currentLocation	yes	yes	yes? -> create always Place resource?	no	edm:Place	ref
dc:relation	yes / yes	missing -> default for	yes	no		literal or ref

Property Name	EDM Class / as of 30/10/2015 implemented in Europeana	Used for LIDO to Europeana xsl	Used for LIDO Profile	Mandatory	Range	Value
		RelatedWorkSet				
dc:source	yes / yes	no (for RelatedWorkSet?)	?	no		literal or ref
dcterms:hasPart	yes / yes	yes	yes	no		literal or ref
dcterms:isPartOf	yes	yes	yes	no		URI of edm:ProvidedCHO or URI of edm:PhysicalThing
edm:hasMet	yes / yes for edm:ProvidedCHO and edm:Agent	no	yes, but for edm:Event	no	edm:Event	ref
dc:rights	yes / yes	yes	yes	no		literal or ref
dc:coverage	yes / yes	yes	no	one of dc:coverage or dc:subject or dc:type or dcterms:spatial must be provided		literal or ref
dc:description	yes / yes	yes	yes	Either dc:title or dc:description must be provided		literal or ref
dc:subject	yes / yes	yes	no (see refinements)	one of dc:coverage or dc:subject or dc:type or dcterms:spatial must be provided		literal or ref
dcterms:subject	no		refine to subjectConcept,	no		ref



Property Name	EDM Class / as of 30/10/2015 implemented in Europeana	Used for LIDO to Europeana xsl	Used for LIDO Profile	Mandatory	Range	Value
			subjectActor etc.			
dcterms:spatial	yes / yes	yes	no -> event-mediated	one of dc:coverage or dc:subject or dc:type or dcterms:spatial must be provided		literal or ref
dc:type	yes / yes	yes	yes	one of dc:coverage or dc:subject or dc:type or dcterms:spatial must be provided		literal or ref
edm:type	yes / yes	yes	no? -> applies to WebResource, not CHO!	yes		literal – one of the Europeana Types (TEXT, VIDEO, SOUND, IMAGE, 3D)
edm:hasType	yes / yes	yes	yes	no	skos:concept	literal or ref

Table 4: Properties of edm:WebResource

Property Name	EDM Class / as of 30/10/2015 implemented in Europeana	Used for LIDO to Europeana xsl	Used for LIDO Profile	Mandatory	Has Range	Value



edm:rights	yes / yes	yes	yes	yes		EDM Rights Statements see: <a href="http://pro.europeana.eu/web/guest/available-rights-statements">http://pro.europeana.eu/web/guest/available-rights-statements</a>
dc:rights	yes / yes	yes	yes	no		literal
dc:description		no	yes	no		literal

Table 5: Properties of ore:Aggregation

Property Name	EDM Class / as of 30/10/2015 implemented in Europeana	Used for LIDO to Europeana xsl	Used for LIDO Profile	Mandatory	Range	Value
edm:aggregatedCHO	yes	yes	yes	yes	edm:ProvidedCHO	URI of provided CHO
edm:dataProvider	yes	yes	yes	yes		literal or ref
edm:hasView	yes	yes	yes	no – only used if edm:isShownAt or edm:isShownBy already used	edm:WebResource	URI
edm:isShownBy	yes	yes	yes	Either edm:isShownAt or edm:isShownBy is mandatory	edm:WebResource	URI
edm:isShownAt	yes	yes	yes	Either edm:isShownAt or edm:isShownBy is mandatory	edm:WebResource	URI
edm:object	yes	yes	yes	no	edm:WebResource	Thumbnail URI

edm:provider	yes	yes	no	yes		
edm:rights	yes	yes	yes	yes		EDM Rights Statements see: <a href="http://pro.europeana.eu/web/guest/available-rights-statements">http://pro.europeana.eu/web/guest/available-rights-statements</a>
dc:identifier	no	no	yes	yes		ref

## 4 HELPDESK SUPPORT SYSTEM

The main objective of this task was the smooth publication of providers' metadata on Europeana. Content Providers needed to get instructions while mapping their in-house metadata to LIDO for exploiting the full expressiveness of LIDO model, as well as technical information about MINT functionalities that allowed them to fit their metadata perfectly according to the AthenaPlus requirements. This task has been closely related to the helpdesk service of WP2 coordinated by ICCU (Task 2.4).

In detail the mailing list [helpdesk@athenaplus.eu](mailto:helpdesk@athenaplus.eu) has been made that content providers could use for raising their metadata related questions.

- NTUA was responsible for the technical matters related to the MINT mapping tool, including maintenance, and assistance on MINT's advance mappings functionality.
- UNIMAR assisted providers during the mapping of their metadata to LIDO, and
- ICCU coordinated the issues and was responsible for the communication between the content providers, NTUA and Europeana – whenever this was necessary for resolving problems detected on Europeana's side when checking the metadata.

In general although there is extended documentation about the most popular topics providers preferred to directly use the list since the advanced metadata topics are hard for them to understand due to their not technical background. The most popular topics discussed in the list are summarized below:

- Mapping of multiple resources for the same record: As discussed in the previous section one of quite recently developed Europeana features was the support of records with more than one digital resources. This proved to be a bit complicated for the providers that wanted to use this functionality since it normally included a modification on their export – so the extra information could be properly used in the mapping – and the use of MINT's advance mapping functionalities like the structural mapping. Although instructions have been circulated to the list the most partners had difficulties in applying them due to their limited IT background.
- Multiple Events: Similarly the proper mapping of multiple Event information for the same object in the source record have proved to be quite complicated for the partners because a different export was required and the creation of the respective LIDO Event structure in MINT. Apart from that the resulted outcome could not be easily observed through MINT because EDM does not support Events. Therefore providers had to go through the resulting LIDO xml to see if the mapping they made was correct.
- Failed transformations: A technical problem providers usually had and was hard for them to understand the reason why it happened was the transformation with invalid files. The problem occurred either because an xpath of their in-house metadata that did not have a value for all the imported records was mapped to a LIDO mandatory element or because they were using a condition that was not always satisfied for a LIDO mandatory element. In both cases the result was the missing value for a mandatory LIDO element that invalidated some of the records. The tricky part of this problem was the fact it couldn't be easily discovered through the preview interfaces – because in most cases the lack of the mandatory element was the exception. In addition this was a problem highly metadata and mapping related that couldn't be understood by content providers as a general problem although it has been many times described and answered in the list.
- Failed OAI publication: This was a problem that in most cases was related to the previous one. MINT only publishes on NTUA's OAI-PMH the valid records of a transformation. Therefore the aforementioned problem – although can be seen through the MINT interface – was in most cases discovered when the number of the imported records did not match the number of the published records. Another problem that happened few times and had the same result (i.e. less records on the OAI-PMH server than the records transformed and published) was the OAI-PMH server unexpected shutdown.
- Europeana Conflicts – Mapping of identifiers: Most of the content providers asked for assistance when Europeana detected conflicts in their records. As described in the LIDO to EDM mechanism

the lido:RecordId values have been used in the XSL for the creation of the edm:ProvidedCHO and ore:Aggregation rdf:about identifiers that are examined by Europeana for detecting conflicts. Unfortunately, although this has been many times discussed in the training sessions and in the mailing list some providers did not map unique values there, or the values were unique among the dataset but not among the whole collection. Therefore, since most of them were not RDF experts they asked for assistance for resolving the conflicts.

- Assistance in usage of advance mappings: The limited background in XML and XSL of the majority of the content providers was the main reason why they were asking for assistance whenever an advance mapping functionality of MINT like structural, conditional and thesauri mapping had to be applied.

The mailing list has been very active and in total 1433 mails have been exchanged from the beginning of the project and it proved to be very helpful for the providers. Without the very affective use of the list it would have been really hard for to project to achieve one of its main objectives that was the publication of **3.5 million** records on Europeana.

## 5 METADATA INGESTION

The main objective of this task was the transformation of provider's metadata to the EDM. In order to achieve it, NTUA has set up a MINT instance for the AthenaPlus project that was available from M06 This constituted the starting point of the aggregation of metadata, which continued till M30. The main functionalities of MINT have been summarized in section 2.1 and they have been presented in detail in D 3.1. In addition D 3.1 presented the customizations exclusively performed for the AthenaPlus MINT instance that are summarized below.

- MINT has been customized so as to assist the content providers to easily map their metadata to LIDO with the aid of element's bookmarks.
- The MINT mapping editor has been totally re-constructed taking into account the experience gathered from the usage of MINT in other projects and also the feedback from content providers. The new mapping editor is based on a new JSON library (JsonSmart) that is much faster and responsive than the one used in previous versions. It also permits the creation of more complex mappings namely
  - Structural mappings on any element,
  - if-else conditional mappingsmaking in that way the implementation of crosswalks from any provider's input schema to LIDO possible.
- MINT fully supports the Europeana Data Model (crosswalk from LIDO to EDM and Europeana previews based on their new portal).
- Progress reports for the metadata production and publication to our (OAI-PMH) repository are supported as a pdf document for download.

Since the delivery of D 3.1 minor bugs have been fixed and many updates of the XSLT mechanism that transforms valid LIDO xml files to valid EDM files have been performed that were reported in detail in the previous section. The main development performed for this task since the delivery of the D 3.1 was the set up of MINT instances for the update of metadata published for the Athena and the LinkedHeritage project. Most of the AthenaPlus content providers also participated into these projects. However, back at the days where these projects were in progress Europeana supported ESE, thus the metadata these projects produced and delivered to Europeana were in ESE. Once providers discovered the advanced expressive power of EDM and the functionalities of the AthenaPlus MINT instance – through which they could produce better quality metadata – they requested the update of the metadata they had published in these projects. Therefore NTUA set up two new MINT instances one for the Athena<sup>8</sup> and one for the LinkedHeritage<sup>9</sup> project identical – as far the customizations are concerned – to the one of AthenaPlus. In addition two separate OAI-PMH repositories<sup>10,11</sup> have been configured and set up for holding the metadata and making the delivery to Europeana possible. These instances have been communicated through the project to all the partners involved in these projects and also to the list of the original projects and so far 936.101 records have been updated for the Athena project and 238.982 for the LinkedHeritage.

### 5.1 MINT instance for producing high quality LIDO metadata

The high quality of metadata is one of the topics that has lately gained a lot of attention. Europeana recognized the importance of high quality metadata by the creation of the metadata quality taskforce<sup>12</sup>. Europeana defines metadata quality as a set of processes which ensures that cultural heritage objects can be identified, discovered and seen in context by audiences, in a manner appropriate to the context in which the data provider created them. A task group with similar objectives to the one created by Europeana but focused on the LIDO metadata standard instead of the EDM has been established within

---

<sup>8</sup> <http://mint-projects.image.ntua.gr/athenaupdate>

<sup>9</sup> <http://mint-projects.image.ntua.gr/linkedheritageupdate>

<sup>10</sup> <http://panic.image.ntua.gr:9000/manager/projects/athenaupdate>

<sup>11</sup> <http://panic.image.ntua.gr:9000/manager/projects/linkedheritageupdate>

<sup>12</sup> <http://pro.europeana.eu/publication/metadata-quality-task-force-report>

the project consortium. This task group has authored guidelines – through the use of specific LIDO elements together with specific values – for the production, visualization and indexing of museum content using LIDO. These guidelines have been used for the configuration of a different MINT instance that has been set up especially for this purpose and several providers volunteered to follow them in producing 100 high quality metadata records (see also D 7.7).

<http://mint-projects.image.ntua.gr/athenaplusmuseum/>

The following table presents the bookmarks of the MINT instance that have been set up and correspond to specific LIDO paths having the cardinality shown in the table. These bookmarks have been selected according to the “visualization and indexing of museum content” document authored by the task group.

**Table 6:LIDO Bookmarks on MINT for based on “visualization and indexing of museum content” document**

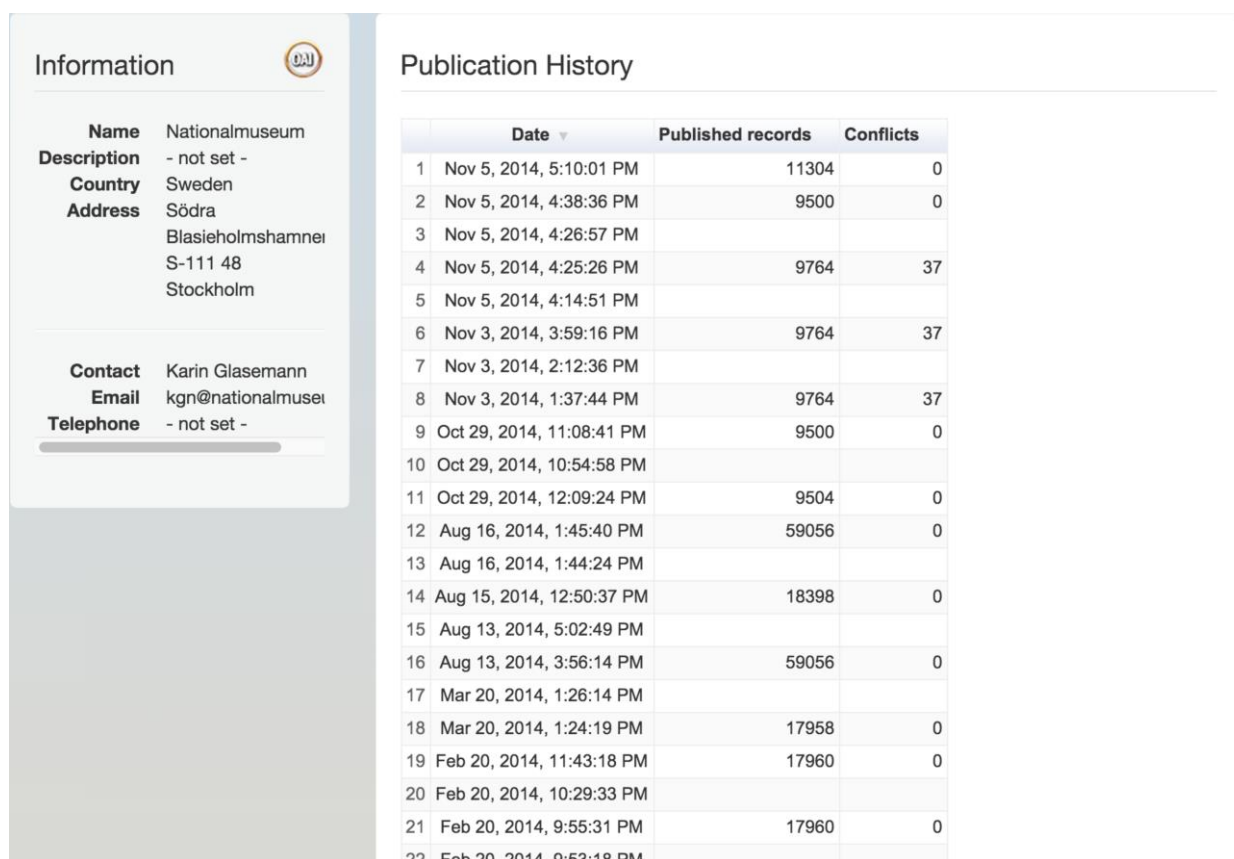
<b>N.</b>	<b>Mandatory?</b> M = Mandatory M/A = Mandatory if applicable   R = Recommended	<b>Repea- table?</b> R = repeatable NR = not repeatable	<b>Element</b>	<b>LIDO path</b>
1.	<b>M</b>	NR	Metadata language for descriptive metadata	lido:descriptiveMetadata+xml:lang
2.	<b>M</b>	R	Object / Work type	lido:objectWorkType
3.	<b>M</b>	R	Cultural Heritage type	lido:classification with @lido:type="generic"
4.	<b>M</b>	R	Thematic context	lido:classification with @lido:type="universal classification" and @lido:source="UDC"
5.	<b>M</b>	R	Title	lido:titleSet
6.	<b>M/A</b>	NR	Repository name	(lido:repositorySet@lido:type="current") lido:repositoryName
7.	<b>R</b>	NR	Inventory number	(lido:repositorySet@lido:type="current") lido:workID
8.	<b>M</b>	NR	Location	(lido:repositorySet@lido:type="current") lido:repositoryLocation
9.	<b>R</b>	R	Object description	lido:objectDescriptionSet - lido:descriptiveNoteValue
10	<b>R</b>	R	Object measurements	lido:objectMeasurementsSet - lido:displayObjectMeasurements
11	<b>R</b>	R	Inscriptions	lido:inscriptions
12	<b>M</b>	R	Event set with event type and at least one sub-element	lido:eventSet with - lido:eventType
13		R	- Event Actor	(lido:eventSet with lido:eventType) - lido:eventActor
14	<b>M/A</b>	NR	-- Actor	(- lido:eventActor) - lido:actor
15	<b>R</b>	R	-- Role Actor	(- lido:eventActor) - lido:roleActor
16	<b>R</b>	R	-- Attribution Qualifier	(- lido:eventActor) - lido:attributionQualifierActor

17	<b>R</b>	R	-- Extent Actor	(- lido:eventActor) - lido:extentActor
18	<b>M/A</b>	R	- Cultural context	(lido:eventSet with lido:eventType) - lido:culture
19	<b>M/A</b>	NR	- Event Date	(lido:eventSet with lido:eventType) - lido:eventDate
20	<b>M/A</b>	R	- Period	(lido:eventSet with lido:eventType) - lido:periodName
21		R	- Event Place	(lido:eventSet with lido:eventType) - lido:eventPlace
22	<b>M/A</b>	NR	-- Place	(- lido:eventPlace) - lido:place
23		R	- Event Materials/Technique	(lido:eventSet with lido:eventType) - lido:eventMaterialsTech
24	<b>M/A</b>	R	-- Materials/Technique	(- lido: eventMaterialsTech) - lido:materialsTech -> lido:termMaterialsTech
25	<b>R</b>	R	-- Extent Materials/Technique	(- lido: eventMaterialsTech) - lido:materialsTech -> lido:extentMaterialsTech
26	<b>M/A</b>	R	Subject	lido:subjectSet with - lido:subjectConcept or lido:subjectActor or lido:subjectDate or lido:subjectPlace
27	<b>R</b>	R	Related works	lido:relatedWorkSet with - lido:relatedWork-> lido:objectWebResource and lido:objectNote - lido:relatedWorkRelType
28	<b>M</b>	NR	Metadata language for administrative metadata	lido:administrativeMetadata+xml:lan g
29	<b>R</b>	R	Rights work	lido:rightsWork -> lido:creditLine
30	<b>M</b>	R	Record identifier	lido:recordID
31	<b>M</b>	NR	Record type	lido:recordType
32	<b>M</b>	R	Record source	lido:recordSource
33	<b>M</b>	NR	Record rights	lido:recordRights
34	<b>R</b>	R	Record Info Link	lido:recordInfoSet with - lido:recordInfoLink
35	<b>M</b>	R	Resource set	lido:resourceSet
36	<b>M</b>	R	- Link resource	(lido:resourceSet) - lido:resourceRepresentation -> lido:linkResource
37	<b>R</b>	R	- Resource description	(lido:resourceSet) - lido:resourceDescription
38	<b>R</b>	NR	- Date taken / recorded	(lido:resourceSet) - lido:resourceDateTaken
39	<b>R</b>	R	- Creator of the resource / photographer	(lido:resourceSet) - lido:resourceSource
40	<b>M</b>	R	- Rights resource	(lido:resourceSet) - rightsResource

## 6 DELIVERY TO EUROPEANA

The main objective of Task 3.4 was the delivery of high quality metadata to Europeana. MINT enables content providers to map their metadata to LIDO and from it to EDM through a user-friendly interface. In addition, it provides validation services for both LIDO and EDM metadata models together with preview interfaces through which providers can a-priori check how would their metadata look like when published on Europeana, ensuring in this way a good view of their records and the production of high quality metadata. Section 6.3 of deliverable D3.1 presented in detail the NoSQL (MongoDB) backend set up for storing the successfully transformed EDM metadata and serving them using the Open Archives Initiative-Protocol for Metadata Harvesting (OAI-PMH).

Since the delivery of D 3.1 a major update of the MINT publication mechanism to the OAI-PMH server has been performed. The main limitation of the first version was that it only allowed the metadata from one schema to be delivered to the OAI-PMH repository and that was EDM. More precisely, the valid EDM records produced through MINT were transferred to the NoSQL database. The publication mechanism was extracting the DC and DCTERM elements from them for the creation of the OAI\_DC records. Since two different flavors – one in EDM and one in OAI\_DC – existed by default two OAI namespaces were created respectively rdf<sup>13</sup> and oai\_dc<sup>14</sup> (see **Figure 5**). In other words the LIDO metadata that were originally produced by the content providers were just acting as an intermediate to EDM that couldn't be re-used by the providers.



The screenshot displays the OAI-PMH interface for Nationalmuseum. On the left, the 'Information' section provides details about the institution, including its name, description, country, address, and contact information. On the right, the 'Publication History' section shows a table of published records with columns for Date, Published records, and Conflicts.

	Date ▾	Published records	Conflicts
1	Nov 5, 2014, 5:10:01 PM	11304	0
2	Nov 5, 2014, 4:38:36 PM	9500	0
3	Nov 5, 2014, 4:26:57 PM		
4	Nov 5, 2014, 4:25:26 PM	9764	37
5	Nov 5, 2014, 4:14:51 PM		
6	Nov 3, 2014, 3:59:16 PM	9764	37
7	Nov 3, 2014, 2:12:36 PM		
8	Nov 3, 2014, 1:37:44 PM	9764	37
9	Oct 29, 2014, 11:08:41 PM	9500	0
10	Oct 29, 2014, 10:54:58 PM		
11	Oct 29, 2014, 12:09:24 PM	9504	0
12	Aug 16, 2014, 1:45:40 PM	59056	0
13	Aug 16, 2014, 1:44:24 PM		
14	Aug 15, 2014, 12:50:37 PM	18398	0
15	Aug 13, 2014, 5:02:49 PM		
16	Aug 13, 2014, 3:56:14 PM	59056	0
17	Mar 20, 2014, 1:26:14 PM		
18	Mar 20, 2014, 1:24:19 PM	17958	0
19	Feb 20, 2014, 11:43:18 PM	17960	0
20	Feb 20, 2014, 10:29:33 PM		
21	Feb 20, 2014, 9:55:31 PM	17960	0
22	Feb 20, 2014, 9:53:18 PM		

**Figure 5: OAI previous implementation**

<sup>13</sup> <http://panic.image.ece.ntua.gr:9000/athenaplus/oai?verb=ListRecords&metadataPrefix=rdf>

<sup>14</sup> [http://panic.image.ece.ntua.gr:9000/athenaplus/oai?verb=ListRecords&metadataPrefix=oai\\_dc](http://panic.image.ece.ntua.gr:9000/athenaplus/oai?verb=ListRecords&metadataPrefix=oai_dc)



The new publication mechanism is more sophisticated and permits the use of arbitrary schemas – in the AthenaPlus case these are LIDO and EDM metadata – and their re-use through OAI-PMH under different or the same namespace. More specifically the new publication method allows:

- The delivery of metadata from more than one schema through MINT.
- The definition of a publication schema that will be used for the creation of the OAI\_DC record.
- The definition of the OAI namespaces under which metadata coming from different schemas are published.

Table 7 presents the OAI configuration of the schemas that has been setup for the AthenaPlus project. As shown, EDM is published on the rdf OAI namespace and it is used for the creation of the OAI-DC record while both LIDO – that is the LIDO schema with some customizations and some values fixed for the AthenaPlus project (for more details see section 4.1.1 of D 3.2) – and LIDO Plain – that is the LIDO schema without any of them – are published on LIDO namespace.

**Table 7: AthenaPlus OAI setup**

Schema	Used for OAI_DC	OAI Namespace
EDM	True	rdf
LIDO	False	Lido
LIDO Plain	False	Lido

Once the new publication mechanism has been tested using part of the AthenaPlus datasets NTUA made the migration to the new one. The migration to the new server was a complicated and critical process that consisted of the following steps:

1. Shutdown of the MINT server for preventing the creation and publication of new metadata.
2. Backup of the database MINT uses for AthenaPlus for recovery in case of problems during the migration.
3. Development of a script executed on the AthenaPlus MINT server that:
  - a. Detected all the published datasets from all the organizations.
  - b. Deleted their corresponding publications records from the publication table of the database.
  - c. Published them again using the updated publication mechanism.
4. Validation of the migration process by comparing the published records for each of organization.

Finally, the web-application that is set up on top of the NoSQL repository for monitoring the contents of the OAI-PMH server has been appropriately updated. The new address for it is

<http://panic.image.ntua.gr:9876/manager/projects/athenaplus>

Figure 6 shows the updated interface that now consists of three pies – each one of them corresponding to OAI namespaces – and as usually the organization on the left with a link to its detailed OAI-PMH reporting page, the total number of records of records published, the date of the last publication. The detailed OAI-PMH reporting page has been also updated (see **Figure 7**) by adding a table on the top of the page that presents OAI namespaces together with links to the OAI-PMH repositories and the number of records published for each one of them. It is important at that point to mention the number of records published to the three OAI namespaces is not always the same. More precisely there is a possibility that the lido and rdf namespaces hold different number of records (the number of records in the oai\_dc and in the rdf namespaces are always the same because as described oai\_dc is produced from the EDM records that are stored in the rdf namespace) the reasons why this is happening because

1. Some providers map directly to EDM (so no LIDO metadata are produced)

- Invalid LIDO records, end up to valid EDM records but since only the valid records are sent to the repository this implies less LIDO records sent than EDM records.

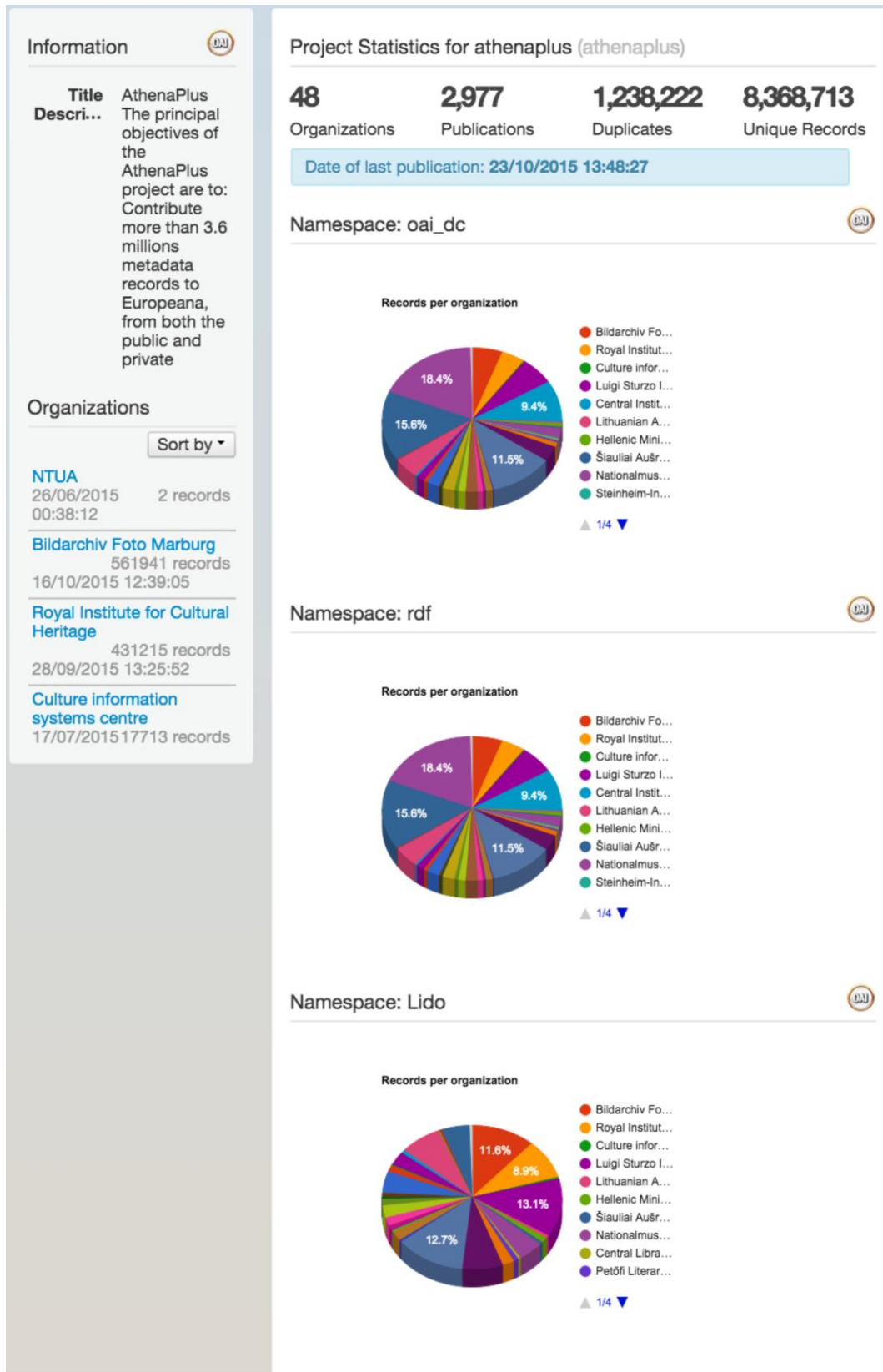


Figure 6: The updated AthenaPlus OAI-PMH monitoring interface

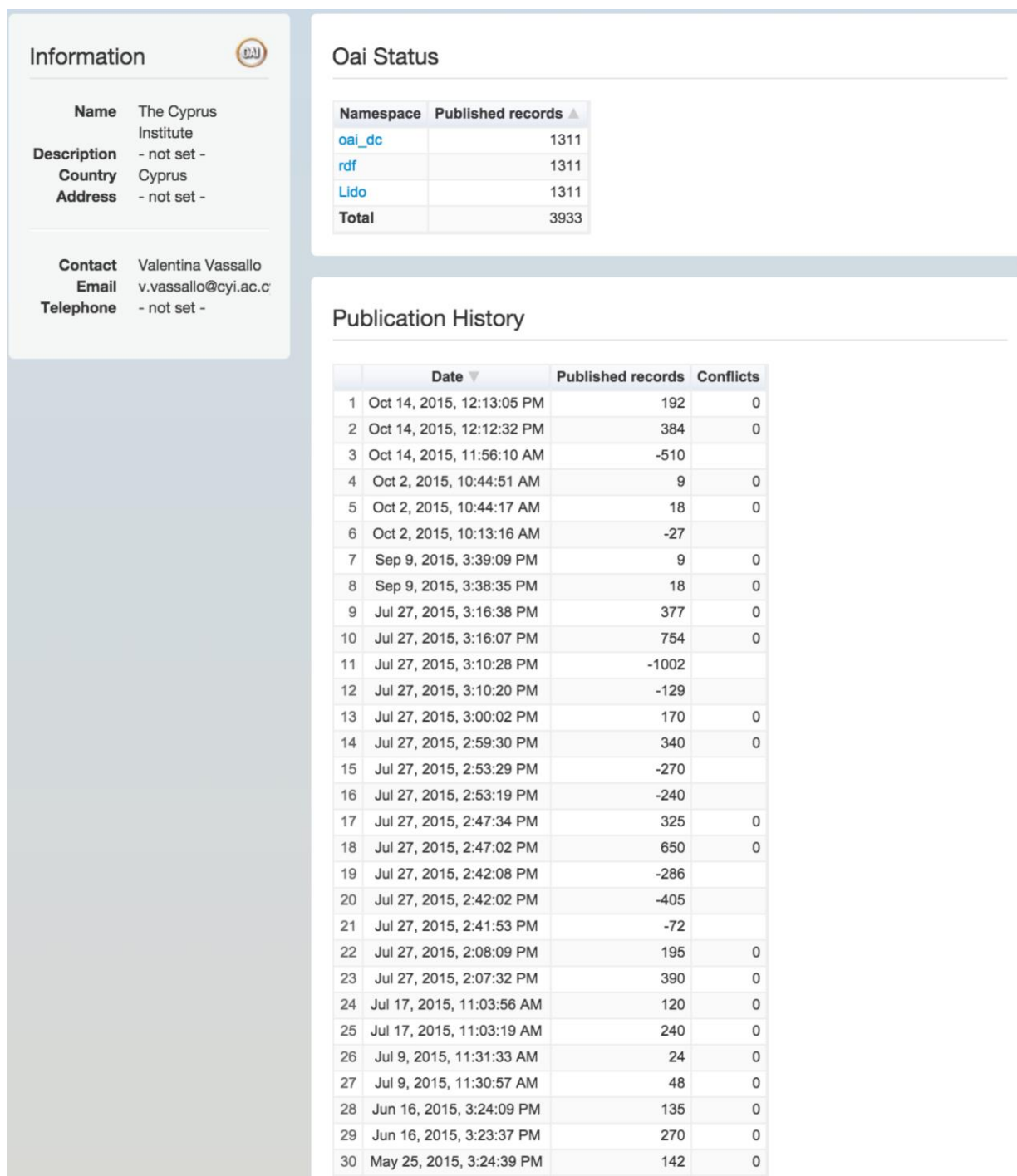


Figure 7: The updated organization OAI-PMH monitoring interface

Europeana in co-operation with the WP2 and WP3 leaders has successfully harvested and published 3.5 million EDM records for the AthenaPlus project.

[http://www.europeana.eu/portal/search.html?query=\\*&qf=PROVIDER:AthenaPlus](http://www.europeana.eu/portal/search.html?query=*&qf=PROVIDER:AthenaPlus)

A detailed report of the records each of the providers published can be found at D 2.4.

## 7 Quality evaluation on LIDO metadata

One of the main objectives of the project was the production of high quality LIDO metadata. LIDO is an event centric metadata standard based on CRM, therefore this section focuses on the analysis of the LIDO elements and values used, putting special effort on the Events and vocabularies. More specifically a representative set of LIDO metadata produced through the MINT instance of the AthenaPlus project are examined as well as the metadata that the providers produced through the MINT instance that has been set up according to the guidelines authored by the AthenaPlus task group for the production, visualization and indexing of museum content using LIDO.

### 7.1 Selection of LIDO metadata for the evaluations

The total number of metadata produced for the AthenaPlus project is 3.5 million records and the evaluation of such a big set of records was not possible. Therefore the creation of a representative LIDO dataset from the AthenaPlus MINT instance had to be made.

The majority of the providers have used LIDO as the intermediate schema for producing the EDM records delivered to Europeana (only few of them have directly used EDM and most of them did that because they already had their metadata in EDM). The LIDO metadata were either directly imported into MINT or produced by the creation of a mapping from provider's in-house metadata to LIDO. Someone may assume at this point that only a mapping was required by each provider but this was not the case for most of them. Since the input sets of metadata varied depending on the export or on the source, providers were using many different mappings one corresponding to each of the import sets (some of the organizations provided content for more than one sub-organizations they were managing, thus different input datasets were employed). Apart from the mappings another important parameter for creating a good sample for evaluation was the number of the input data resulting into LIDO metadata after the mapping. More specifically a provider could have many inputs with the number of records contained in each of them varying (e.g. import A with 50 records and import B with 1.000 records). Therefore the approach we followed for creating a LIDO dataset for quality evaluation from the AthenaPlus MINT instance was the selection of the 20% of LIDO records that were either directly imported into MINT or they have been produced from a transformation. In that way a set of 670.149 records has been created taking into account all the mappings and also the original size of the imported datasets.

As for the MINT instance that has been set up according to the guidelines authored by the AthenaPlus Task Group for the Production, Visualization and Indexing of museum content using LIDO (MINT-TGPVIL), the selection of LIDO metadata was much easier because providers have been asked to provide a dataset of minimum 100 records. So all the LIDO records have been used that result to 2,513 records.

### 7.2 Quality Evaluation

Good quality metadata of an item should be sufficient for the identification of the item, its discovery and its re-use. Therefore the evaluation of the LIDO records is based on elements that can be used for fulfilling these criteria. The examination has been performed using MINT and its statistics functionality. So the LIDO datasets created as described in the previous sections have been imported into a MINT instance and an evaluation based on the instances and the values of specific elements has been performed. More specifically the use and values of the following elements have been examined.

#### Language attribute

The descriptive and administrative metadata language has been set mandatory in the LIDO schema used for the AthenaPlus. So all the records have an xml language attribute. However, by the examination of the language attributes values used, we examine the diversity of the content provided through the project.

**Table 8: LIDO use of xml language attributes – AthenaPlus Instance**

Language	Instances	Percentage
it	232470	34.69%
ca	73720	11.00%
sv	56212	8.39%
en	162221	24.21%
el	42146	6.29%
de	28710	4.28%
nl	16264	2.43%
fr	13326	1.99%
et	11673	1.74%
sl	7505	1.12%
hu	7352	1.10%
pl	6995	1.04%
lt	4665	0.70%
ro	2145	0.32%
cs	1487	0.22%
lv	1542	0.23%
bg	1008	0.15%
ua	532	0.08%
ca	117	0.02%
ru	58	0.01%
ar	1	0.00%

**Table 9: LIDO use of xml language attributes – MINT-TGPVIL**

Language	Instances	Percentage
en	855	34.02%
de	414	16.47%
hr	408	16.24%
sv	221	8.79%
lt	201	8.00%
hu	156	6.21%
pl	114	4.54%
nl	99	3.94%
it	45	1.79%

What these tables clearly reflect is the consortium of the AthenaPlus project that consist of partners around Europe who provide multilingual metadata.

### Object Work Type & Classification

The object work type and the classification elements in LIDO are used for providing a way to discover. Object work type can take vocabulary values (conceptID) and also term values. The following tables present the amount of instances these xpaths have in the metadata and the number of unique records.

**Table 10: LIDO use of ObjectWorkType – AthenaPlus Instance**

	Values	Average per record	Unique Values
<b>conceptID</b>	171833	0.256410142	6292
<b>Term</b>	875908	1.307034704	11531

**Table 11 : LIDO use of ObjectWorkType – MINT-TGPVIL**

	Values	Average per record	Unique Values
<b>conceptID</b>	1800	0.716275368	323
<b>Term</b>	3738	1.487465181	709

What we can observe from these tables is that the objectWorkType xpath of the MINT instance has 1.55 instances per record 0.25 of them with the use of the conceptID and 1.3 with the use of term. This can be interpreted as the preference of the content providers for terms instead of vocabulary URIs. When we examined the distinct values of the conceptID path we observed that despite the fact that the unique values number is high and someone would expect 6292 distinct URI values, most of them are terms that are mistakenly mapped to conceptID. This is not the case for the MINT-TGPVIL instance that not only has more ObjectWorkType instances per record (2.1) but the 323 unique values are all unique because this element in the MINT-TGPVIL instance is controlled by AAT thesaurus (i.e. only values from AAT can be produced). The comparison of the results coming from the two instances in this case present that although a decent quality on metadata has been established through the AthenaPlus project – the instances of ObjectWorkType remain is 1.3 so all the records contain that information – this can further be improved by setting more strict mapping requirements.

Similar results can also be obtained for classification element that can also take vocabulary values or terms. Here the difference of conceptID usage is bigger (0.004 in AthenaPlus 1.2 in MINT MINT-TGPVIL) that results to a total 2.3 instances of classification in the MINT-TGPVIL instead of 1.7 that are in AthenaPlus.

**Table 12: LIDO use of Classification – AthenaPlus Instance**

	Values	Average per record	Unique Values
<b>conceptID</b>	2832	0.004225926	255
<b>Term</b>	1142488	1.704826837	2753

**Table 13: LIDO use of Classification – MINT-TGPVIL**

	Values	Average per record	Unique Values
<b>conceptID</b>	3211	1.277755671	77
<b>Term</b>	2963	1.179068842	45

### Title & Description & Location

The title, description and location elements also provide vital information about the item. More specifically these elements are mostly used for identifying and discovering the item. The instances of these elements have been examined together with the percentage of them that hold multilingual values. In all the cases the average per record was higher for the MINT-TGPVIL instance but this doesn't mean that those of the AthenaPlus instance were low – 0.96 for the title, 0.57 for the Description while EDM requires one of these two values. What is also worth pointing is the very high percentage of multilingual values produced through the MINT-TGPVIL instance that was nearly 70%. On the other hand the multilingual percentage

of the LIDO metadata produced through the AthenaPlus instance is low for the title (13.05) but sufficient for the description (39.05) and the location (77.45)

**Table 14: LIDO use of Title – AthenaPlus Instance**

	Instances	Average per record	Multilingual	Multilingual %
<b>Title</b>	649600	0.96	84747	13.05%

**Table 15: LIDO use of Title – MINT-TGPVIL**

	Instances	Average per record	Multilingual	Multilingual %
<b>Title</b>	2832	1.12	1897	66.98%

**Table 16: LIDO use of Description – AthenaPlus Instance**

	Instances	Average per record	Multilingual	Multilingual %
<b>Description</b>	385992	0.57	150804	39.05%

**Table 17: LIDO use of Description – MINT-TGPVIL**

	Instances	Average per record	Multilingual	Multilingual %
<b>Description</b>	2341	0.93	1675	71.55%

**Table 18: LIDO use of Location – AthenaPlus Instance**

	Instances	Average per record	Multilingual	Multilingual %
<b>Location</b>	225078	0.33	174329	77.45%

**Table 19: LIDO use of Location – MINT-TGPVIL**

	Instances	Average per record	Multilingual	Multilingual %
<b>Location</b>	4082	1.62	3169	77.63%

## Event

A strong point of LIDO is its Event centric structure – that is one of the biggest limitations of EDM although in theory it supports Events. Unfortunately, not all the information that is hold within a LIDO event can be represented to EDM as explained in section 3. However the AthenaPlus content providers have been encouraged to use the Event complex structure and the following tables present the different kinds of Events that have been produced by the two MINT instances. It can be observed that in both cases 18 different Events types are used. In addition for the LIDO metadata produced through the AthenaPlus instance there are 1.086.175 event types in 670.149 records that means that each LIDO record has 1.4 event types in average (i.e. it is described by more than one event) while in the MINT-TGPVIL each record reached 2.52 event types (6334/2513)

**Table 20: LIDO use of EventType – AthenaPlus Instance**

Production	698791	64.34%
Publication	98684	9.09%
Creation	62792	5.78%
Provenance	47596	4.38%
(Non-specified)	14182	1.31%

Acquisition	39664	3.65%
Modification	6878	0.63%
Excavation	12364	1.14%
Use	3288	0.30%
Type creation	2536	0.23%
Restoration	3348	0.31%
Part addition	3358	0.31%
Commissioning	2549	0.23%
Finding	87082	8.02%
Part removal	1218	0.11%
Designing	1126	0.10%
Collecting	322	0.03%
Destruction	376	0.03%
Move	21	0.00%
<b>Total</b>	<b>1086175</b>	<b>100%</b>

**Table 21: LIDO use of EventType – MINT-TGPVIL**

Production	2621	41.38%
Creation	1788	28.23%
Provenance	690	10.89%
Designing	169	2.67%
Acquisition	141	2.23%
Collecting	217	3.43%
Modification	152	2.40%
Restoration	138	2.18%
Commissioning	98	1.55%
Part addition	76	1.20%
Publication	27	0.43%
Part removal	34	0.54%
Use	34	0.54%
Destruction	18	0.28%
Excavation	8	0.13%
Beginning of Existence	114	1.80%
Execution	5	0.08%
Marketing	4	0.06%
<b>Total</b>	<b>6334</b>	<b>100%</b>



## 8 CONCLUSION

This deliverable presented the platform for metadata aggregation and delivery to Europeana as this has been evolved within the AthenaPlus project. In detail it summarized all the activities performed that are related to the WP3 tasks that are:

- Task 3.1: Compliance with CH metadata modelling requirements.
- Task 3.2: Help desk support system.
- Task 3.3: Metadata Ingestion.
- Task 3.4: Delivery to Europeana.

The platform for metadata aggregation and delivery to Europeana of which main component is the MINT mapping tool can be considered successful because it enabled the AthenaPlus content providers not only to transform and deliver to Europeana **3.5 million** records but also to produce high quality metadata in both LIDO and EDM that are exploitable through the NTUA's OAI-PMH server.

### 8.1 Results

The activities reported in this deliverable contributed to the achievement of the following milestones:

- MS4 10-20% content uploaded in MINT
- MS5 60-70% content uploaded in MINT
- MS6 100% content uploaded in MINT
- MS7 MINT/AthenaPlus ready for delivering content to Europeana

### 8.2 Impact

The main impact of WP3 was the production and delivery of **3.5 million** high quality metadata records to Europeana.

## **9 APPENDIX 1: DEFINITION OF TERMS AND ABBREVIATIONS**

CSV	Comma Separated Values
DC	Dublin Core
EDM	Europeana Data Model
ESE	Europeana Semantic Elements
LIDO	Lightweight Information Describing Objects
XML	Extensible Markup Language
XSD	XML Schema
JSON	JavaScript Object Notation

## 10 APPENDIX 2: LIDO to EDM XSLT

```

<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet exclude-result-prefixes="lido xml" version="2.0"
  xmlns:crm="http://www.cidoc-crm.org/rdfs/cidoc_crm_v5.0.2_english_label.rdfs#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:dcterms="http://purl.org/dc/terms/"
  xmlns:edm="http://www.europeana.eu/schemas/edm/"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:lido="http://www.lido-schema.org"
  xmlns:ore="http://www.openarchives.org/ore/terms/"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdaGr2="http://rdvocab.info/ElementsGr2/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:skos="http://www.w3.org/2004/02/skos/core#"
  xmlns:wgs84="http://www.w3.org/2003/01/geo/wgs84_pos#"
  xmlns:xalan="http://xml.apache.org/xalan"
  xmlns:xml="http://www.w3.org/XML/1998/namespace"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="UTF-8" standalone="yes" indent="yes"/>

  <xsl:variable name="var0">
    <item>TEXT</item>
    <item>VIDEO</item>
    <item>IMAGE</item>
    <item>SOUND</item>
    <item>3D</item>
  </xsl:variable>

  <!-- Specify edm:provider -->
  <xsl:param name="edm_provider" select="'Athena_Plus'"/>
  <!-- Specify edm:providedCHO baseURL -->
  <xsl:param name="edm_providedCHO" select="'http://mint-projects.image.ntua.gr/Athena_Plus/ProvidedCHO/'"/>
  <!-- Specify ore:Aggregation baseURL -->
  <xsl:param name="ore_Aggregation" select="'http://mint-projects.image.ntua.gr/Athena_Plus/Aggregation/'"/>

  <xsl:function name="lido:langSelect">
    <xsl:param name="metadataLang"/>
    <xsl:param name="localLang"/>
    <xsl:if test="( string-length($metadataLang) &lt; 4 and string-length($metadataLang)
  &gt; 0) or
      (string-length($localLang) &lt; 4 and string-length($localLang) &gt; 0)">
      <xsl:choose>
        <xsl:when test="(string-length($localLang) &lt; 4 and string-length($localLang)
  &gt; 0)">
          <xsl:value-of select="$localLang"/>
        </xsl:when>
        <xsl:when test="( string-length($metadataLang) &lt; 4 and string-
  length($metadataLang) &gt; 0)">
          <xsl:value-of select="$metadataLang"/>
        </xsl:when>
        <xsl:otherwise></xsl:otherwise>
      </xsl:choose>
    </xsl:if>
  </xsl:function>

  <xsl:template match="/">
    <rdf:RDF>

    <xsl:for-each select="/lido:lidoWrap/lido:lido | lido:lido">

      <xsl:variable name="descLang">
        <xsl:for-each select="lido:descriptiveMetadata/@xml:lang">
          <xsl:value-of select="."/>
        </xsl:for-each>
      </xsl:variable>

      <xsl:variable name="adminLang">
        <xsl:for-each select="lido:administrativeMetadata/@xml:lang">
          <xsl:value-of select="."/>
        </xsl:for-each>
      </xsl:variable>
    </xsl:for-each>
  </rdf:RDF>

```

```

</xsl:variable>

<xsl:variable name="dataProvider">
  <xsl:choose>
    <xsl:when
test="lido:administrativeMetadata/lido:recordWrap/lido:recordSource[lido:type='europeana:dataPr
ovider']">
      <xsl:for-each
select="lido:administrativeMetadata/lido:recordWrap/lido:recordSource[lido:type='europeana:data
Provider']/lido:legalBodyName/lido:appellationValue">
        <xsl:if test="position() = 1">
          <edm:dataProvider>
            <xsl:value-of select="."/>
          </edm:dataProvider>
        </xsl:if>
      </xsl:for-each>
    </xsl:when>
    <xsl:otherwise>
      <xsl:for-each
select="lido:administrativeMetadata/lido:recordWrap/lido:recordSource/lido:legalBodyName/lido:a
ppellationValue">
        <xsl:if test="position() = 1">
          <edm:dataProvider>
            <xsl:value-of select="."/>
          </edm:dataProvider>
        </xsl:if>
      </xsl:for-each>
    </xsl:otherwise>
  </xsl:choose>
</xsl:variable>

<edm:ProvidedCHO>
  <xsl:attribute name="rdf:about">
    <xsl:for-each select="lido:administrativeMetadata/lido:recordWrap/lido:recordID">
      <xsl:if test="position() = 1">
        <xsl:value-of select="concat($edm_providedCHO, $dataProvider, '/')"/>
      </xsl:if>
    </xsl:for-each>
  </xsl:attribute>

  <!-- dc:contributor : lido:eventActor with lido:eventType NOT production or creation or
designing or publication -->
  <!-- dc:contributor Resource -->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
    <xsl:if test="not((lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00228'))">
      <xsl:for-each select="lido:eventActor">
        <xsl:if test="lido:actorInRole/lido:actor/lido:actorID[starts-with(., 'http://') or
starts-with(., 'https://')]">
          <dc:contributor>
            <xsl:attribute name="rdf:resource">
              <xsl:for-each select="lido:actorInRole/lido:actor/lido:actorID[starts-
with(., 'http://') or starts-with(., 'https://')]">
                <xsl:if test="position() = 1">
                  <xsl:value-of select="."/>
                </xsl:if>
              </xsl:for-each>
            </xsl:attribute>
          </dc:contributor>
        </xsl:if>
      </xsl:for-each>
    </xsl:if>
  </xsl:for-each>

  <!-- dc:contributor : lido:eventActor with lido:eventf NOT production or creation or
designing or publication -->
  <!-- dc:contributor Resource dereferenced to prefLabel-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
    <xsl:if test="not((lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224') or (lido:eventType/lido:conceptID = 'http://terminology.lido-

```

```

schema.org/lido00228'))">
  <xsl:for-each select="lido:eventActor">
    <xsl:if test="lido:actorInRole/lido:actor/lido:actorID[starts-with(.,
'http://partage.vocnet.org/')] ">
      <xsl:variable name="partID"
select="lido:actorInRole/lido:actor/lido:actorID[1]/text()" />

      <xsl:variable name="labelLang">
        <xsl:choose>
          <xsl:when
test="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl:valu
e-of select="$descLang" /></xsl:when>
          <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
        </xsl:choose>
      </xsl:variable>
      <xsl:variable name="prefLabel"
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]" />
      <!-- <xsl:if test =
"not(lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred'
or not(@lido:pref)]) or
not(lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred'
or not(@lido:pref)])/text() = $prefLabel">-->
      <dc:contributor>
        <xsl:attribute name="xml:lang" select="$labelLang" />
        <xsl:value-of select="$prefLabel" />
      </dc:contributor>
      <!--</xsl:if>-->
    </xsl:if>
  </xsl:for-each>
</xsl:if>
</xsl:for-each>
<!-- dc:contributor Resource dereferenced to prefLabel-->

<!-- dc:contributor Literal-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
  <xsl:if test="not((lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00228'))">
    <xsl:if test="lido:eventActor[not(lido:actorInRole/lido:actor/lido:actorID[starts-
with(., 'http://') or starts-with(., 'https://')])]"> <!-- FILTER ACTOR NAME -->
      <xsl:for-each
select="lido:eventActor[not(lido:actorInRole/lido:actor/lido:actorID[starts-with(., 'http://')
or starts-with(.,
'https://')])]/lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='
preferred' or not(@lido:pref)]">
        <dc:contributor>
          <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) &gt;
0">
            <xsl:attribute name="xml:lang">
              <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
            </xsl:attribute>
          </xsl:if>
          <xsl:value-of select="."/>
        </dc:contributor>
      </xsl:for-each>
    </xsl:if>
  </xsl:if>
</xsl:for-each>
<!-- dc:contributor Literal-->

<!-- dc:contributor : lido:culture from any lido:eventSet -->
<!-- dc:contributor Resource-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:culture/lido:conc
eptID[starts-with(., 'http://') or starts-with(., 'https://')]">
  <dc:contributor>
    <xsl:attribute name="rdf:resource">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </dc:contributor>
</xsl:for-each>
<!-- dc:contributor Resource-->

```

```

    <!-- dc:contributor : lido:culture from any lido:eventSet -->
    <!-- dc:contributor Resource dereferenced to prefLabel-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:culture/lido:conceptID[starts-with(., 'http://partage.vocnet.org/')]">
        <xsl:variable name="partID" select="./text()" />

        <xsl:variable name="labelLang">
            <xsl:choose>
                <xsl:when
test="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl:valu
e-of select="$descLang" /></xsl:when>
                <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
            </xsl:choose>
        </xsl:variable>
        <xsl:variable name="prefLabel"
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]" />
        <!--<xsl:if test = "not(..lido:term) or not(..lido:term/text() =
$prefLabel)"-->
            <dc:contributor>
                <xsl:attribute name="xml:lang" select="$labelLang" />
                <xsl:value-of select="$prefLabel" />
            </dc:contributor>
            <!--</xsl:if-->

        </xsl:for-each>
    <!-- dc:contributor Resource dereferenced to prefLabel-->

    <!-- dc:contributor Literal-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:culture">
        <xsl:if test="not(lido:conceptID)"> <!-- FILTER ACTOR NAME -->
            <xsl:for-each select="lido:term[@lido:pref='preferred' or (not(@lido:pref) and
not(@lido:addedSearchTerm='yes'))]">
                <dc:contributor>
                    <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
                        <xsl:attribute name="xml:lang">
                            <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
                        </xsl:attribute>
                    </xsl:if>
                    <xsl:value-of select="."/>
                </dc:contributor>
            </xsl:for-each>
        </xsl:if>
    </xsl:for-each>
    <!-- dc:contributor Literal-->

    <!-- dc:creator Resource-->

    <!-- dc:creator : lido:eventActor with lido:eventType = production or creation or
designing-->
    <!-- dc:creator Resource dereferenced to prefLabel-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
        <xsl:if test="(lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224')">
            <xsl:for-each select="lido:eventActor">
                <xsl:if test="lido:actorInRole/lido:actor/lido:actorID[starts-with(.,
'http://partage.vocnet.org/')]">
                    <xsl:variable name="partID"
select="lido:actorInRole/lido:actor/lido:actorID[1]/text()" />

                    <xsl:variable name="labelLang">
                        <xsl:choose>
                            <xsl:when
test="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl:valu
e-of select="$descLang" /></xsl:when>
                            <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
                        </xsl:choose>
                    </xsl:variable>
                    <xsl:variable name="prefLabel"
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]" />
                    <!-- <xsl:if test =
"not(lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred'

```

```

or not(@lido:pref)) or
not(lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred'
or not(@lido:pref)]/text() = $prefLabel)">-->
    <dc:creator>
        <xsl:attribute name="xml:lang" select="$labelLang" />
        <xsl:value-of select="$prefLabel" />
    </dc:creator>
<!--</xsl:if>-->

</xsl:if>
</xsl:for-each>
</xsl:if>
</xsl:for-each>
<!-- dc:creator Resource dereferenced to prefLabel-->

<!-- dc:creator Literal-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
    <xsl:if test="(lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224')">
        <xsl:if test="lido:eventActor[not (lido:actorInRole/lido:actor/lido:actorID[starts-
with(., 'http://') or starts-with(., 'https://')]]"> <!-- FILTER ACTOR NAME -->

            <xsl:if test="lido:eventActor/lido:displayActorInRole">
                <xsl:for-each select="lido:eventActor/lido:displayActorInRole">
                    <dc:creator>
                        <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) &gt;
0">
                            <xsl:attribute name="xml:lang">
                                <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
                            </xsl:attribute>
                        </xsl:if>
                        <xsl:value-of select="."/>
                    </dc:creator>
                </xsl:for-each>
            </xsl:if>

            <xsl:if
test="lido:eventActor/lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue">
                <xsl:for-each
select="lido:eventActor/lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue">
                    <dc:creator>
                        <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang))
&gt; 0">
                            <xsl:attribute name="xml:lang">
                                <xsl:value-of
select="lido:langSelect($descLang,@xml:lang)"/>
                            </xsl:attribute>
                        </xsl:if>
                        <xsl:value-of select="."/>
                    </dc:creator>
                </xsl:for-each>
            </xsl:if>
            </xsl:if>
            </xsl:if>
        </xsl:for-each>
        <!-- dc:creator Literal-->

        <!-- dc:date -->
        <!-- dc:date : lido:eventDate with lido:eventType NOT production or creation or
designing -->
        <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
            <xsl:if test="not((lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224'))">
                <xsl:for-each select="lido:eventDate">
                    <xsl:if test="lido:date/lido:earliestDate | lido:displayDate">
                        <dc:date>
                            <xsl:choose>
                                <xsl:when test="lido:date/lido:earliestDate = lido:date/lido:latestDate">
                                    <xsl:value-of select="lido:date/lido:earliestDate"/>
                                </xsl:when>
                                <xsl:when test="lido:date/lido:earliestDate">

```

```

        <xsl:value-of select="concat(lido:date/lido:earliestDate, '/',
lido:date/lido:latestDate)"/>
    </xsl:when>
    <xsl:otherwise>
        <xsl:value-of select="lido:displayDate" />
    </xsl:otherwise>
</xsl:choose>
</dc:date>
</xsl:if>
</xsl:for-each>
</xsl:if>
</xsl:for-each>
<!-- dc:date -->

    <!-- dc:description : lido:objectDescriptionSet -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:objectDescriptionWrap/lido:
objectDescriptionSet[lido:descriptiveNoteValue/string-length(.)>0]">
        <dc:description>
            <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
                <xsl:attribute name="xml:lang">
                    <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:if test="@lido:type">
                <xsl:value-of select="concat(@lido:type, ' ' )"/>
            </xsl:if>
            <xsl:for-each select="lido:descriptiveNoteValue">
                <xsl:value-of select="concat(., ' ' )"/>
            </xsl:for-each>
            <xsl:if test="string-length(lido:sourceDescriptiveNote[1])>0">
                <xsl:value-of select="concat(' (' , lido:sourceDescriptiveNote[1], ') ')"
/>
            </xsl:if>
            <xsl:if test="string-length(lido:descriptiveNoteID[1])>0">
                <xsl:value-of select="concat(' (' , lido:descriptiveNoteID[1], ') ')" />
            </xsl:if>
        </dc:description>
    </xsl:for-each>
    <!-- dc:description -->

    <!-- dc:format : lido:eventMaterialsTech//lido:termMaterials NOT @lido:type=material --
>
    <!-- dc:format Resource -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTec
h/lido:materialsTech/lido:termMaterialsTech[not(lower-case(@lido:type)='material')]">
        <xsl:for-each select="lido:conceptID[starts-with(., 'http://') or starts-with(.,
'https://')]">
            <dc:format>
                <xsl:attribute name="rdf:resource">
                    <xsl:value-of select="."/>
                </xsl:attribute>
            </dc:format>
        </xsl:for-each>
    </xsl:for-each>
    <!-- dc:format Resource -->

    <!-- dc:format : lido:eventMaterialsTech//lido:termMaterials NOT @lido:type=material --
>
    <!-- dc:format Resource dereferenced to prefLabel-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTec
h/lido:materialsTech/lido:termMaterialsTech[not(lower-case(@lido:type)='material')]">
        <xsl:for-each select="lido:conceptID[starts-with(.,
'http://partage.vocnet.org/')]">
            <xsl:variable name="partID" select="./text()" />
            <xsl:variable name="labelLang">
                <xsl:choose>
                    <xsl:when
test="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl
:value-of select="$descLang" /></xsl:when>
                    <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
                </xsl:choose>
            </xsl:variable>
            <xsl:variable name="prefLabel"
select="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]"

```



```

/>
        <xsl:if test = "not(..//lido:term) or not(..//lido:term/text() = $prefLabel)">
            <dc:format>
                <xsl:value-of select="$prefLabel" />
            </dc:format>
        </xsl:if>
    </xsl:for-each>
</xsl:for-each>
<!-- dc:format Resource dereferenced to prefLabel -->

<!-- dc:format Literal-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTech[lido:displayMaterialsTech or lido:materialsTech/lido:termMaterialsTech[not(lower-case(@lido:type)='material'])]">
    <xsl:choose>
        <xsl:when test="lido:displayMaterialsTech">
            <dc:format>
                <xsl:value-of select="lido:displayMaterialsTech[1]/text()" />
            </dc:format>
        </xsl:when>
        <xsl:otherwise>
            <xsl:if test = "lido:materialsTech/lido:termMaterialsTech[not(lower-case(@lido:type)='material')]/lido:term[@lido:pref='preferred' or (not(@lido:pref) and not(@lido:addedSearchTerm='yes'))]">
                <dc:format>
                    <xsl:value-of
select="lido:materialsTech/lido:termMaterialsTech[not(lower-case(@lido:type)='material')]/lido:term[@lido:pref='preferred' or (not(@lido:pref) and not(@lido:addedSearchTerm='yes'))]" />
                </dc:format>
            </xsl:if>
        </xsl:otherwise>
    </xsl:choose>
</xsl:for-each>
<!-- dc:format Literal-->

<!-- dc:identifier : lido:objectPublishedID -->
<xsl:for-each select="lido:objectPublishedID">
    <dc:identifier>
        <xsl:value-of select="."/>
    </dc:identifier>
</xsl:for-each>
<!-- dc:identifier -->

<!-- dc:identifier : lido:workID -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:repositoryWrap/lido:repositorySet[not(@lido:type='former')]/lido:workID">
    <dc:identifier>
        <xsl:value-of select="."/>
    </dc:identifier>
</xsl:for-each>
<!-- dc:identifier -->

<!-- dc:identifier -->
<xsl:for-each select="lido:administrativeMetadata/lido:recordWrap/lido:recordID">
    <dc:identifier>
        <xsl:value-of select="."/>
    </dc:identifier>
</xsl:for-each>
<!-- dc:identifier -->

<!-- dc:language : lido:classification / @lido:type=language (MANDATORY with edm:type=TEXT) -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification[@lido:type='language']/lido:term">
    <dc:language>
        <xsl:value-of select="."/>
    </dc:language>
</xsl:for-each>
<!-- dc:language -->

<!-- dc:publisher : lido:eventActor with lido:eventType = publication -->
<!-- dc:publisher Resource-->
<xsl:for-each

```

```

select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
  <xsl:if test="(lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00228')">
    <xsl:for-each select="lido:eventActor">
      <xsl:if test="lido:actorInRole/lido:actor/lido:actorID[starts-with(., 'http://')
or starts-with(., 'https://')]">
        <dc:publisher>
          <xsl:attribute name="rdf:resource">
            <xsl:for-each select="lido:actorInRole/lido:actor/lido:actorID[starts-
with(., 'http://') or starts-with(., 'https://')]">
              <xsl:if test="position() = 1">
                <xsl:value-of select="."/>
              </xsl:if>
            </xsl:for-each>
          </xsl:attribute>
        </dc:publisher>
      </xsl:if>
    </xsl:for-each>
  </xsl:if>
<!-- dc:publisher Resource-->

  <!-- dc:publisher : lido:eventActor with lido:eventType = publication -->
<!-- dc:publisher Resource dereferenced to prefLabel-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
  <xsl:if test="(lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00228')">
    <xsl:for-each select="lido:eventActor">
      <xsl:if test="lido:actorInRole/lido:actor/lido:actorID[starts-with(.,
'http://partage.vocnet.org/')]">
        <xsl:variable name="partID"
select="lido:actorInRole/lido:actor/lido:actorID[1]/text()" />

        <xsl:variable name="labelLang">
          <xsl:choose>
            <xsl:when
test="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl:valu
e-of select="$descLang" /></xsl:when>
            <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
          </xsl:choose>
        </xsl:variable>
        <xsl:variable name="prefLabel"
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]" />
        <!-- <xsl:if test =
"not(lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred'
or not(@lido:pref)]) or
not(lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred'
or not(@lido:pref)]/text() = $prefLabel)"-->
          <dc:publisher>
            <xsl:attribute name="xml:lang" select="$labelLang" />
            <xsl:value-of select="$prefLabel" />
          </dc:publisher>
        <!-- </xsl:if-->
      </xsl:if>
    </xsl:for-each>
  </xsl:if>
<!-- dc:publisher Resource dereferenced to prefLabel-->

  <!-- dc:publisher Literal-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
  <xsl:if test="(lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00228')">
    <xsl:if test="not(lido:eventActor/lido:actorInRole/lido:actor/lido:actorID)"> <!--
FILTER ACTOR NAME -->

    <xsl:if test="lido:eventActor/lido:displayActorInRole">
      <xsl:for-each select="lido:eventActor/lido:displayActorInRole">
        <dc:publisher>
          <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) &gt;
0">
            <xsl:attribute name="xml:lang">
              <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
            </xsl:attribute>

```

```

        </xsl:if>
        <xsl:value-of select="."/>
    </dc:publisher>
</xsl:for-each>
</xsl:if>

<xsl:if
test="lido:eventActor/lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue">
    <xsl:for-each
select="lido:eventActor/lido:actorInRole/lido:actor/lido:nameActorSet/lido:appellationValue">
        <dc:publisher>
            <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)
            &gt; 0)">
                <xsl:attribute name="xml:lang">
                    <xsl:value-of
select="lido:langSelect($descLang,@xml:lang)"/>
                </xsl:attribute>
            </xsl:if>
            <xsl:value-of select="."/>
        </dc:publisher>
    </xsl:for-each>
</xsl:if>
</xsl:if>
</xsl:if>
</xsl:for-each>
<!-- dc:publisher Literal-->

<!-- dc:relation -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
        <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://purl.org/dc/elements/1.1/relation' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote) ">
            <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
                <xsl:if test = "starts-with(.,'http')">
                    <dc:relation>
                        <xsl:attribute name="rdf:resource">
                            <xsl:value-of select = "."/>
                        </xsl:attribute>
                    </dc:relation>
                </xsl:if>
            </xsl:for-each>
            <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
                <dc:relation>
                    <xsl:value-of select = "."/>
                </dc:relation>
            </xsl:for-each>
            <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
                <dc:relation>
                    <xsl:value-of select = "."/>
                </dc:relation>
            </xsl:for-each>
        </xsl:if>
    </xsl:for-each>

<!-- No LIDO property is mapped to dc:coverage of EDM -->

<!-- dc:rights : lido:rightsWorkSet (rights for the CHO) -->
    <xsl:for-each
select="lido:administrativeMetadata/lido:rightsWorkWrap/lido:rightsWorkSet">
        <xsl:choose>
            <xsl:when test="lido:creditLine">
                <dc:rights>
                    <xsl:value-of select="lido:creditLine"/>
                </dc:rights>
            </xsl:when>
            <xsl:when
test="lido:rightsHolder/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or
not (@lido:pref)]">
                <xsl:for-each
select="lido:rightsHolder/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or
not (@lido:pref)]">
                    <dc:rights>
                        <xsl:value-of select="."/>
                    </dc:rights>
                </xsl:for-each>
            </xsl:when>
        </xsl:choose>
    </xsl:for-each>

```

```

        </dc:rights>
      </xsl:for-each>
    </xsl:when>
  </xsl:choose>
</xsl:for-each>

<!-- dc:source -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test =
"lido:relatedWorkRelType/lido:conceptID='http://purl.org/dc/elements/1.1/source' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(.,'http')">
        <dc:source>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </dc:source>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <dc:source>
        <xsl:value-of select = "."/>
      </dc:source>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <dc:source>
        <xsl:value-of select = "."/>
      </dc:source>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

<!-- dc:subject : lido:subjectConcept -->
<!-- dc:subject Resource-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectConcept/lido:conceptID[starts-with(., 'http://') or starts-with(.,
'https://')]">
  <dc:subject>
    <xsl:attribute name="rdf:resource">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </dc:subject>
</xsl:for-each>
<!-- dc:subject Resource-->

<!-- dc:subject : lido:subjectConcept -->
<!-- dc:subject Resource dereferenced to prefLabel-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectConcept/lido:conceptID[starts-with(.,
'http://bib.arts.kuleuven.be/photoVocabulary/')]">
  <xsl:variable name="partID" select="./text()" />

  <xsl:variable name="labelLang">
    <xsl:choose>
      <xsl:when
test="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl
:value-of select="$descLang" /></xsl:when>
      <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name="prefLabel"
select="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]"
/>

  <xsl:if test = "not(../lido:term) or not(../lido:term/text() = $prefLabel)">
    <dc:subject>
      <xsl:attribute name="xml:lang" select="$labelLang" />
      <xsl:value-of select="$prefLabel" />
    </dc:subject>
  </xsl:if>
</xsl:for-each>

```

```

    <!-- dc:subject Resource dereferenced to prefLabel-->

    <!-- dc:subject Literal-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectConcept">
      <xsl:for-each select="lido:term[@lido:pref='preferred' or (not(@lido:pref) and
not(@lido:addedSearchTerm='yes'))]">
        <dc:subject>
          <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) >
0">
            <xsl:attribute name="xml:lang">
              <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
            </xsl:attribute>
          </xsl:if>
          <xsl:value-of select="."/>
        </dc:subject>
      </xsl:for-each>
    </xsl:for-each>
    <!-- dc:subject Literal-->

    <!-- dc:subject : lido:subjectActor -->
    <!-- dc:subject Resource-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectActor/lido:actor/lido:actorID[starts-with(., 'http://') or starts-with(.,
'https://')]">
      <dc:subject>
        <xsl:attribute name="rdf:resource">
          <xsl:value-of select="."/>
        </xsl:attribute>
      </dc:subject>
    </xsl:for-each>
    <!-- dc:subject Resource-->

    <!-- dc:subject : lido:subjectActor -->
    <!-- dc:subject Resource dereferenced to prefLabel-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectActor/lido:actor/lido:actorID[starts-with(.,
'http://partage.vocnet.org/')]">
      <xsl:variable name="partID" select="./text()" />

      <xsl:variable name="labelLang">
        <xsl:choose>
          <xsl:when
test="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl:valu
e-of select="$descLang" /></xsl:when>
          <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
        </xsl:choose>
      </xsl:variable>
      <xsl:variable name="prefLabel"
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]" />
      <!-- <xsl:if test =
"not(..//lido:nameActorSet/lido:appellationValue[@lido:pref='preferred' or not(@lido:pref)]) or
not(..//lido:nameActorSet/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)]/text() = $prefLabel)">-->
        <dc:subject>
          <xsl:attribute name="xml:lang" select="$labelLang" />
          <xsl:value-of select="$prefLabel" />
        </dc:subject>
      <!-- </xsl:if>-->

    </xsl:for-each>
    <!-- dc:subject Resource dereferenced to prefLabel-->

    <!-- dc:subject Literal-->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectActor[lido:displayActor or lido:actor]">
      <xsl:choose>
        <xsl:when test="lido:displayActor">
          <dc:subject>
            <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang))
> 0">
              <xsl:attribute name="xml:lang">

```

```

        <xsl:value-of select="lido:langSelect ($descLang,@xml:lang)"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:value-of select="lido:displayActor[1]/text()" />
  </dc:subject>
</xsl:when>
<xsl:otherwise>
  <xsl:if test="not(lido:actor/lido:actorID)"> <!-- FILTER CONCEPT LABEL -->
>
    <dc:subject>
      <xsl:if test="string-length(
lido:langSelect($descLang,@xml:lang)) > 0">
        <xsl:attribute name="xml:lang">
          <xsl:value-of
select="lido:langSelect($descLang,@xml:lang)"/>
        </xsl:attribute>
      </xsl:if>
      <xsl:value-of select=
"lido:actor/lido:nameActorSet/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)][1]/text()" />
    </dc:subject>
  </xsl:if>
</xsl:otherwise>
</xsl:choose>
</xsl:for-each>
<!-- dc:subject Literal-->

<!-- dc:title : lido:titleSet / @lido:pref=preferred or empty -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:titleWrap/lido:titleSet/lid
o:appellationValue[@lido:pref='preferred' or not(@lido:pref)]">
  <dc:title>
    <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
      <xsl:attribute name="xml:lang">
        <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:value-of select="."/>
  </dc:title>
</xsl:for-each>
<!-- dc:title -->

<!-- dc:type : lido:objectWorkType -->
<!-- dc:type Resource-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:objectWorkTypeWrap/lido:obj
ectWorkType/lido:conceptID[starts-with(., 'http://') or starts-with(., 'https://')]">
  <dc:type>
    <xsl:attribute name="rdf:resource">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </dc:type>
</xsl:for-each>
<!-- dc:type Resource-->

<!-- dc:type Resource dereferenced to prefLabel-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:objectWorkTypeWrap/lido:obj
ectWorkType/lido:conceptID[starts-with(., 'http://partage.vocnet.org/')]">
  <xsl:variable name="partID" select="./text()" />
  <xsl:variable name="labelLang">
    <xsl:choose>
      <xsl:when
test="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl
:value-of select="$descLang" /></xsl:when>
      <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:variable name="prefLabel"
select="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]"
/>
  <xsl:if test = "not(../lido:term) or not(../lido:term/text() = $prefLabel)">
    <dc:type>
      <xsl:attribute name="xml:lang" select="$labelLang" />
      <xsl:value-of select="$prefLabel" />
    </dc:type>
  </xsl:if>

```

```

</xsl:for-each>
  <!-- dc:type Resource dereferenced to prefLabel-->

  <!-- dc:type Literal-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:objectWorkTypeWrap/lido:obj
ectWorkType">
    <xsl:for-each select="lido:term[@lido:pref='preferred' or (not(@lido:pref) and
not(@lido:addedSearchTerm='yes'))]">
      <dc:type>
        <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
          <xsl:attribute name="xml:lang">
            <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
          </xsl:attribute>
        </xsl:if>
        <xsl:value-of select="."/>
      </dc:type>
    </xsl:for-each>
  </xsl:for-each>
  <!-- dc:type Literal-->

  <!-- dc:type : lido:classification -->
  <!-- dc:type Resource-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:cla
ssification/lido:conceptID[starts-with(., 'http://') or starts-with(., 'https://')]">
    <dc:type>
      <xsl:attribute name="rdf:resource">
        <xsl:value-of select="."/>
      </xsl:attribute>
    </dc:type>
  </xsl:for-each>
  <!-- dc:type Resource-->

  <!-- dc:type Resource dereferenced to prefLabel-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:cla
ssification/lido:conceptID[starts-with(., 'http://partage.vocnet.org/')]">
    <xsl:variable name="partID" select="./text()" />
    <xsl:variable name="labelLang">
      <xsl:choose>
        <xsl:when
test="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl
:value-of select="$descLang" /></xsl:when>
        <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
      </xsl:choose>
    </xsl:variable>
    <xsl:variable name="prefLabel"
select="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]"
/>
    <xsl:if test = "not(../lido:term) or not(../lido:term/text() = $prefLabel)">
      <dc:type>
        <xsl:attribute name="xml:lang" select="$labelLang" />
        <xsl:value-of select="$prefLabel" />
      </dc:type>
    </xsl:if>
  </xsl:for-each>
  <!-- dc:type Resource dereferenced to prefLabel-->

  <!-- dc:type Literal-->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:cla
ssification[not(@lido:type='language')
and not (@lido:type='europeana:project') and not(lido:term[(. = 'IMAGE') or (. =
'VIDEO') or (. = 'TEXT') or (. = '3D') or (. = 'SOUND')])]">
    <xsl:for-each select="lido:term[@lido:pref='preferred' or (not(@lido:pref) and
not(@lido:addedSearchTerm='yes'))]">
      <dc:type>
        <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
          <xsl:attribute name="xml:lang">
            <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
          </xsl:attribute>
        </xsl:if>
        <xsl:value-of select="."/>
      </dc:type>
    </xsl:for-each>

```

```

</xsl:for-each>
<!-- dc:type Literal-->

<!-- dcterms:alternative : lido:titleSet / @lido:pref=alternative -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:titleWrap/lido:titleSet/lid
o:appellationValue[starts-with(@lido:pref, 'alternat')]">
  <dcterms:alternative>
    <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
      <xsl:attribute name="xml:lang">
        <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:value-of select="."/>
  </dcterms:alternative>
</xsl:for-each>
<!-- dcterms:alternative -->

<!-- dct:conformsTo -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID='http://purl.org/dc/terms/' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or lido:relatedWork/lido:object/lido:objectNote) ">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(., 'http')">
        <dcterms:conformsTo>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </dcterms:conformsTo>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <dcterms:conformsTo>
        <xsl:value-of select = "."/>
      </dcterms:conformsTo>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <dcterms:conformsTo>
        <xsl:value-of select = "."/>
      </dcterms:conformsTo>
    </xsl:for-each>

    </xsl:if>
  </xsl:for-each>

  <!-- dcterms:created : lido:eventDate with lido:eventType = production or creation or
designing -->
  <!-- dcterms:created -->
  <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event">
    <xsl:if test="(lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00007') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00012') or (lido:eventType/lido:conceptID = 'http://terminology.lido-
schema.org/lido00224') ">
      <xsl:for-each select="lido:eventDate">
        <xsl:if test="lido:date/lido:earliestDate | lido:displayDate">
          <dcterms:created>
            <xsl:choose>
              <xsl:when test="lido:date/lido:earliestDate = lido:date/lido:latestDate">
                <xsl:value-of select="lido:date/lido:earliestDate"/>
              </xsl:when>
              <xsl:when test="lido:date/lido:earliestDate">
                <xsl:value-of select="concat(lido:date/lido:earliestDate, '/',
lido:date/lido:latestDate)"/>
              </xsl:when>
              <xsl:otherwise>
                <xsl:value-of select="lido:displayDate" />
              </xsl:otherwise>
            </xsl:choose>
          </dcterms:created>
        </xsl:if>
      </xsl:for-each>
    </xsl:if>
  </xsl:if>

```



```

</xsl:for-each>
<!-- dcterms:created -->

<!-- dcterms:extent : lido:objectMeasurementsSet -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:objectMeasurementsWrap/lido
:objectMeasurementsSet[lido:displayObjectMeasurements or
lido:objectMeasurements/lido:measurementsSet]">
  <dcterms:extent>
    <xsl:choose>
      <xsl:when test="lido:displayObjectMeasurements"><xsl:value-of
select="lido:displayObjectMeasurements[1]/text()"/></xsl:when>
      <xsl:otherwise>
        <xsl:for-each
select="lido:objectMeasurements/lido:measurementsSet[string-
length(lido:measurementValue)>0]">
          <xsl:value-of select="concat(lido:measurementType, ': ',
lido:measurementValue, ' ', lido:measurementUnit)"/>
          <xsl:for-each select="../lido:extentMeasurements[string-
length(.)>0]"><xsl:value-of select="concat(' (', .., ')')"/></xsl:for-each>
        </xsl:for-each>
      </xsl:otherwise>
    </xsl:choose>
  </dcterms:extent>
</xsl:for-each>
<!-- dcterms:extent -->

<!-- No LIDO property is mapped to the following properties of EDM -->
<!-- dcterms:hasFormat -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://purl.org/dc/terms/hasFormat' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(.,'http')">
        <dcterms:hasFormat>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "./"/>
          </xsl:attribute>
        </dcterms:hasFormat>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <dcterms:hasFormat>
        <xsl:value-of select = "./"/>
      </dcterms:hasFormat>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <dcterms:hasFormat>
        <xsl:value-of select = "./"/>
      </dcterms:hasFormat>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

<!-- dcterms:hasPart -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://purl.org/dc/terms/hasPart' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "lido:relatedWork/lido:object/lido:objectWebResource[starts-
with(.,'http')]">
        <dcterms:hasPart>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "./"/>
          </xsl:attribute>
        </dcterms:hasPart>
      </xsl:if>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

```

```

</xsl:if>
</xsl:for-each>
<xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
  <dcterms:hasPart>
    <xsl:value-of select = "."/>
  </dcterms:hasPart>
</xsl:for-each>
<xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
  <dcterms:hasPart>
    <xsl:value-of select = "."/>
  </dcterms:hasPart>
</xsl:for-each>
</xsl:if>
</xsl:for-each>
<!-- dcterms:hasVersion -->
<!-- dcterms:isFormatOf -->
<!-- dcterms:isPartOf -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://purl.org/dc/terms/isPartOf' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(., 'http')">
        <dcterms:isPartOf>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </dcterms:isPartOf>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <dcterms:isPartOf>
        <xsl:value-of select = "."/>
      </dcterms:isPartOf>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <dcterms:isPartOf>
        <xsl:value-of select = "."/>
      </dcterms:isPartOf>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>
<!-- dcterms:isReferencedBy -->
<!-- dcterms:isReplacedBy -->
<!-- dcterms:isReplacedBy -->
<!-- dcterms:RequiredBy -->
<!-- dcterms:issued -->
<!-- dcterms:isVersionOf -->

<!-- dcterms:medium : lido:eventMaterialsTech//lido:termMaterials / @lido:type=material
-->
<!-- dcterms:medium Resource-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTec
h/lido:materialsTech/lido:termMaterialsTech[lower-
case(@lido:type)='material']/lido:conceptID[starts-with(., 'http://') or starts-with(.,
'https://')]">
  <dcterms:medium>
    <xsl:attribute name="rdf:resource">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </dcterms:medium>
</xsl:for-each>
<!-- dcterms:medium Resource-->

<!-- dcterms:medium Resource dereferenced to prefLabel-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTec
h/lido:materialsTech/lido:termMaterialsTech[lower-
case(@lido:type)='material']/lido:conceptID[starts-with(., 'http://partage.vocnet.org/')]">
  <xsl:variable name="partID" select="./text()" />

```

```

        <xsl:variable name="labelLang">
          <xsl:choose>
            <xsl:when
test="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$descLang]"><xsl
:value-of select="$descLang" /></xsl:when>
            <xsl:otherwise><xsl:value-of select="'en'" /></xsl:otherwise>
          </xsl:choose>
        </xsl:variable>
        <xsl:variable name="prefLabel"
select="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel[@xml:lang=$labelLang]"
/>
        <xsl:if test = "not(..//lido:term) or not(..//lido:term/text() = $prefLabel)">
          <dcterms:medium>
            <xsl:attribute name="xml:lang" select="$labelLang" />
            <xsl:value-of select="$prefLabel" />
          </dcterms:medium>
        </xsl:if>
      </xsl:for-each>
      <!-- dcterms:medium Resource dereferenced to prefLabel-->
      <!-- dcterms:medium Literal (display -> dc:format, see above) -->
      <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTec
h/lido:materialsTech/lido:termMaterialsTech[lower-case(@lido:type)='material']">
        <xsl:for-each select="lido:term[@lido:pref='preferred' or (not(@lido:pref) and
not(@lido:addedSearchTerm='yes'))]">
          <dcterms:medium>
            <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang)) > 0">
              <xsl:attribute name="xml:lang">
                <xsl:value-of select="lido:langSelect($descLang,@xml:lang)" />
              </xsl:attribute>
            </xsl:if>
            <xsl:value-of select="."/>
          </dcterms:medium>
        </xsl:for-each>
      </xsl:for-each>
      <!-- dcterms:medium Literal-->
      <!-- dcterms:provenance : lido:repositorySet -> lido:repositoryName and/or
lido:repositoryLocation -->
      <!-- First approach was:
IF NOT lido:repositoryName THEN dcterms:spatial :
lido:repositorySet/lido:repositoryLocation
changed by request of Europeana
-->
      <xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:repositoryWrap/lido:reposit
orySet[@lido:type='current']">
        <xsl:if
test="lido:repositoryName/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)] or
lido:repositoryLocation/lido:namePlaceSet/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)]">
          <xsl:choose>
            <xsl:when
test="lido:repositoryName/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)] and
lido:repositoryLocation/lido:namePlaceSet/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)]">
              <dcterms:provenance>
                <xsl:value-of
select="concat(lido:repositoryName[1]/lido:legalBodyName[1]/lido:appellationValue[@lido:pref='p
referred' or not(@lido:pref)][1], ', ',
lido:repositoryLocation[1]/lido:namePlaceSet[1]/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)][1])" />
              </dcterms:provenance>
            </xsl:when>
            <xsl:when
test="lido:repositoryName/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)]">
              <dcterms:provenance>
                <xsl:value-of
select="lido:repositoryName/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)][1]" />
            </xsl:when>
          </xsl:choose>
        </xsl:for-each>
      </xsl:for-each>

```

```

        </dcterms:provenance>
      </xsl:when>
    <xsl:when
test="lido:repositoryLocation/lido:namePlaceSet/lido:appellationValue[@lido:pref='preferred' or
not (@lido:pref)]">
      <dcterms:provenance>
        <xsl:value-of
select="lido:repositoryLocation/lido:namePlaceSet/lido:appellationValue[@lido:pref='preferred'
or not (@lido:pref)][1]"/>
      </dcterms:provenance>
    </xsl:when>
  </xsl:choose>
</xsl:if>
</xsl:for-each>
<!-- dcterms:provenance -->

<!-- dct:references -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://purl.org/dc/terms/references' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(., 'http')">
        <dcterms:references>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </dcterms:references>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <dcterms:references>
        <xsl:value-of select = "."/>
      </dcterms:references>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <dcterms:references>
        <xsl:value-of select = "."/>
      </dcterms:references>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

<!-- dcterms:spatial : lido:eventPlace from any lido:eventSet -->
<!-- dcterms:spatial Resource-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventPlace/lido:p
lace/lido:placeID[starts-with(., 'http://') or starts-with(., 'https://')]">
  <dcterms:spatial>
    <xsl:attribute name="rdf:resource">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </dcterms:spatial>
</xsl:for-each>
<!-- dcterms:spatial Resource-->
<!-- dcterms:spatial Literal-->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventPlace[lido:d
isplayPlace or lido:place/lido:namePlaceSet/lido:appellationValue]">
  <dcterms:spatial>
    <xsl:choose>
      <xsl:when test="lido:displayPlace"><xsl:value-of
select="lido:displayPlace"/></xsl:when>
      <xsl:otherwise><xsl:value-of
select="lido:place/lido:namePlaceSet/lido:appellationValue[1]/text()" /></xsl:otherwise>
    </xsl:choose>
  </dcterms:spatial>

</xsl:for-each>
<!-- dcterms:spatial Literal-->

<!-- dcterms:spatial : lido:subjectPlace -->

```

```

    <!-- dct:terms:spatial Resource -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectPlace/lido:place/lido:placeID[starts-with(., 'http://') or starts-with(.,
'https://')]">
      <dct:terms:spatial>
        <xsl:attribute name="rdf:resource">
          <xsl:value-of select="."/>
        </xsl:attribute>
      </dct:terms:spatial>
    </xsl:for-each>
    <!-- dct:terms:spatial Resource -->
    <!-- dct:terms:spatial Literal -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectPlace[lido:displayPlace or
lido:place/lido:namePlaceSet/lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)]]">
      <xsl:choose>
        <xsl:when test="lido:displayPlace">
          <dct:terms:spatial>
            <xsl:if test="string-length( lido:langSelect($descLang,@xml:lang))
            &gt; 0">
              <xsl:attribute name="xml:lang">
                <xsl:value-of select="lido:langSelect($descLang,@xml:lang)"/>
              </xsl:attribute>
            </xsl:if>
            <xsl:value-of select="lido:displayPlace[1]/text()" />
          </dct:terms:spatial>
        </xsl:when>
        <xsl:otherwise>
          <dct:terms:spatial>
            <xsl:if test="string-length(
lido:langSelect($descLang,@xml:lang)) &gt; 0">
              <xsl:attribute name="xml:lang">
                <xsl:value-of
select="lido:langSelect($descLang,@xml:lang)"/>
              </xsl:attribute>
            </xsl:if>
            <xsl:value-of select=
"lido:place/lido:namePlaceSet//lido:appellationValue[@lido:pref='preferred' or
not(@lido:pref)][1]/text()" />
          </dct:terms:spatial>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:for-each>
    <!-- dct:terms:spatial Literal -->

    <!-- No LIDO property is mapped to the following properties of EDM -->
    <!-- dct:tableOfContents -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:objectDescriptionWrap/lido:
objectDescriptionSet[@lido:type='tableOfContents']/lido:descriptiveNoteValue">
      <dct:terms:tableOfContents>
        <xsl:value-of select="."/>
      </dct:terms:tableOfContents>
    </xsl:for-each>

    <!-- dct:temporal -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:
subject/lido:subjectDate">
      <xsl:if test="lido:date/lido:earliestDate | lido:displayDate">
        <dct:terms:temporal>
          <xsl:choose>
            <xsl:when test="lido:date/lido:earliestDate = lido:date/lido:latestDate">
              <xsl:value-of select="lido:date/lido:earliestDate"/>
            </xsl:when>
            <xsl:when test="lido:date/lido:earliestDate">
              <xsl:value-of select="concat(lido:date/lido:earliestDate, '/',
lido:date/lido:latestDate)"/>
            </xsl:when>
            <xsl:otherwise>
              <xsl:value-of select="lido:displayDate" />
            </xsl:otherwise>
          </xsl:choose>
        </dct:terms:temporal>
      </xsl:if>
    </xsl:for-each>

```

```

</xsl:if>
</xsl:for-each>
<!-- dc:date -->

<xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification[@lido:type='temporal']/lido:term">
  <dcterms:temporal>
    <xsl:value-of select="."/>
  </dcterms:temporal>
</xsl:for-each>

<!-- edm:currentLocation -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectIdentificationWrap/lido:repositoryWrap/lido:repositorySet[@lido:type='current']/lido:repositoryLocation/lido:placeID[@lido:type='URI']">
  <edm:currentLocation>
    <xsl:attribute name="rdf:resource">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </edm:currentLocation>
</xsl:for-each>

<!-- edm:incorporates -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://www.europeana.eu/schemas/edm/incorporates' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote) ">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(.,'http')">
        <edm:incorporates>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </edm:incorporates>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <edm:incorporates>
        <xsl:value-of select = "."/>
      </edm:incorporates>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <edm:incorporates>
        <xsl:value-of select = "."/>
      </edm:incorporates>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

<!-- edm:isDerivativeOf -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWorkSet">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://www.europeana.eu/schemas/edm/isDerivativeOf' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote) ">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(.,'http')">
        <edm:isDerivativeOf>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </edm:isDerivativeOf>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <edm:isDerivativeOf>
        <xsl:value-of select = "."/>
      </edm:isDerivativeOf>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

```

```

</xsl:for-each>
<xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
  <edm:isDerivativeOf>
    <xsl:value-of select = "."/>
  </edm:isDerivativeOf>
</xsl:for-each>
</xsl:if>
</xsl:for-each>

<!-- edm:isNextInSequence -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://www.europeana.eu/schemas/edm/isNextInSequence' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(., 'http')">
        <edm:isNextInSequence>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </edm:isNextInSequence>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <edm:isNextInSequence>
        <xsl:value-of select = "."/>
      </edm:isNextInSequence>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <edm:isNextInSequence>
        <xsl:value-of select = "."/>
      </edm:isNextInSequence>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

<!-- edm:isRelatedTo -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:objectRelationWrap/lido:relatedWorksWrap/lido:relatedWork
Set">
  <xsl:if test = "lido:relatedWorkRelType/lido:conceptID =
'http://www.europeana.eu/schemas/edm/isRelatedTo' and
(lido:relatedWork/lido:object/lido:objectWebResource or
lido:relatedWork/lido:object/lido:objectID or
lido:relatedWork/lido:object/lido:objectNote)">
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectWebResource">
      <xsl:if test = "starts-with(., 'http')">
        <edm:isRelatedTo>
          <xsl:attribute name="rdf:resource">
            <xsl:value-of select = "."/>
          </xsl:attribute>
        </edm:isRelatedTo>
      </xsl:if>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectID">
      <edm:isRelatedTo>
        <xsl:value-of select = "."/>
      </edm:isRelatedTo>
    </xsl:for-each>
    <xsl:for-each select = "lido:relatedWork/lido:object/lido:objectNote">
      <edm:isRelatedTo>
        <xsl:value-of select = "."/>
      </edm:isRelatedTo>
    </xsl:for-each>
  </xsl:if>
</xsl:for-each>

<!-- edm:isRepresentationOf -->
<!-- edm:isSimilarTo -->
<!-- edm:isSuccessorOf -->
<!-- edm:realizes -->

<!-- edm:type : lido:classification / @lido:type=europeana:type -->

```

```

    <xsl:if
test="(lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification/lido:term = 'IMAGE') or
(lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification/lido:term = 'VIDEO') or
(lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification/lido:term = 'TEXT') or
(lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification/lido:term = '3D') or
(lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification/lido:term = 'SOUND')">
    <xsl:for-each
select="lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification/lido:term[(. = 'IMAGE') or (. = 'VIDEO') or (. = 'TEXT') or (. = '3D') or (. = 'SOUND')]">
        <xsl:if test="position() = 1">
            <xsl:if test="index-of($var0/item, normalize-space()) > 0">
                <edm:type>
                    <xsl:value-of select="."/>
                </edm:type>
            </xsl:if>
        </xsl:if>
    </xsl:for-each>
</xsl:if>

    <!-- owl:sameAs -->
    <!-- No LIDO property is mapped to owl:sameAs of EDM -->
</edm:ProvidedCHO>

    <!-- edm:WebResource : lido:resourceSet -->
    <xsl:for-each
select="lido:administrativeMetadata/lido:resourceWrap/lido:resourceSet[lido:resourceRepresentation/lido:linkResource]">
        <xsl:if test="lido:resourceRepresentation/lido:linkResource[starts-with(., 'http://') or starts-with(., 'https://')]">
            <xsl:for-each select="lido:resourceRepresentation">
                <edm:WebResource>
                    <xsl:attribute name="rdf:about">
                        <xsl:for-each select="lido:linkResource[starts-with(., 'http://') or starts-with(., 'https://')]">
                            <xsl:if test="position() = 1">
                                <xsl:value-of select="."/>
                            </xsl:if>
                        </xsl:for-each>
                    </xsl:attribute>
                    <xsl:for-each select="lido:rightsResource">
                        <xsl:choose>
                            <xsl:when test="lido:creditLine">
                                <dc:rights>
                                    <xsl:value-of select="lido:creditLine"/>
                                </dc:rights>
                            </xsl:when>
                            <xsl:when
test="lido:rightsHolder/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or not(@lido:pref)]">
                                <xsl:for-each
select="lido:rightsHolder/lido:legalBodyName/lido:appellationValue[@lido:pref='preferred' or not(@lido:pref)]">
                                    <dc:rights>
                                        <xsl:value-of select="."/>
                                    </dc:rights>
                                </xsl:for-each>
                            </xsl:when>
                        </xsl:choose>
                    </xsl:for-each>
                </xsl:for-each select="lido:resourceMeasurementsSet">
                    <xsl:if test = "lido:measurementType =
'http://www.europeana.eu/schemas/edm/isNextInSequence' and
lido:measurementUnit = 'URI'">
                        <edm:isNextInSequence>
                            <xsl:attribute name="rdf:resource">
                                <xsl:value-of select="lido:measurementValue"/>
                            </xsl:attribute>
                        </edm:isNextInSequence>
                    </xsl:if>

```



```

        </xsl:for-each>
    </edm:WebResource>
</xsl:for-each>
</xsl:if>
</xsl:for-each>
<!-- edm:WebResource -->

    <!-- edm:WebResource : lido:recordInfoLink -->
    <xsl:for-each
select="lido:administrativeMetadata/lido:recordWrap/lido:recordInfoSet/lido:recordInfoLink[starts-with(., 'http://') or starts-with(., 'https://')]">
        <edm:WebResource>
            <xsl:attribute name="rdf:about">
                <xsl:if test="position() = 1">
                    <xsl:value-of select="."/>
                </xsl:if>
            </xsl:attribute>
        </edm:WebResource>
    </xsl:for-each>
    <!-- edm:WebResource -->

    <!-- edm:Agent : lido:eventActor or lido:subjectActor -->
    <xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventActor/lido:actorInRole |
lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:subject/lido:subjectActor">
        <xsl:if test="lido:actor/lido:actorID[starts-with(., 'http://') or starts-with(., 'https://')]">
            <edm:Agent>
                <xsl:attribute name="rdf:about">
                    <xsl:for-each select="lido:actor/lido:actorID[starts-with(., 'http://') or starts-with(., 'https://')]">
                        <xsl:if test="position() = 1">
                            <xsl:value-of select="."/>
                        </xsl:if>
                    </xsl:for-each>
                </xsl:attribute>
                <!-- include prefLabel from Partage creator authority / xml:lang="en" -->
                <xsl:for-each select="lido:actor/lido:actorID[starts-with(., 'http://partage.vocnet.org')]">
                    <xsl:variable name="partID" select=".[1]/text()" />
                    <xsl:if test="position() = 1">
                        <xsl:for-each
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:prefLabel">
                            <!-- prefLabel -->
                            <skos:prefLabel>
                                <xsl:attribute name="xml:lang" select="@xml:lang" />
                                <xsl:value-of select="."/>
                            </skos:prefLabel>
                        </xsl:for-each>
                        <xsl:for-each
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:altLabel">
                            <!-- altLabel -->
                            <skos:altLabel>
                                <xsl:attribute name="xml:lang" select="@xml:lang" />
                                <xsl:value-of select="."/>
                            </skos:altLabel>
                        </xsl:for-each>
                        <!-- broader -->
                        <xsl:for-each
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:broader/@rdf:resource">
                            <skos:broader>
                                <xsl:attribute name="rdf:resource" select="." />
                            </skos:broader>
                        <!-- narrower -->
                        </xsl:for-each>
                        <xsl:for-each
select="$PPActors/skos:Concept[@rdf:about=$partID]/skos:narrower/@rdf:resource">
                            <skos:narrower>
                                <xsl:attribute name="rdf:resource" select="." />
                            </skos:narrower>
                        </xsl:for-each>
                    </xsl:if>
                </xsl:for-each>
            </edm:Agent>
        </xsl:if>
    </xsl:for-each>

```

```

</xsl:for-each>
<!-- edm:Agent -->

<!-- edm:Place : lido:eventPlace or lido:subjectPlace -->
<xsl:for-each
select="lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventPlace |
lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:subject/
lido:subjectPlace">
  <xsl:if test="lido:place/lido:placeID[starts-with(., 'http://') or starts-with(.,
'https://')]">
    <edm:Place>
      <xsl:attribute name="rdf:about">
        <xsl:for-each select="lido:place/lido:placeID[starts-with(., 'http://') or
starts-with(., 'https://')]">
          <xsl:if test="position() = 1">
            <xsl:value-of select="."/>
          </xsl:if>
        </xsl:for-each>
      </xsl:attribute>
    </edm:Place>
  </xsl:if>
</xsl:for-each>
<!-- edm:Place -->

<!-- edm:TimeSpan -->
<!-- No LIDO property is mapped to this EDM class -->

<!-- skos:Concept : lido:objectWorkType or lido:classification or lido:termMaterialsTech
or lido:culture or lido:subjectConcept -->
<xsl:for-each select="
lido:descriptiveMetadata/lido:objectClassificationWrap/lido:objectWorkTypeWrap/lido:objectWorkT
ype |
lido:descriptiveMetadata/lido:objectClassificationWrap/lido:classificationWrap/lido:classification[not(starts-with(@lido:type, 'europeana'))] |
lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:eventMaterialsTech/lido:m
aterialsTech/lido:termMaterialsTech |
lido:descriptiveMetadata/lido:eventWrap/lido:eventSet/lido:event/lido:culture |
lido:descriptiveMetadata/lido:objectRelationWrap/lido:subjectWrap/lido:subjectSet/lido:subject/
lido:subjectConcept
">
  <xsl:if test="lido:conceptID[starts-with(., 'http://') or starts-with(., 'https://')]">
    <skos:Concept>
      <xsl:attribute name="rdf:about">
        <xsl:for-each select="lido:conceptID[starts-with(., 'http://') or starts-with(.,
'https://')]">
          <xsl:if test="position() = 1">
            <xsl:value-of select="."/>
          </xsl:if>
        </xsl:for-each>
      </xsl:attribute>
      <!-- include prefLabel from Partage vocabulary -->
      <xsl:for-each select="lido:conceptID[starts-with(., 'http://partage.vocnet.org')]">
        <xsl:variable name="partID" select=".[1]/text()" />
        <xsl:if test="position() = 1">
          <!-- prefLabel -->
          <xsl:for-each select
="{$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:prefLabel">
            <skos:prefLabel>
              <xsl:attribute name="xml:lang" select="@xml:lang" />
              <xsl:value-of select="." />
            </skos:prefLabel>
          </xsl:for-each>
          <!-- altLabel -->
          <xsl:for-each
select="{$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:altLabel">
            <skos:altLabel>
              <xsl:attribute name="xml:lang" select="@xml:lang" />
              <xsl:value-of select="." />
            </skos:altLabel>
          </xsl:for-each>
          <!-- broader -->
          <xsl:for-each
select="{$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:broader/@rdf:resource">

```

```

    <skos:broader>
      <xsl:attribute name="rdf:resource" select="." />
    </skos:broader>
    <!-- narrower -->
  </xsl:for-each>
  <xsl:for-each
select="$PPTerminology/skos:Concept[@rdf:about=$partID]/skos:narrower/@rdf:resource">
    <skos:narrower>
      <xsl:attribute name="rdf:resource" select="." />
    </skos:narrower>
  </xsl:for-each>
  </xsl:if>
</xsl:for-each>
</skos:Concept>
</xsl:if>
</xsl:for-each>
<!-- skos:Concept -->

<!-- ore:Aggregation -->
<ore:Aggregation>
  <xsl:attribute name="rdf:about">
    <xsl:for-each select="lido:administrativeMetadata/lido:recordWrap/lido:recordID">
      <xsl:if test="position() = 1">
        <xsl:value-of select="concat($ore_Aggregation, $dataProvider, '/')"/>
      </xsl:if>
    </xsl:for-each>
  </xsl:attribute>

  <!-- edm:aggregatedCHO -->
  <edm:aggregatedCHO>
    <xsl:attribute name="rdf:resource">
      <xsl:for-each select="lido:administrativeMetadata/lido:recordWrap/lido:recordID">
        <xsl:if test="position() = 1">
          <xsl:value-of select="concat($edm_providedCHO, $dataProvider, '/')"/>
        </xsl:if>
      </xsl:for-each>
    </xsl:attribute>
  </edm:aggregatedCHO>
  <!-- edm:aggregatedCHO -->

  <!-- edm:dataProvider : lido:recordSource -->
  <xsl:choose>
    <xsl:when
test="lido:administrativeMetadata/lido:recordWrap/lido:recordSource[lido:type='europeana:dataPr
ovider']">
      <xsl:for-each
select="lido:administrativeMetadata/lido:recordWrap/lido:recordSource[lido:type='europeana:data
Provider']/lido:legalBodyName/lido:appellationValue">
        <xsl:if test="position() = 1">
          <edm:dataProvider>
            <xsl:value-of select="." />
          </edm:dataProvider>
        </xsl:if>
      </xsl:for-each>
    </xsl:when>
    <xsl:otherwise>
      <xsl:for-each
select="lido:administrativeMetadata/lido:recordWrap/lido:recordSource/lido:legalBodyName/lido:a
ppellationValue">
        <xsl:if test="position() = 1">
          <edm:dataProvider>
            <xsl:value-of select="." />
          </edm:dataProvider>
        </xsl:if>
      </xsl:for-each>
    </xsl:otherwise>
  </xsl:choose>
  <!-- edm:dataProvider -->

  <!-- edm:hasView : lido:resourceRepresentation / @lido:type=image_master or empty -->
  <xsl:for-each
select="lido:administrativeMetadata/lido:resourceWrap/lido:resourceSet/lido:resourceRepresentat
ion[@lido:type='image_master' or not(@lido:type) ]">
    <xsl:if test="position() > 1">
      <xsl:for-each select="lido:linkResource[starts-with(., 'http://') or starts-with(.,
'https://') or starts-with(., 'ftp://')]">
        <edm:hasView>

```

```

        <xsl:attribute name="rdf:resource">
        <xsl:value-of select="."/>
        </xsl:attribute>
    </edm:hasView>
</xsl:for-each>
</xsl:if>
</xsl:for-each>
<!-- edm:hasView -->

<!-- edm:hasView : lido:resourceRepresentation / @lido:type=image_master or empty -->
    <xsl:for-each
select="lido:administrativeMetadata/lido:resourceWrap/lido:resourceSet/lido:resourceRepresentat
ion[@lido:type='image_thumb']">
        <xsl:if test="position() > 1">
            <xsl:for-each select="lido:linkResource[starts-with(., 'http://') or starts-with(.,
'https://') or starts-with(., 'ftp://')]">
                <edm:hasView>
                    <xsl:attribute name="rdf:resource">
                    <xsl:value-of select="."/>
                    </xsl:attribute>
                </edm:hasView>
            </xsl:for-each>
        </xsl:if>
    </xsl:for-each>
<!-- edm:hasView -->

    <!-- edm:isShownAt : lido:recordInfoLink -->
        <xsl:if
test="lido:administrativeMetadata/lido:recordWrap/lido:recordInfoSet/lido:recordInfoLink[starts
-with(., 'http://') or starts-with(., 'https://')]">
            <xsl:for-each
select="lido:administrativeMetadata/lido:recordWrap/lido:recordInfoSet/lido:recordInfoLink">
                <edm:isShownAt>
                    <xsl:attribute name="rdf:resource">
                    <xsl:if test="position() = 1">
                        <xsl:value-of select="."/>
                    </xsl:if>
                    </xsl:attribute>
                </edm:isShownAt>
            </xsl:for-each>
        </xsl:if>
    <!-- edm:isShownAt -->

    <!-- edm:isShownBy : lido:resourceRepresentation / @lido:type=image_master or empty --
>
        <xsl:for-each
select="lido:administrativeMetadata/lido:resourceWrap/lido:resourceSet/lido:resourceRepresentat
ion[@lido:type='image_master' or not(@lido:type)]">
            <xsl:if test="position() = 1">
                <xsl:for-each select="lido:linkResource[starts-with(., 'http://') or starts-with(.,
'https://') or starts-with(., 'ftp://')]">
                    <edm:isShownBy>
                        <xsl:attribute name="rdf:resource">
                        <xsl:value-of select="."/>
                        </xsl:attribute>
                    </edm:isShownBy>
                </xsl:for-each>
            </xsl:if>
        </xsl:for-each>
    <!-- edm:isShownBy -->

    <!-- edm:object -->
        <xsl:for-each
select="lido:administrativeMetadata/lido:resourceWrap/lido:resourceSet/lido:resourceRepresentat
ion[@lido:type='image_thumb']">
            <xsl:if test="position() = 1">
                <xsl:for-each select="lido:linkResource[starts-with(., 'http://') or starts-with(.,
'https://') ]">
                    <edm:object>
                        <xsl:attribute name="rdf:resource">
                        <xsl:value-of select="."/>
                        </xsl:attribute>
                    </edm:object>
                </xsl:for-each>
            </xsl:if>
        </xsl:for-each>
    </xsl:if>

```

```

</xsl:for-each>
<!-- edm:object -->

<!-- edm:provider -->
<edm:provider>
  <xsl:value-of select="$edm_provider" />
</edm:provider>
<!-- edm:provider -->

<!-- edm:rights : lido:rightsResource (MANDATORY, URI taken from a set of URIs defined
for use in Europeana) -->
  <xsl:for-each
select="lido:administrativeMetadata/lido:resourceWrap/lido:resourceSet/lido:rightsResource/lido
:rightsType[lido:conceptID[starts-with(., 'http://creativecommons.org/') or starts-with(.,
'http://www.europeana.eu/rights')] or lido:term[starts-with(., 'http://creativecommons.org/')
or starts-with(., 'http://www.europeana.eu/rights')]]">
    <xsl:if test="position() = 1">
      <edm:rights>
        <xsl:attribute name="rdf:resource">
          <xsl:choose>
            <xsl:when test="lido:conceptID[starts-with(.,
'http://creativecommons.org/') or starts-with(., 'http://www.europeana.eu/rights')]">
              <xsl:value-of select="lido:conceptID[starts-with(.,
'http://creativecommons.org/') or starts-with(.,
'http://www.europeana.eu/rights')][1]/text()" />
            </xsl:when>
            <xsl:when test="lido:term[starts-with(.,
'http://creativecommons.org/') or starts-with(., 'http://www.europeana.eu/rights')]">
              <xsl:value-of select="lido:term[starts-with(.,
'http://creativecommons.org/') or starts-with(.,
'http://www.europeana.eu/rights')][1]/text()" />
            </xsl:when>
          </xsl:choose>
        </xsl:attribute>
      </edm:rights>
    </xsl:if>
  </xsl:for-each>
  <!-- edm:rights -->
</ore:Aggregation>
<!-- ore:Aggregation -->
</xsl:for-each>
</rdf:RDF>
</xsl:template>

<xsl:variable name="PPTerminology">
<skos:Concept rdf:about="http://partage.vocnet.org/part00036">
  <skos:prefLabel xml:lang="en">works on paper</skos:prefLabel>
  <skos:altLabel xml:lang="de">Kunst auf Papier</skos:altLabel>
  <skos:prefLabel xml:lang="sv">konst på papper</skos:prefLabel>
  <skos:prefLabel xml:lang="fi">paperityöt</skos:prefLabel>
  <skos:altLabel xml:lang="es">obras en papel</skos:altLabel>
  <skos:prefLabel xml:lang="es">obras sobre papel</skos:prefLabel>
  <skos:prefLabel xml:lang="sl">dela na papirju</skos:prefLabel>
  <skos:prefLabel xml:lang="pt">trabalho sobre papel</skos:prefLabel>
  <skos:prefLabel xml:lang="pl">prace na papierze</skos:prefLabel>
  <skos:prefLabel xml:lang="nl">werken op papier</skos:prefLabel>
  <skos:prefLabel xml:lang="it">oggetti di carta</skos:prefLabel>
  <skos:prefLabel xml:lang="cs">práce na papíře</skos:prefLabel>
  <skos:prefLabel xml:lang="ca">obra sobre paper</skos:prefLabel>
  <skos:altLabel xml:lang="fr">œuvre sur papier</skos:prefLabel>
  <skos:prefLabel xml:lang="hu">papír alapú tárgy</skos:prefLabel>
  <skos:prefLabel xml:lang="no">arbeid på papir</skos:prefLabel>
  <skos:prefLabel xml:lang="hr">radovi na papiru</skos:prefLabel>
  <skos:prefLabel xml:lang="de">Werk auf Papier</skos:prefLabel>
  <skos:altLabel xml:lang="de">Arbeiten auf Papier (visuelle Werke)</skos:altLabel>
  <skos:altLabel xml:lang="de">Werke auf Papier</skos:altLabel>
  <skos:altLabel xml:lang="de">Graphik (visuelles Werk)</skos:altLabel>
  <skos:altLabel xml:lang="de">Grafik (visuelles Werk)</skos:altLabel>
  <skos:inScheme rdf:resource="http://partage.vocnet.org"/>
  <skos:broader rdf:resource="http://partage.vocnet.org/part01377"/>
  <skos:exactMatch rdf:resource="http://vocab.getty.edu/aat/300189621"/>
  <skos:closeMatch rdf:resource="http://d-nb.info/gnd/4395805-9"/>
  <skos:editorialNote xml:lang="en">PPObjectType</skos:editorialNote>
</skos:Concept>
</xsl:variable>

<xsl:variable name="PPActors">

```

```
<skos:ConceptScheme rdf:about="http://partage.vocnet.org/Actor">
  <rdf:type rdf:resource="http://www.w3.org/2004/02/skos/core#ConceptScheme"/>
</skos:ConceptScheme>
<skos:Concept rdf:about="http://partage.vocnet.org/part-kue00023474">
  <skos:inScheme rdf:resource="http://partage.vocnet.org/Actor"/>
  <skos:prefLabel xml:lang="en">Prutscher, Otto</skos:prefLabel>
  <skos:scopeNote xml:lang="en">Person / Activity Dates: 1895-1949 / Birth: 1880.04.07 /
1882.04.07 (Wien) / Death: 1949.02.15 (Wien)</skos:scopeNote>
</skos:Concept>
</xsl:variable>
</xsl:stylesheet>
```