PARTHENOS

Pooling Activities, Resources and Tools for Heritage E-research Networking, Optimization and Synergies

Report on services and tools (final)

Alessia Bardi, CNR

George Bruseker, FORTH

Matej Durco, OEAW

Klaus Illmayer, OEAW

Daan Broeder, DANS

Matteo Lorenzini, CNR

Stefan Resch, OEAW

Go Sugimoto, OEAW

Maria Theodoridou, FORTH

Massimiliano Assante, CNR

DATE 20th March 2019

PARTHENOS is a Horizon 2020 project funded by the European Commission. The views and opinions expressed in this publication are the sole responsibility of the author and do not necessarily reflect the views of the European Commission.

HORIZON 2020 - INFRADEV-4-2014/2015:

Grant Agreement No. 654119

PARTHENOS

Pooling Activities, Resources and Tools for Heritage E-research Networking, Optimization and Synergies

Report on services and tools (Final)

Deliverable Number	D6.4
Dissemination Level	PUBLIC
Delivery date	29 March 2019
Status	Final
Author(s)	Alessia Bardi, CNR George Bruseker, FORTH Matej Durco, OEAW Klaus Illmayer, OEAW Daan Broeder, DANS Matteo Lorenzini, CNR Stefan Resch, OEAW Go Sugimoto, OEAW Maria Theodoridou, FORTH
Contributors	Massimiliano Assante, CNR Sheena Bassett, PIN Athanasios Karasimos, AA Kostas Petrakis, FORTH

Project Acronym	PARTHENOS
Project Full title	Pooling Activities, Resources and Tools for Heritage E-research Networking, Optimization and Synergies
Grant Agreement nr.	654119

Deliverable/Document Information

Deliverable nr./title	D6.4 Report on services and tools (Final)
Document title	Report on services and tools (Final)
Author(s)	Alessia Bardi, George Bruseker, Matej Durco, Klaus Illmayer, Matteo Lorenzini, Stefan Resch, Go Sugimoto, Maria Theodoridou, Massimiliano Assante, Daan Broeder
Dissemination level/distribution	PUBLIC

Document History

Version/date	Changes/approval	Author/Approved by
V 0.1 2019-01-23	Updated sections about 3M Editor and X3ML-engine	George Bruseker
V 0.2 2019-02-03	Updated section on specialised tools	Daan Broeder
V 0.3 2019-02-07	Updated table 3	Daan Broeder
V 0.4 2019-02-19	Updated methodology section	Alessia Bardi
2019-02-20	Added description of RDF viewer	George Bruseker
2019-02-25	● Updated 6.3 Modes of integration ● New Section Findability and accessibility of services New Use Case Section	Massimiliano Assante
2019-02-25	Updated section on T6.2 and methodology	Alessia Bardi
15.03.2019	Final review	Sheena Bassett

Table of Contents

List of Figures

List of Tables

1. Executive Summary

The objectives of WP6 are:

- To set up a set of tools and services enabling cross-discipline interoperability;
- To share tools and services across different communities.

This document reports on the status of actions carried out towards achieving those two objectives in the context of three tasks:

Task 6.2 - Tools and services enabling interoperability

This task addresses the provision of interoperability tools/services to the PARTHENOS research communities and contributes to the first objective of WP6. The provision status of tools and services for interoperability is summarised in Table 1.

Task 6.3 - Sharing specialized tools

This task addresses the provision of tools/services for manipulating, presenting and publishing similar data within the disciplinary domains (or possibly outside) and contributes to the second objective of the WP. A number of specialised services has been selected for integration in the PARTHENOS infrastructure, as summarised in Table 3.

Task 6.5 - Resource discovery tools

This task contributes to the second objective of the WP by delivering resource discovery tooling for the PARTHENOS data space. An extensive evaluation of existing solutions supporting resource discovery has been carried out, based on which Metaphactory has been eventually selected as a framework to be customized to power the so-called “PARTHENOS Discovery” application.

This deliverable extends and finalises D6.2 “Report on services and tools (interim)” that was delivered in April 2017.

Outline of the report

The document is organised into five main sections (after this Section 1). Section 2describes the methodology adopted by T6.2 and T6.3 for the selection and the delivery of relevant tools and services to the PARTHENOS community. Section 3 describes the integration status of services and tools for interoperability into the PARTHENOS infrastructure (T6.2). Section 4 describes the work that has been done in T6.3 for the integration of community specific services (T6.3). Section 5 describes the progress of T6.5 for the development and integration of advanced resource discovery tools. Section 6 describes how the PARTHENOS VRE has been created and integrated into the Portal on the website to make the services and data more accessible and findable for end users, especially those who have not been involved in the project, who need a means of exploring what is available. Section 7 concludes the deliverable with the description of two use cases: Reference Resources Integration (RUBRICA) and the Language Resource Switchboard.

2. Methodology

Tasks 6.2 and 6.3 share a common methodology for the selection and assessment of tools and services to be integrated into the PARTHENOS infrastructure. The methodology is implemented together with WP2 (Community involvement and requirements) and WP5 (Interoperability and semantics) and it is composed of one preliminary phase of service selection (Figure 1), one phase of service integration and assessment (Figure 2) and one final phase for community service integration into a Virtual Research Environment openly available to all researchers in the disciplines covered by the PARTHENOS consortium.

2.1 Phase of service selection

In order to identify the services to be integrated into the PARTHENOS infrastructure, WP2 and WP6 performed investigations on the users' requirements and use case scenarios (WP2) and on the services offered by the research infrastructures in the consortium from which other research communities could benefit (WP6). The investigation carried out in WP2 resulted in report D2.1 "Report on User Requirements"[1], from which a set of technical functional and non-functional requirements for the design and implementation of the PARTHENOS infrastructure were distilled, together with a set of use case scenarios showing key examples of the paths that a researcher usually follows to achieve results in his/her field.

[1] D2.1 Report on User Requirements https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5a65b7331&appId=PPGMS (last visited 18/02/2019)

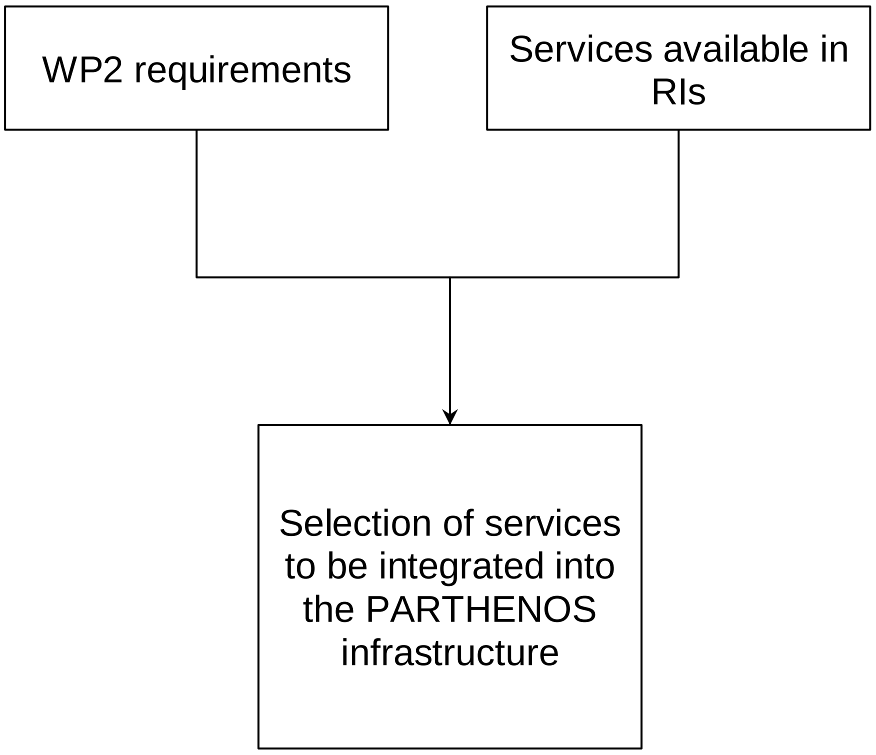

WP2 requirements
Services available in RIs
Selection of services to be integrated into the PARTHENOS infrastructure

Figure 1. Preliminary phase: selection of services

2.2 Phase of service integration and assessment

For each selected service, the methodology depicted in figure 2 has been followed. The service developers/maintainer and WP6 team agrees on one integration mode among those described in details in deliverable D2.4 “Assessment Interoperability Services and Tools” (Section 8.3) [8]. The service is then integrated into the PARTHENOS infrastructure and made available on the PARTHENOS Lab VRE2, a Virtual Research Environment that serves as a laboratory to integrate and develop cross community use cases identified by the PARTHENOS Project consortium.

Services available in the PARTHENOS Lab VRE have been evaluated in Spring 2018 by the Assessment Core taskforcE (ACE Team), a small task force composed of people from WP2, 5 and 6 whose aim was to coordinate the service assessment activities which were reported in deliverable D2.4 “Report on the Assessment of Interoperability, Services and Tools” [8]. Technical issues and gaps of services and tools described in D2.4 were reported via the PARTHENOS issue tracker to the service/tool developers and WP6 members.

[2] PARTHENOS LAB Virtual Research Environment: https://parthenos.d4science.org/group/parthenos_lab (last visited 18/02/2019)

The new version of the tool/service was then deployed on the PARTHENOS infrastructure.

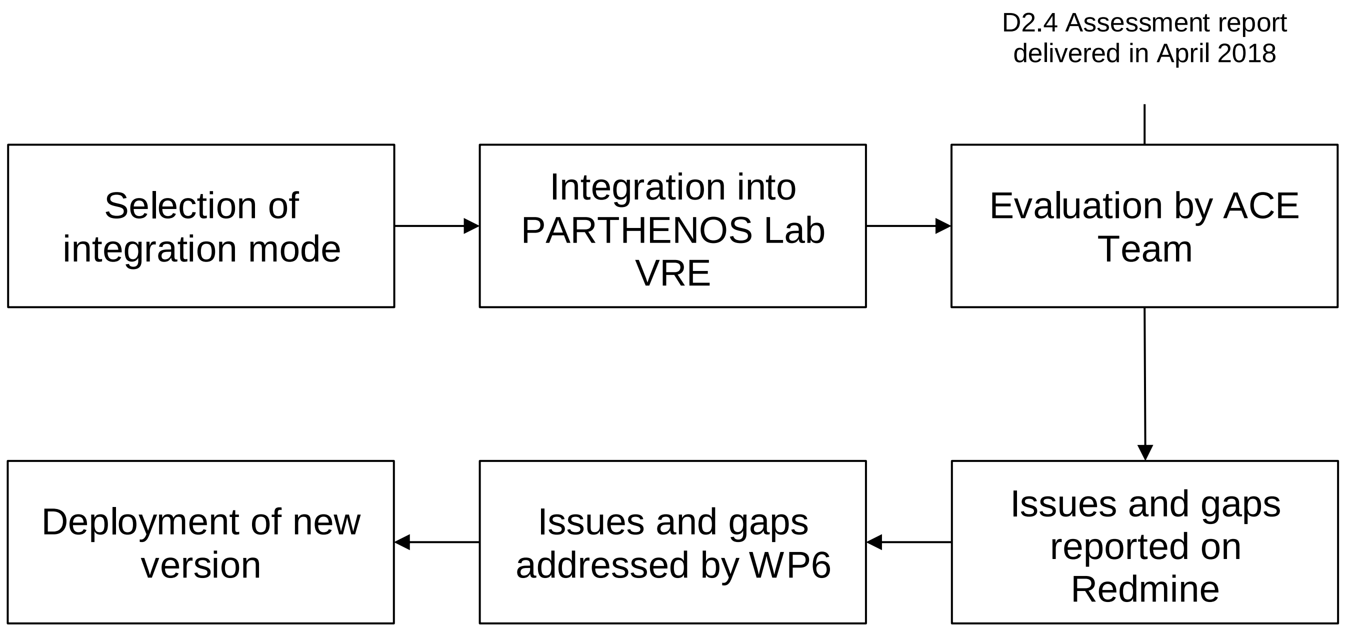

D2.4 Assessment report delivered in April 2018
Selection of integration mode
Integration into PARTHENOS Lab VRE
Evaluation by ACE Team
Issues and gaps reported on Redmine
Issues and gaps addressed by WP6
Deployment of new version

Figure 2. Integration and assessment of services

2.3 Phase of community service promotion

In the case of community specialised services (T6.3), the process depicted in figure 3 is also applied. The goal of this third phase is to identify the services and tools that can be made publicly accessible to researchers via a dedicated open VRE named “The PARTHENOS VRE”. The decision of making a community service available from the PARTHENOS VRE is guided by two criteria:

1. The value added by the service/tool to the offer of the PARTHENOS VRE:
    a. Usefulness of the tool/service for the implementation of one or more use cases depicted in D2.1;
    b. Overlapping functionality with other services/tools.
2. Maturity of the service/tool
    a. Technology Readiness Level (TRL) of the service/tool;
    b. User-friendliness of the tool/service, including the availability of clear documentation for non IT-experts end-users.

The first criterion is applied to identify the useful services/tools and to select the set of services that maximise the offer of the PARTHENOS VRE and minimise the duplication of functionality.

The second criterion is applied to verify if the service/tool is ready to be promoted to the PARTHENOS VRE for public use or if it should be consolidated or extended. It is also possible that some services and tools may not reach a sufficient level of maturity by the end of the project to be included in the PARTHENOS VRE. In such a case, the service/tool will still be available from the PARTHENOS Lab VRE, which will also be maintained after the end of the project in order to support additional testing and integration activities of the developers and other IT-experts researchers willing to contribute to the evolution of the service/tool.

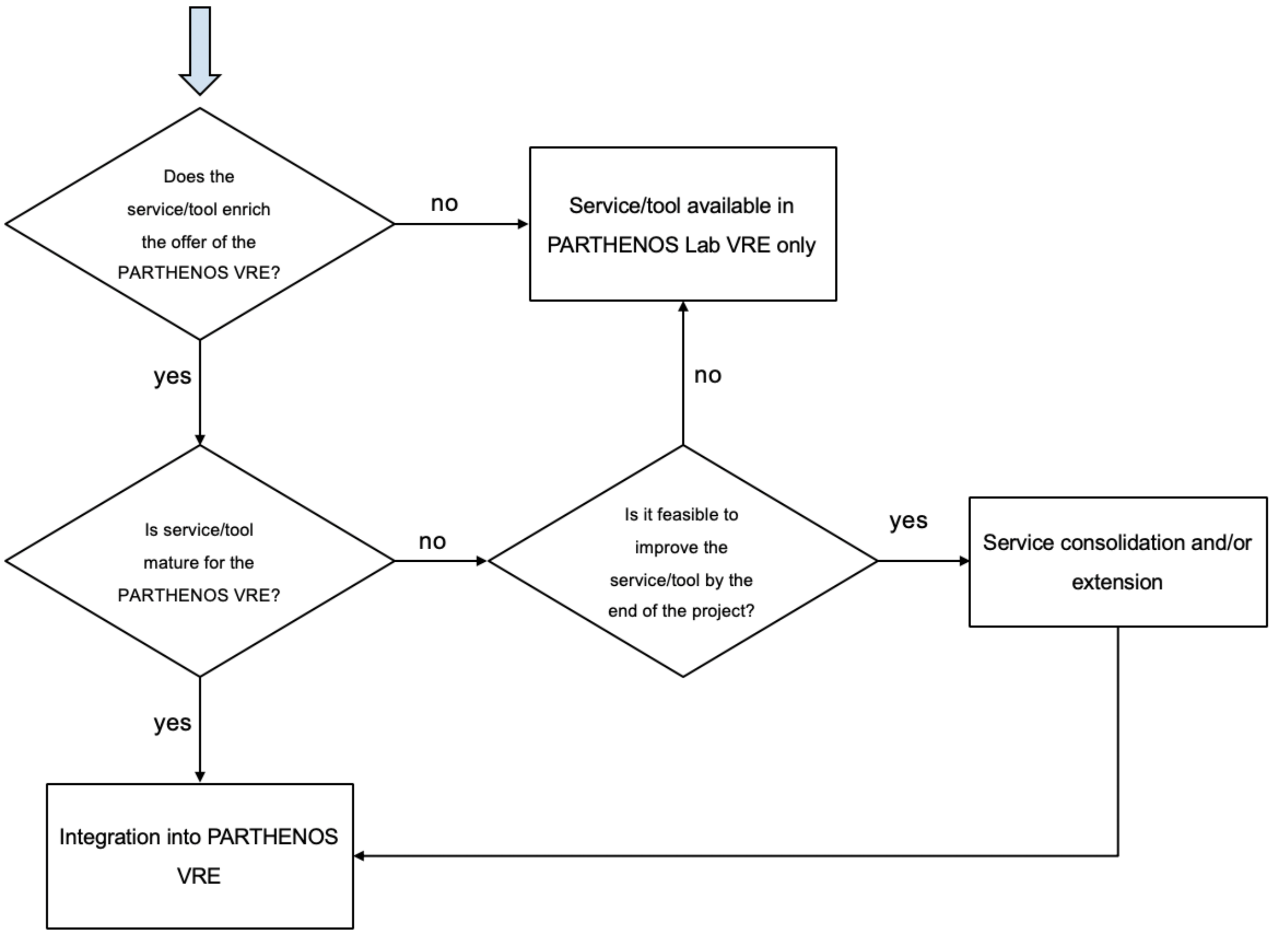

Does the service/tool enrich the offer of the PARTHENOS VRE?
no
Service/tool available in PARTHENOS Lab VRE only
yes
no
Is service/tool mature for the PARTHENOS VRE?
no
Is it feasible to improve the service/tool by the end of the project?
yes
Service consolidation and/or extension
yes
Integration into PARTHENOS VRE

Figure 3. Promotion of community services/tools into the PARTHENOS Showcase VRE

3. Tools and services enabling interoperability (Task 6.2)

Task 6.2 addresses the provision of interoperability tools/services to the PARTHENOS research communities. Table 1 summarizes the services and tools that have been already identified relevant for the PARTHENOS community and their integration stage into the PARTHENOS infrastructure.

Table 1 T6.2: tools and services enabling interoperability

Service/Tool	Description	Responsible partner	Addressed requirement(s)	Integration stage
X3ML toolkit	A set of open source components that assist the definition of mappings from XML to PARTHENOS Entities Model RDF for information integration. It includes the 3M Editor.	FORTH	Metadata experts should be able to define the mappings to the PARTHENOS Entities model.	Deployed
RDF Viewer	This tool allows for the intuitive and compact viewing of RDF files in a human readable form. It is used in this project for quality testing of semantic mappings.	FORTH	Metadata quality control	Deployed: integrated with the 3M Editor
D-NET Software Toolkit	Enabling framework for the realization and operation of aggregative metadata infrastructure	CNR-ISTI	Set-up of a PARTHENOS aggregator to enable cross-RI search and browse	Deployed

X3ML engine	Transformation engine capable of applying mappings defined via the 3M Editor.	FORTH	Automatic application of mappings on the PARTHENOS side	Deployed: integrated with D-Net
Metadata Cleaner	Service for the harmonization of values according to controlled vocabularies	CNR-ISTI	Adoption of common controlled vocabularies	Deployed: integrated with D-Net
Metadata Inspector	GUI for the visualization of transformed metadata records and detection of "uncleaned" records	CNR-ISTI	Metadata quality control	Deployed: integrated with D-Net

This set of tools are used in an integrated environment for the realisation of the PARTHENOS Content Cloud framework (see Figure 4 and deliverable D6.1 [1]).

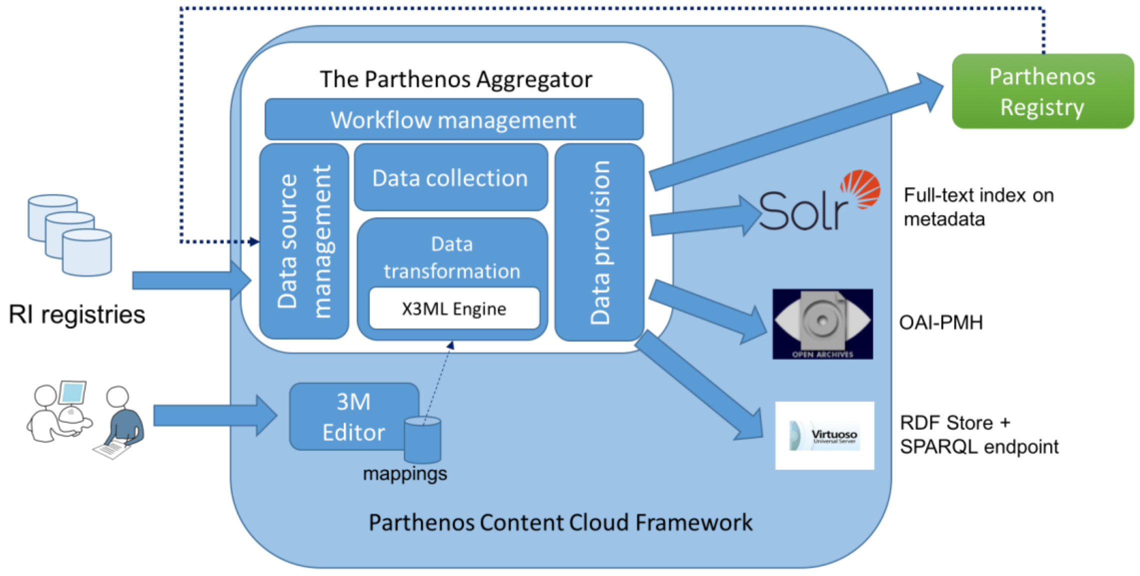


Figure 4. High Level Architecture of the PARTHENOS Content Cloud Framework

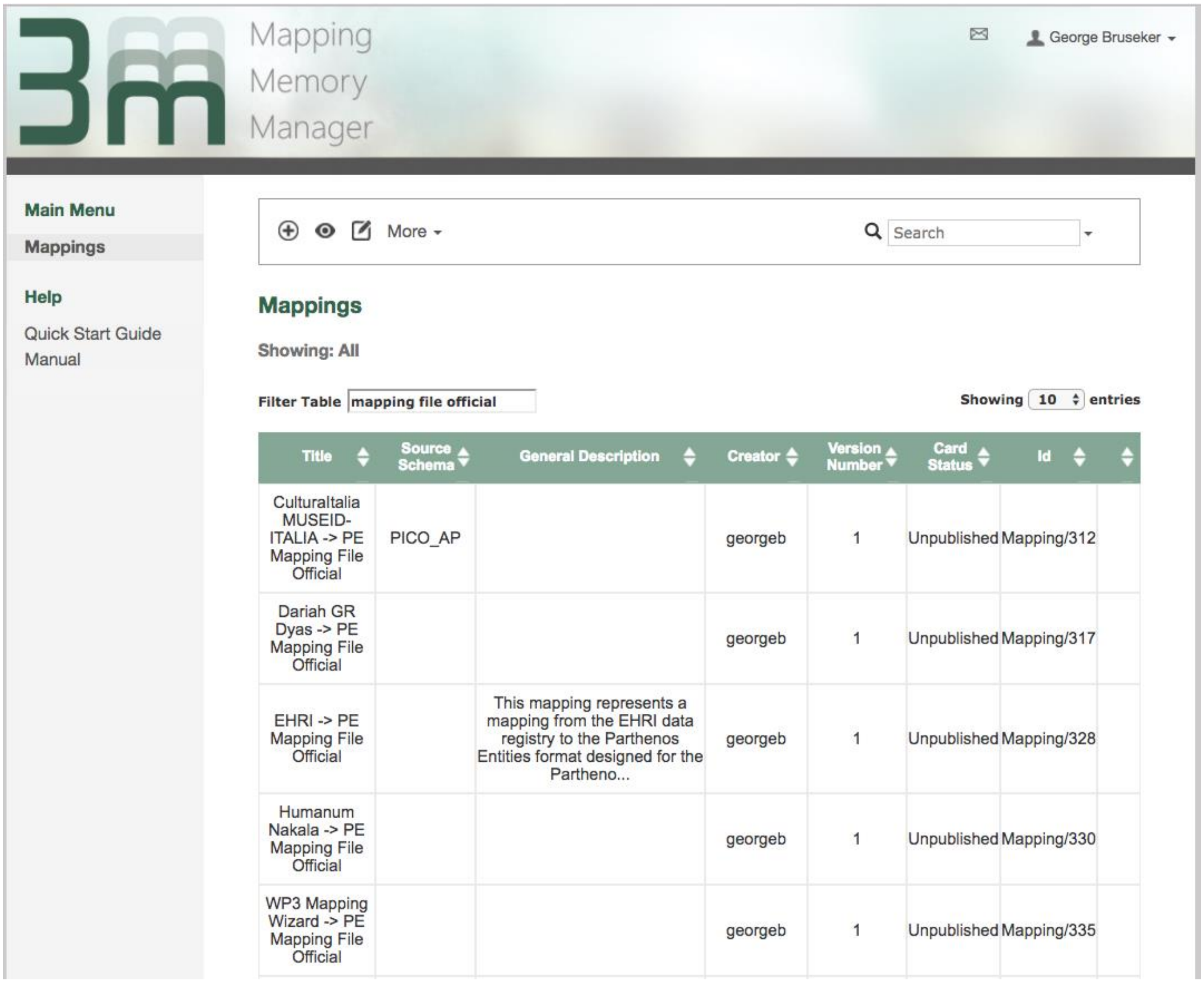
3.1 X3ML Toolkit

Target requirement/use case

The PARTHENOS consortium expressed the requirement for setting up an aggregative infrastructure to collect metadata from the research infrastructures participating in the consortium. In order to build a homogenous information space, where metadata records collected from different providers are harmonised according to the PARTHENOS Entities model (CIDOC PE), providers must define metadata mappings from their registries to CIDOC PE. The X3ML toolkit supports the definition of mapping with a user-friendly GUI and has been used with success in several projects (e.g. ARIADNE, ResearchSpace, VRE4EIC).

Description

The X3ML Toolkit [6] consists of a set of software components that assist the data provisioning process for information integration. It follows the SYNERGY Reference Model, a rich and comprehensive Reference Model for a better practice of data provisioning and aggregation processes, adapted from ongoing work of the CIDOC CRM SIG. The key components of the toolkit are: (a) 3M, the Mapping Memory Manager (see Figure 5), (b) the 3M Editor, and (c) the X3ML Engine.



Mappings

Showing: All

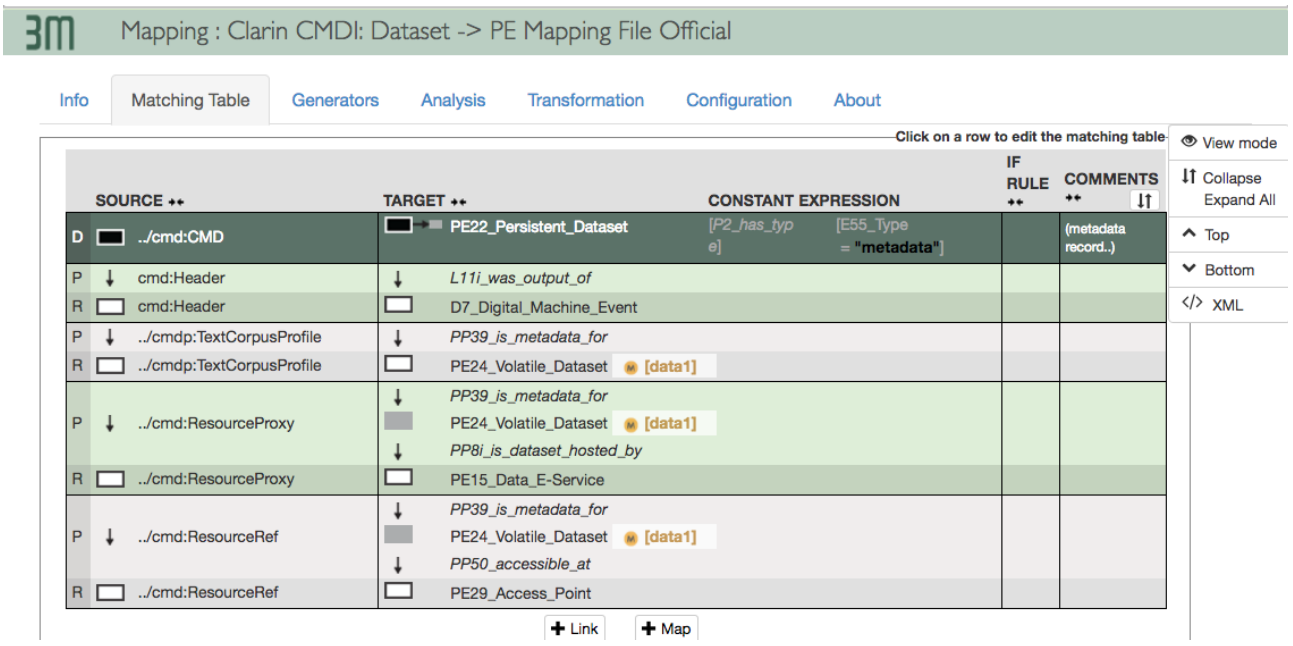
Filter Table: mapping file official

Showing 10 entries

Title	Source Schema	General Description	Creator	Version Number	Card Status	Id	
CulturaItalia MUSEID-ITALIA -> PE Mapping File Official	PICO_AP		georgeb	1	Unpublished	Mapping/312	
Dariah GR Dyas -> PE Mapping File Official			georgeb	1	Unpublished	Mapping/317	
EHRI -> PE Mapping File Official		This mapping represents a mapping from the EHRI data registry to the Parthenos Entities format designed for the Partheno...	georgeb	1	Unpublished	Mapping/328	
Humanum Nakala -> PE Mapping File Official			georgeb	1	Unpublished	Mapping/330	
WP3 Mapping Wizard -> PE Mapping File Official			georgeb	1	Unpublished	Mapping/335	

Figure 5. 3M Mapping Memory Manager interface

The X3ML Toolkit is a web application suite containing several software sub-components that exploit several external services. Its main functionality is to assist users during the mapping definition process, using a human-friendly user interface and a set of sub-components that either suggest or validate the user input. Figure 6, Figure 7 and Figure 8 show the GUI of the 3M Editor for the definition and test of mappings.

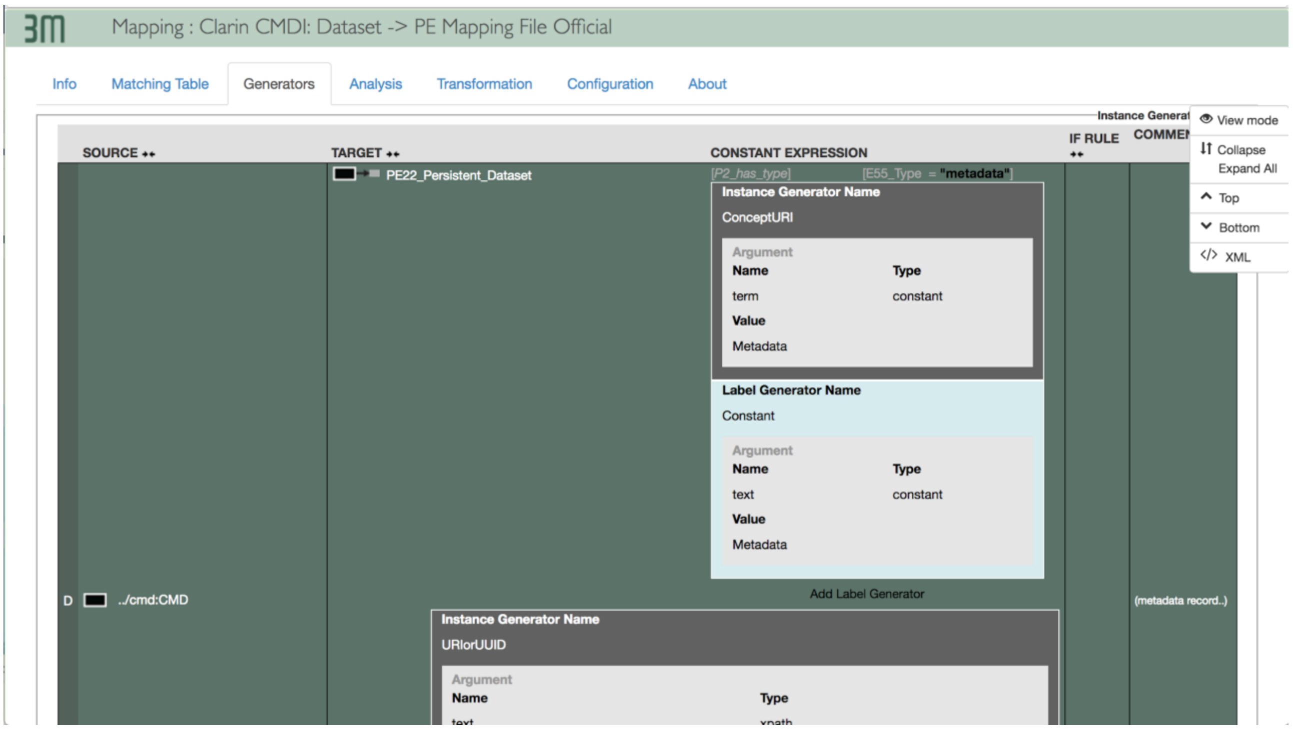

3M Mapping : Clarin CMDI: Dataset -> PE Mapping File Official

Info | Matching Table | Generators | Analysis | Transformation | Configuration | About

Click on a row to edit the matching table

	SOURCE	TARGET	CONSTANT EXPRESSION	IF RULE	COMMENTS
D	../cmd:CMD	PE22_Persistent_Dataset	[P2_has_type] [E55_Type = "metadata"]		(metadata record..)
P	cmd:Header	*L11i_was_output_of*			
R	cmd:Header	D7_Digital_Machine_Event			
P	../cmdp:TextCorpusProfile	*PP39_is_metadata_for*			
R	../cmdp:TextCorpusProfile	PE24_Volatile_Dataset [data1]			
P	../cmd:ResourceProxy	*PP39_is_metadata_for* PE24_Volatile_Dataset [data1] *PP8i_is_dataset_hosted_by*			
R	../cmd:ResourceProxy	PE15_Data_E-Service			
P	../cmd:ResourceRef	*PP39_is_metadata_for* PE24_Volatile_Dataset [data1] *PP50_accessible_at*			
R	../cmd:ResourceRef	PE29_Access_Point			

Figure 6. 3M Editor declarative mapping interface


3M Mapping : Clarin CMDI: Dataset -> PE Mapping File Official

Info | Matching Table | Generators | Analysis | Transformation | Configuration | About

SOURCE | TARGET | CONSTANT EXPRESSION | IF RULE | COMMENTS

D ../cmd:CMD | PE22_Persistent_Dataset | [P2_has_type] [E55_Type = "metadata"] | (metadata record..)

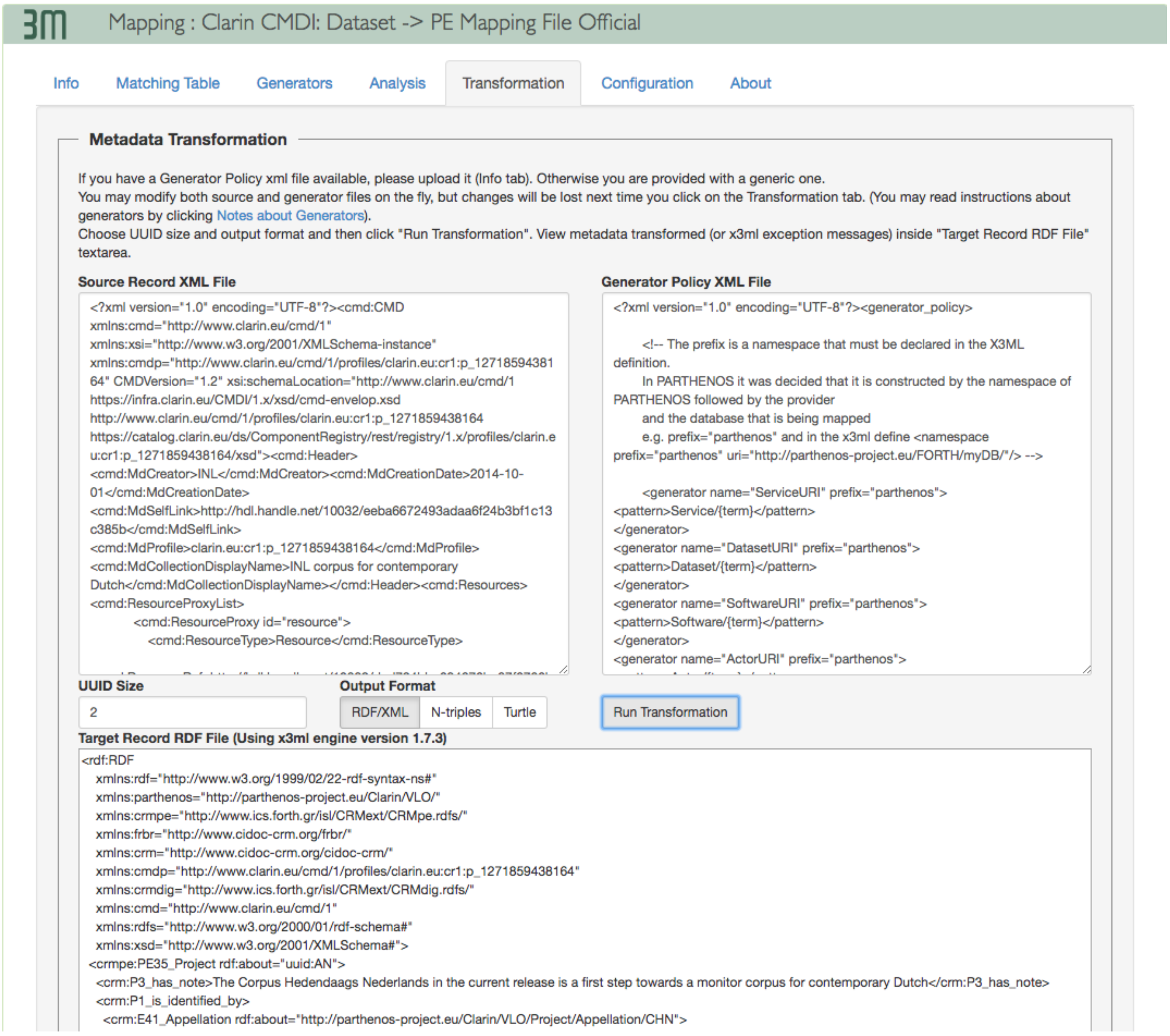
Instance Generator Name
ConceptURI
Argument
Name term **Type** constant
Value Metadata

Label Generator Name
Constant
Argument
Name text **Type** constant
Value Metadata

Add Label Generator

Instance Generator Name
URIorUUID
Argument
Name text **Type** xpath

Figure 7. 3M Editor instance generator interface



Metadata Transformation

If you have a Generator Policy xml file available, please upload it (Info tab). Otherwise you are provided with a generic one.
You may modify both source and generator files on the fly, but changes will be lost next time you click on the Transformation tab. (You may read instructions about generators by clicking Notes about Generators).
Choose UUID size and output format and then click "Run Transformation". View metadata transformed (or x3ml exception messages) inside "Target Record RDF File" textarea.

Source Record XML File

```
<?xml version="1.0" encoding="UTF-8"?><cmd:CMD
xmlns:cmd="http://www.clarin.eu/cmd/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:cmdp="http://www.clarin.eu/cmd/1/profiles/clarin.eu:cr1:p_12718594381
64" CMDVersion="1.2" xsi:schemaLocation="http://www.clarin.eu/cmd/1
https://infra.clarin.eu/CMDI/1.x/xsd/cmd-envelop.xsd
http://www.clarin.eu/cmd/1/profiles/clarin.eu:cr1:p_1271859438164
https://catalog.clarin.eu/ds/ComponentRegistry/rest/registry/1.x/profiles/clarin.e
u:cr1:p_1271859438164/xsd"><cmd:Header>
<cmd:MdCreator>INL</cmd:MdCreator><cmd:MdCreationDate>2014-10-
01</cmd:MdCreationDate>
<cmd:MdSelfLink>http://hdl.handle.net/10032/eeba6672493adaa6f24b3bf1c13
c385b</cmd:MdSelfLink>
<cmd:MdProfile>clarin.eu:cr1:p_1271859438164</cmd:MdProfile>
<cmd:MdCollectionDisplayName>INL corpus for contemporary
Dutch</cmd:MdCollectionDisplayName></cmd:Header><cmd:Resources>
<cmd:ResourceProxyList>
        <cmd:ResourceProxy id="resource">
            <cmd:ResourceType>Resource</cmd:ResourceType>
```

Generator Policy XML File

```
<?xml version="1.0" encoding="UTF-8"?><generator_policy>

      <!-- The prefix is a namespace that must be declared in the X3ML
definition.
      In PARTHENOS it was decided that it is constructed by the namespace of
PARTHENOS followed by the provider
      and the database that is being mapped
      e.g. prefix="parthenos" and in the x3ml define <namespace
prefix="parthenos" uri="http://parthenos-project.eu/FORTH/myDB/"/> -->

      <generator name="ServiceURI" prefix="parthenos">
<pattern>Service/{term}</pattern>
</generator>
<generator name="DatasetURI" prefix="parthenos">
<pattern>Dataset/{term}</pattern>
</generator>
<generator name="SoftwareURI" prefix="parthenos">
<pattern>Software/{term}</pattern>
</generator>
<generator name="ActorURI" prefix="parthenos">
```

UUID Size: 2

Output Format: [✓ RDF/XML] N-triples Turtle

Run Transformation

Target Record RDF File (Using x3ml engine version 1.7.3)

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:parthenos="http://parthenos-project.eu/Clarin/VLO/"
  xmlns:crmpe="http://www.ics.forth.gr/isl/CRMext/CRMpe.rdfs/"
  xmlns:frbr="http://www.cidoc-crm.org/frbr/"
  xmlns:crm="http://www.cidoc-crm.org/cidoc-crm/"
  xmlns:cmdp="http://www.clarin.eu/cmd/1/profiles/clarin.eu:cr1:p_1271859438164"
  xmlns:crmdig="http://www.ics.forth.gr/isl/CRMext/CRMdig.rdfs/"
  xmlns:cmd="http://www.clarin.eu/cmd/1"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
 <crmpe:PE35_Project rdf:about="uuid:AN">
  <crm:P3_has_note>The Corpus Hedendaags Nederlands in the current release is a first step towards a monitor corpus for contemporary Dutch</crm:P3_has_note>
  <crm:P1_is_identified_by>
   <crm:E41_Appellation rdf:about="http://parthenos-project.eu/Clarin/VLO/Project/Appellation/CHN">
```

Figure 8. 3M Editor transformation test facility

All the components of the X3ML Toolkit have been developed as open source components in the context of the projects CultureBrokers, ARIADNE[3], KRIPIS – POLITEIA[4], and VRE4EIC[5]. More specifically, the X3ML Toolkit components have been released under the European Union Public Licence whereas the X3ML engine has been released under the Apache 2 licence. FORTH (Foundation for Research & Technology – Hellas) is the main developer of the X3ML Toolkit and was supported by DelvingBV (www.delving.eu) in the initial implementation of the X3ML engine.

[3] ARIADNE: http://www.ariadne-infrastructure.eu

[4] KRIPIS – POLITEIA: http://politeia.ims.forth.gr

[5] VRE4EIC: https://www.vre4eic.eu

The X3ML Toolkit supports the data aggregation process by providing mechanisms for data transformation and URI generation. Mappings are specified using the X3ML mapping definition language, an XML-based, declarative, and human readable language that supports the cognitive process of a mapping in such a way that the generated maps can be collaboratively created and discussed by domain experts with little to no IT knowledge.
Unlike XSLT, that is comprehensible only by IT experts with training, the X3ML mapping definition language is designed from the ground up to be understood equally by IT experts as by non-technical domain experts whose data is the subject of transformation. This enables a domain expert to verify the semantics of a mapping by reading and validating the schema matching. This model carefully distinguishes between mapping activities carried out by the domain experts, who know and provide the data, from the activities of the IT technicians, who actually implement data translation and integration solutions.

Usually, schema matching is used to describe the process of identifying that two different concepts are semantically related. This allows the definition of the appropriate mappings to be used as rules for the transformation process. However, a common problem is that the IT experts do not fully understand the semantics of the schema matching - it’s not their data - and the domain experts do not understand how to use the technical solutions for creating mappings. For this reason, the X3ML Toolkit relies on two distinct components which separate task of schema matching from the URI generation processes. The schema matching can be fully performed by the domain expert and the URI generation by the IT expert, therefore solving the bottleneck that requires the IT expert to fully understand the mapping. Furthermore, this approach keeps the schema mappings between different systems harmonized since their definitions do not change, in contrast to the URIs that may change between different institutions and are independent of the semantics. Moreover, this approach completely separates the definition of the schema matching from the actual execution. This is important because different processes might have different life cycles; in particular, the schema matching definition has a different life cycle compared to the URI generation process. The former is subject to changes more rarely compared to the latter. The development of the X3ML toolkit attempts to support the need to involve and record the knowledge of the data producers and ensure the best possible semantic and contextual mapping of cultural data.

The X3ML toolkit is a “community web application”. While it supports security for editing files (a user can edit a mapping only if the creator of the mapping authorized him/her), it allows the whole community to view a mapping file (with the 3M Mapping Memory Manager, Figure 5). This means that one can look at other users mapping definitions to learn techniques and see approaches to different types of information. It is also possible to share editing rights with others. Although only one person can edit at a time, this provides some capability to share the mapping process with someone else online – to suggest and consult, or to make corrections.

During the course of the project, the RDF Viewer, a separate project for navigating RDF files in a straightforward and intuitive fashion was added into the 3M editor environment. This was done in order to support feedback to data mappers with regards to the quality of their mappings. This feature was highly well received and helped improve the quality and understanding of mappings. The code and functionality of this tool is described here (https://github.com/cpetrakis/RDFVisualizer-webapp). For a full description of the tool, see Section 2.2.

Integration into the PARTHENOS infrastructure

The X3ML toolkit was consolidated in order to be installable on Linux machines (the initial version worked only on Windows). The X3ML toolkit is deployed in the PARTHENOS infrastructure and integrated in the PARTHENOS Project Virtual Research Environment.

Assessment

The X3ML toolkit has been made available to all PARTHENOS members.

The RIs participating in the PARTHENOS consortium were invited to a training workshop in Rome, October 2016, where FORTH members presented the PARTHENOS Entities model, the mapping techniques, the usage of the X3ML toolkit and provided the participants with guidelines for setting up and making a complete mapping. A set of mapping exercises have been implemented and made available to all the partners.

Following the training, thirteen partners including RIs and cross WP collaborators started using the toolkit in order to define the mapping of their source schema to the PARTHENOS Entities Model. FORTH continuously supported the partners during their mapping efforts and collected their feedback in order to resolve bugs and enhance the usability of the graphical user interface.

3.2 RDF Viewer

Target requirement/use case

Data mappers required a visual tool to check the quality of their mappings.

Description

RDF Visualizer is a generic browsing mechanism that gives the user a flexible, highly configurable, detailed overview of an RDF dataset / database, designed and developed to overcome the drawbacks of the existing RDF data visualization methods/tools. RDFV is currently used by the 3M editor of the X3ML suite of tools and has been tested with large datasets.

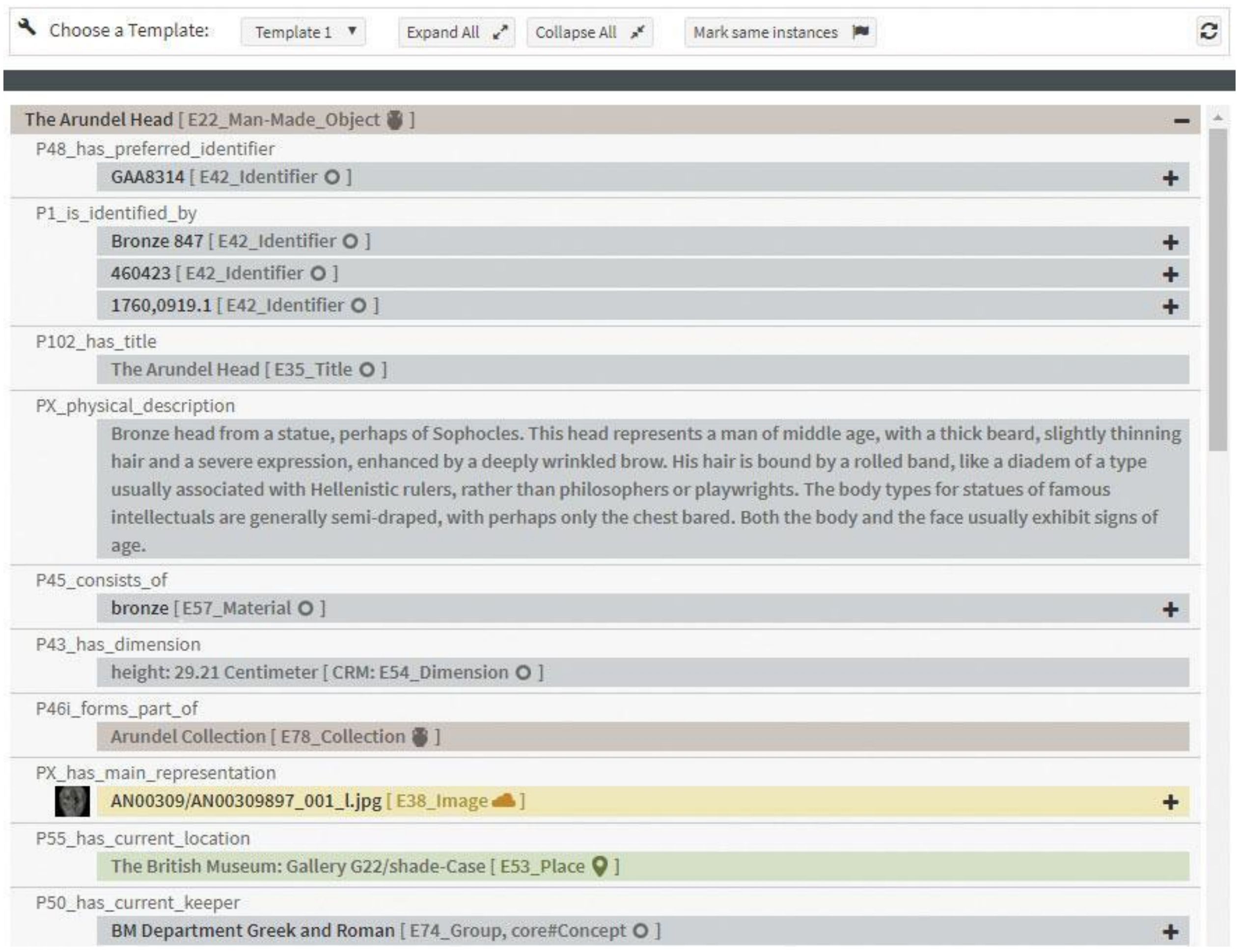
Although a great deal of emphasis has been placed on the validation of the produced RDF structure and format, the efficient visualization of the constructed database contents that enables semantic validation by domain experts has largely been ignored and is achieved either by manual inspection of multiple files, by formulation and execution of complex SPARQL queries, or by custom user interfaces that work only with a particular RDF schema without intervention by a programmer.

The basic principles that an efficient and user-friendly RDF data visualization tool should be able to live up to are:

1. The ability to display data of any schema and RDF format.
2. The ability to display all nodes of any class/instance.
3. The application of configuration rules to improve the layout or presentation for known classes and properties (e.g., hide URIs that are meaningless for the user).
4. The display of a high density of information in one screen (which is not possible in solutions based on "object templates").

The existing approaches fail to fulfil the complete set of principles enumerated above. Specifically, approaches that display all the nodes of any class/schema and support any schema and format (principles 1 and 2) fail to display a high density of information in one screen and those that succeed with regards to the latter fail in relation to the rest of the principles.

In order to meet the requirements laid out by the complete set of principles, our team designed and implemented the RDFV tool. RDFV presents RDF data as an indented list to handle the density and depth of information (principle 4) starting from a specified RDF resource (URI). In order to achieve this, all incoming links are inverted to display all the nodes of every class/instance (principle 2) in a schema agnostic way (principle 1). Users are able to configure the display of schema-dependent information according to their preferences (principle 4). By editing an xml file, users are also able to define priority-based rule chains that are used to define schema-dependent style and order of properties that are inherited to subclasses and sub-properties (principle 4). For anything not covered by a rule, default options are applied based on our experience and best practices. The user interface of the tool has been designed in cooperation with potential users, with a focus on usability and readability.



The Arundel Head [E22_Man-Made_Object]

P48_has_preferred_identifier
GAA8314 [E42_Identifier]

P1_is_identified_by
Bronze 847 [E42_Identifier]
460423 [E42_Identifier]
1760,0919.1 [E42_Identifier]

P102_has_title
The Arundel Head [E35_Title]

PX_physical_description
Bronze head from a statue, perhaps of Sophocles. This head represents a man of middle age, with a thick beard, slightly thinning hair and a severe expression, enhanced by a deeply wrinkled brow. His hair is bound by a rolled band, like a diadem of a type usually associated with Hellenistic rulers, rather than philosophers or playwrights. The body types for statues of famous intellectuals are generally semi-draped, with perhaps only the chest bared. Both the body and the face usually exhibit signs of age.

P45_consists_of
bronze [E57_Material]

P43_has_dimension
height: 29.21 Centimeter [CRM: E54_Dimension]

P46i_forms_part_of
Arundel Collection [E78_Collection]

PX_has_main_representation
AN00309/AN00309897_001_l.jpg [E38_Image]

P55_has_current_location
The British Museum: Gallery G22/shade-Case [E53_Place]

P50_has_current_keeper
BM Department Greek and Roman [E74_Group, core#Concept]

Figure 9. RDFV user interface

Moreover, rich functions are provided to the user to control the display of data items, such as identifying same instances, expanding collapsing big texts or big sets of results, showing object's URI in hover, displaying images and image galleries, removing prefixes, selecting non-displayed URIs, retrieving the path of the sub-graph for a specific node etc. RDFV supports browsing of content in triple stores (Virtuoso and BlazeGraph) and in local .ttl files. It has been integrated in the 3M interface of the X3ML suite of tools for data mapping and transformation. Added into the 3M interface it adds an important validation tool for data mapped and transformed by domain experts who wish to check and correct the resulting outputs, enabling an iterative and collaborative evaluation of the resultant RDF.

The tool has been developed as an open source project, consists of a java API and a maven java web application, is available on GitHub, and has been released under Apache 2 licence.

Integration into the PARTHENOS infrastructure

The tool has been integrated into the 3M Editor available in the PARTHENOS Project VRE.

Assessment

The RDF Viewer has been integrated after the final assessment performed by the ACE team in Spring 2018. Data mappers provided feedback about the tool to FORTH, who is maintaining the tool, and reports that the tool was highly well received and helped improve the quality and understanding of mappings.

3.3 D-NET Software Toolkit

Target requirement/use case

The PARTHENOS consortium expressed the requirement for setting up an aggregative infrastructure to collect metadata from the research infrastructures in the consortium, transform them according to the PARTHENOS data model to form a homogenous information space to expose to third-party consumers through a number of APIs. D-NET offers out-of-the data management services and tools for assembly into workflows to facilitate the construction of domain-specific aggregative infrastructures like the one requested by the PARTHENOS consortium.

D-NET features several built-in plugins for the collection of metadata records (in XML, CSV, TSV and similar) via different exchange protocols (e.g. OAI-PMH, SFTP, FTP(S), HTTP(S), local file system). Additional plug-ins can be easily implemented and integrated if needed.

From the data processing point of view, D-NET features a Transformation Service capable of transforming metadata records from one format to another by applying mappings of different kinds (XSLTs, Groovy scripts, Java code, and D-NET transformation rules). For the publishing of the generated uniform information space, D-NET can be configured to export metadata records via OAI-PMH and to index metadata records into a Solr index.
For the full list of features of D-NET please refer to [2].

The D-NET Software toolkit has been used with success in several EC projects for the provision of content to the European Cultural Heritage aggregator Europeana (e.g. EFG, EAGLE, HOPE), initiatives for Open Access (DRIVER, OpenAIRE) and national and international aggregators like CeON (Poland), La Referencia (South America), Recolecta (Spain), SNRD (Sistema Nacional de Repositorios Digitales, Argentina).

Description

The D-NET Software Toolkit 6 (D-NET for brevity) is a service-oriented framework specifically designed to support developers at constructing custom aggregative infrastructures in a cost-effective way [2]. D- NET offers data management services capable of providing access to different kinds of external data sources, storing and processing information objects compliant to any data models, converting them into common formats, and exposing information objects to third-party applications through several standard access APIs. D-NET services are obtained by encapsulating advanced and state-of-the-art open source products for data storage, indexing, and processing – such as PostgreSQL, MongoDB, Apache Solr, and Apache HBase – in order to serve broad application needs. Most importantly, D-NET offers infrastructure enabling services that facilitate the construction of domain-specific aggregative infrastructures by selecting and configuring the needed services and easily combining them to form autonomic data processing workflows. The combination of out-of-the-box data management services and tools for assembling them into workflows makes the toolkit an appealing starting platform for developers dealing with the realization of aggregative infrastructures.7

[6] The D-NET Software Toolkit: http://www.d-net.research-infrastructures.eu/

[7] For a more detailed description of D-NET, its features, and its role in the PARTHENOS infrastructure please refer to D6.1, Section 2.5 [1].

CNR-ISTI (Italy) is the main developer of D-NET, supported by teams from University of Athens (Greece), Athena Research Centre (Greece), and University of Bielefeld (Germany). D-NET is open source (Apache licence), developed in Java, based on Service-Oriented Architecture paradigms. Its first software release was designed and developed within the DRIVER and DRIVER-II EC projects (2006-2008). The motive driving its development was that of constructing the European repository infrastructure for Open Access repositories. The infrastructure had to harvest (tens of) millions of Dublin Core metadata records from hundreds of OAI-PMH repository data sources, harmonizing the structure and values of such records to form a uniform information space.

Integration into the PARTHENOS infrastructure

The PARTHENOS Aggregator has been deployed in the PARTHENOS infrastructure and it is available for the aggregation managers at:
https://beta-parthenos.d4science.org/aggregator.

Assessment

The instance of D-NET for PARTHENOS was deployed on a beta server where developers and infrastructure administrators from CNR-ISTI and T6.2 performed technical and usability tests on the data processing workflow and the new plugins devised in the assessment phase. It is worth noting that the D-NET instance is not supposed to be used directly by end-users, but only by the administrators of the PARTHENOS infrastructure.

During the technical assessment phase D-NET was consolidated and upgraded to Java 8. As described above, D-NET is a framework for the construction of aggregative data infrastructure. As such, part of the assessment phase focused on the selection of available data management services and the design of the data processing workflow to be set up for the PARTHENOS aggregator (which is a D-NET instance). The data processing workflow devised for PARTHENOS is shown in Figure 10. The workflow comprises two automatic sub-workflows: the aggregation workflow automatically collects input metadata records, transforms them according to a defined X3ML mapping, and makes the resulting RDF records available to metadata experts for inspection. The publishing workflow is executed in order to officially publish the records. The publishing comprises: (1) the indexing of metadata on a Solr index - the Solr schema is defined in collaboration with T6.5; (2) the availability of an OAI-PMH export of the metadata (in CIDOC-CRM and oai_dc); (3) the update of the RDF store; and (4) the update of the PARTHENOS registry.

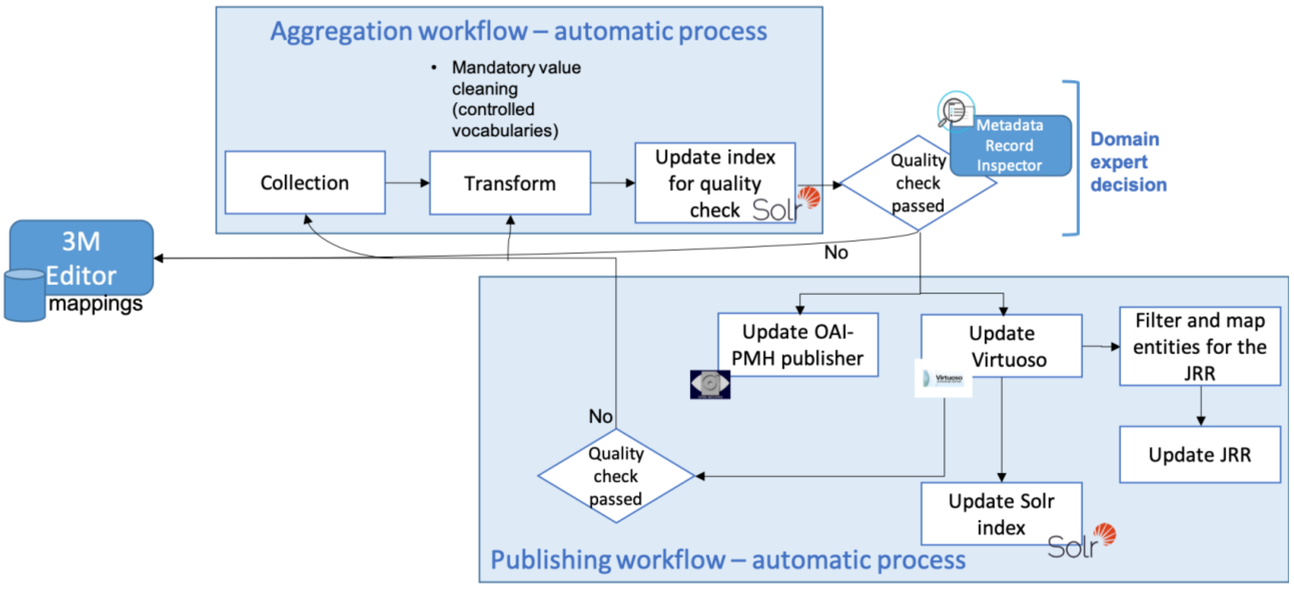

Aggregation workflow – automatic process
Mandatory value cleaning (controlled vocabularies)
Collection
Transform
Update index for quality check
Solr
Quality check passed
Metadata Record Inspector
Domain expert decision
3M Editor
mappings
No
Update OAI-PMH publisher
Update Virtuoso
Filter and map entities for the JRR
Update JRR
No
Quality check passed
Update Solr index
Solr
Publishing workflow – automatic process

Figure 10. The workflow devised for the PARTHENOS aggregative infrastructure

An analysis of the APIs of the research infrastructures has been carried out to ensure that D-NET could collect metadata records from all the available sources. The analysis revealed that D-NET natively covers the majority of API protocols used by research infrastructures. A summary of the analysis is available in Table 2.

Table 2. Analysis of the APIs of PARTHENOS research infrastructure with respect to the collection plugins natively available in D-NET

Research Infrastructure	Registry	Endpoint	Type	D-Net collector plugin	Action
ARIADNE	Ariadne Registry	SFTP server	SFTP with public key authentication	SFTP plugin	Plugin extended to support public key authentication.
CLARIN	Virtual Language Observatory	https://vlo.clarin.eu/resultsets/	XML dumps via HTTPS	HTTP plugin	
CENDARI	CENDARI Registry	Not available	XML dumps	Filesystem plugin	
Cultura Italia	CulturaItalia Catalogue	http://www.culturaitalia.it/oaiProviderCI/OAIHandler	OAI-PMH	OAI-PMH plugin	
DARIAH-DE	DARIAH-DE Portal	https://colreg.de.dariah.eu/colreg-ui/api/collections/	HTTP API	Dedicated collector plugin under development.	
DARIAH GR/ΔΥΑΣ	ΔΥΑΣ Organizations and Collections Registry	http://registries.dyas-net.gr/en/developer	OAI-PMH	OAI-PMH plugin	

EHRI	EHRI ICA Archives	https://portal.ehri-project.eu/units/<ID>/export	HTTP API	Dedicated collector plugin developed.	
HUMA-NUM	Isidore Collection Level	https://data.huma-num.fr/parthenos/isidore/	HTTP API	HTTP plugin	
HUMA-NUM	Isidore Item Levels (7 APIs)	https://api.rechercheisidore.fr/resource/search	HTTP API	Dedicated collector plugin developed.	
HUMA-NUM	Nakala Collection Level	http://data.huma-num.fr/parthenos/nakala/	HTTP API	HTTP plugin	
HUMA-NUM	Nakala Item Level (4 APIs)	https://www.nakala.fr/oai/11280/<collID>	OAI-PMH	OAI-PMH plugin	
	LRE MAP	Not available	XML dump	fileGzip plugin	
	METASHARE	Not available	XML dump	Targz plugin	

In order to be able to process the X3ML mappings produced by metadata experts of research infrastructures via the 3M Editor tool, D-NET has been extended by integrating the X3ML Engine, a transformation engine developed by FORTH capable of processing X3ML mappings.

3.4 X3ML Engine

Target requirement/use case

The adoption of the X3ML Engine is required for the execution of mappings generated via the 3M Editor.

Description

The X3ML Engine is a transformation engine that realizes the transformation of the source records to the target format. The engine takes as input the source data (currently in the form of an XML document), the description of the mappings in the X3ML mapping definition file and the URI generation policy file and is responsible for transforming the source records into a valid RDF document which corresponds to the input XML file, with respect to the given mappings and policy.

The X3ML Engine implementation was initiated by the CultureBrokers project, co-funded by the Swedish Arts Council and the British Museum. Currently, FORTH is maintaining the implementation and has received funding from the projects ARIADNE, KRIPIS POLITEIA, and VRE4EIC.

The source code is available on GitHub[8] and is licensed under the Apache Licence 2.0. A ready to use artefact is provided via the FORTH ISL Maven Repository[9].

Integration into the PARTHENOS infrastructure

The X3ML Engine is integrated into D-NET, hence its deployment in the infrastructure is implicit with the deployment of D-NET.

Assessment

The X3ML Engine is a Java library and it has been integrated into D-NET. It is not a tool that is directly accessible by end-users. The evaluation phase is instead carried out by members of T6.2 from CNR-ISTI. The technical assessment focused on the feasibility of its integration into the PARTHENOS aggregator based on the D-NET software toolkit. In order for the integration to be possible, FORTH and CNR-ISTI collaborated to update the library with new Java methods and consolidate the code by upgrading some old library dependencies.

[8] X3ML Engine source code: https://github.com/isl/x3ml
[9] FORTH ISL Maven Repository: http://www.ics.forth.gr/isl/maven/

3.5 Metadata Cleaner

Target requirement/use case

The inclusion of the Metadata Cleaner was not initially planned, because the value cleaning can also be performed by defining specific rules in the X3ML mappings. However, the PARTHENOS Consortium agreed that a mechanism to ensure that all controlled fields (i.e. metadata fields whose values must comply to a controlled vocabulary) contain valid values was needed. For this goal, CNR-ISTI proposed to include in the D-NET instance of PARTHENOS the Metadata Cleaner so that, in the transformation phase of the aggregation workflow (see Figure 10), each record is transformed by the X3ML Engine and, afterwards, the controlled fields are further cleaned by the Metadata Cleaner. If a controlled field cannot be harmonised according to the proper vocabulary, the record is marked in order to enable inspection via the Metadata Inspector.

Description

The Metadata Cleaner is a D-NET service that harmonises values in metadata records based on a set of thesauri. A D-NET thesaurus consists of a controlled vocabulary that is a list of authoritative terms together with associations between terms and their synonyms. Data curators – typically based on instructions from data providers and domain experts – are provided with user interfaces to create/remove vocabularies and edit them to add/remove new terms and their synonyms. Given a metadata format, the metadata cleaner service can be configured to associate the metadata fields to specific vocabularies. The service, provided records conforming to the metadata format, processes the records to clean field values according to the given associations between fields and vocabularies. Specifically, field values are replaced by a vocabulary term only if the value falls in the synonym list for the term. If no match is found, the field is marked as ‘invalid’. The ‘invalid’ marker is exploited by the Metadata Inspector (see Section 3.5) to highlight non-cleaned records and suggest the update of D-NET vocabularies or the update of the values in the input record.

Integration into the PARTHENOS infrastructure

The Metadata Cleaner is integrated into D-NET, hence its deployment in the infrastructure is implicit with the deployment of D-NET.

Assessment

The Metadata Cleaner is natively integrated in D-NET and no issues had to be addressed at the technical level. The configuration of the Metadata Cleaner was evaluated by T2.3 members (with the support of T6.2 and T5.2). Specifically, the Metadata Cleaner has been configured to clean all fields that are defined as controlled fields in the PARTHENOS data model. For detailed information on the adopted vocabulary, see D5.4 "Report on integration of reference resources" [9].

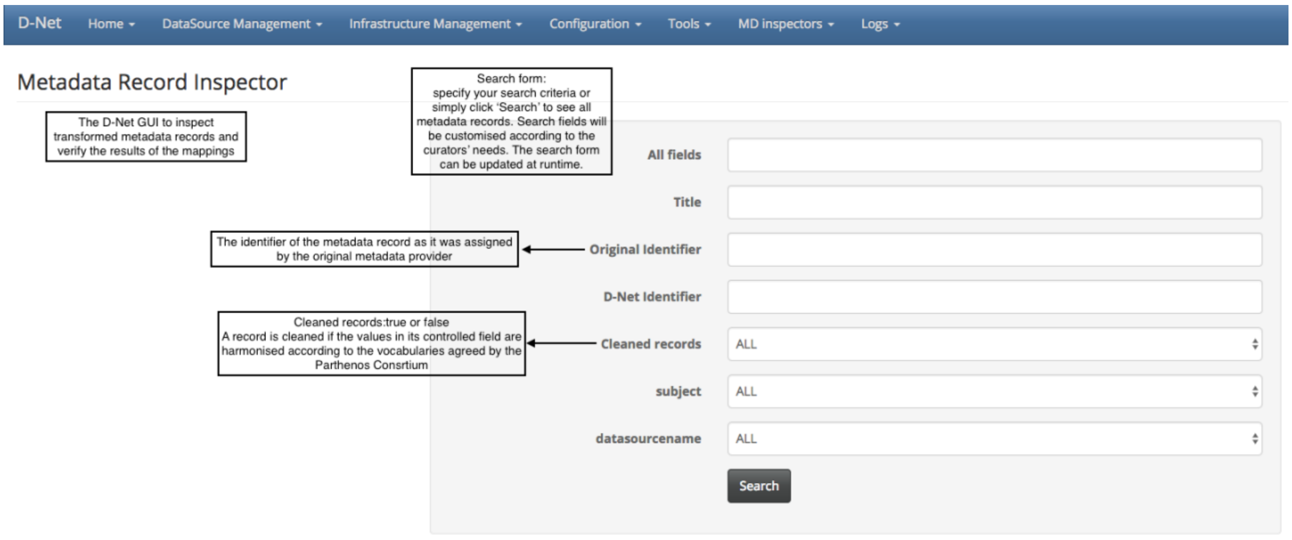
3.6 Metadata Inspector

Target requirement/use case

The PARTHENOS consortium expressed the requirement for having a tool to check the results of the transformation and the vocabulary harmonisation.

Description

The Metadata Inspector is a Web GUI integrated into D-NET that provides data curators with an overview of the information space, where they can search and browse records and verify the correctness of the transformation phase (e.g. no mapping mistakes or semantic inconsistencies, no records marked as 'invalid' by the Metadata Cleaner). Upon positive verification of the records in the information space, data curators can inform the PARTHENOS infrastructure administrators that the records can be published (see 'Publishing workflow' in Figure 10). Figure 11 and Figure 12 show two screenshots of the Metadata Inspector.



Metadata Record Inspector

The D-Net GUI to inspect transformed metadata records and verify the results of the mappings

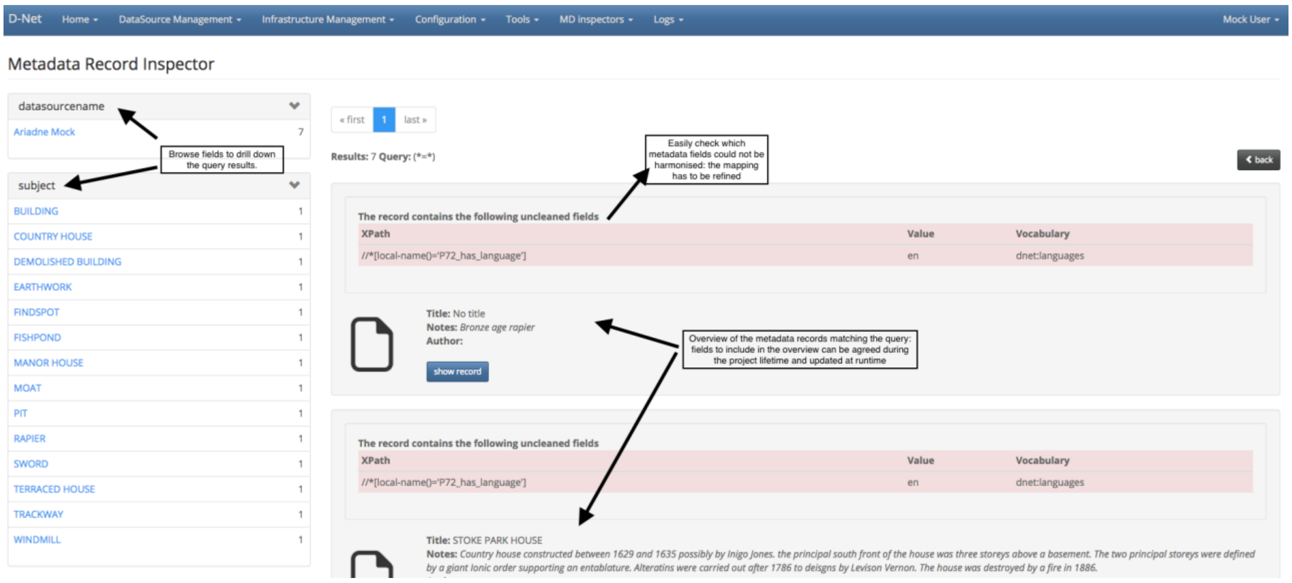
Search form: specify your search criteria or simply click 'Search' to see all metadata records. Search fields will be customised according to the curators' needs. The search form can be updated at runtime.

The identifier of the metadata record as it was assigned by the original metadata provider

Cleaned records:true or false
A record is cleaned if the values in its controlled field are harmonised according to the vocabularies agreed by the Parthenos Consrtium

All fields	
Title	
Original Identifier	
D-Net Identifier	
Cleaned records	ALL
subject	ALL
datasourcename	ALL

Figure 11. The main search form of the Metadata Inspector



Metadata Record Inspector

datasourcename	
Ariadne Mock	7

Browse fields to drill down the query results.

subject	
BUILDING	1
COUNTRY HOUSE	1
DEMOLISHED BUILDING	1
EARTHWORK	1
FINDSPOT	1
FISHPOND	1
MANOR HOUSE	1
MOAT	1
PIT	1
RAPIER	1
SWORD	1
TERRACED HOUSE	1
TRACKWAY	1
WINDMILL	1

Results: 7 **Query:** (*=*)

Easily check which metadata fields could not be harmonised: the mapping has to be refined

The record contains the following uncleaned fields

XPath	Value	Vocabulary
//*[local-name()='P72_has_language']	en	dnet:languages

Title: No title
Notes: *Bronze age rapier*
Author:

Overview of the metadata records matching the query: fields to include in the overview can be agreed during the project lifetime and updated at runtime

The record contains the following uncleaned fields

XPath	Value	Vocabulary
//*[local-name()='P72_has_language']	en	dnet:languages

Title: STOKE PARK HOUSE
Notes: *Country house constructed between 1629 and 1635 possibly by Inigo Jones. the principal south front of the house was three storeys above a basement. The two principal storeys were defined by a giant Ionic order supporting an entablature. Alteratins were carried out after 1786 to deisgns by Levison Vernon. The house was destroyed by a fire in 1886.*

Figure 12. The Metadata Inspector shows metadata records with "uncleaned" fields

Integration into the PARTHENOS infrastructure

The Metadata Inspector is integrated into D-NET, hence its deployment in the infrastructure is implicit with the deployment of D-NET.

Assessment

The Metadata Inspector is a native feature of D-NET and no issues had to be addressed at the technical level. Metadata experts of the research infrastructures (i.e. those who defined the mappings with the 3M Editor) used the Metadata Inspector to verify the metadata quality before the records are officially published. From the users and the ACE team, suggestions about the visualization of the RDF records and the available facets for browsing have been received and processed.

4. Sharing specialized tools (Task 6.3)

This task addresses the provision of tools/services for manipulating, presenting and publishing similar data within the disciplinary domains (or possibly outside). The task follows the same approach as the task 6.2 for the selection, the assessment, the design, implementation, evaluation and deployment of the tools/services. For example, these include advanced visualisation; annotation of texts/images/video/3D; workflow description; and more. The task harmonises such tools and make them available after adapting or making the changes necessary for generalization. The results are documented in a section of the interim report on tools, D6.2 [11], and likewise in the final report, D6.4, incorporating the amendments made after testing in D2.3.

Based on an exemplary use case, we have developed a blueprint for service providers (SP) describing the different options on how to integrate their services into the D4Science infrastructure.

This deliverable is an update of D6.2. Since the publication of D6.2 a number of modifications of the original plan were executed.

1. Some changes are based on a second evaluation of the proposed list of to integrate community services, especially in context of PARTHENOS partners having expertise about and access to those proposed services and those proposed services being suitable for integration in the PARTHENOS VRE.
2. Especially on the basis of the outcome of a technical workshop (Oct 17-19, 2017) selection criteria related to the technical expertise of community service specialists were added to aspects of usefulness for solving the WP2 use case inventory
3. A balanced view with respect to a shared responsibility for the landscape of SSH service infrastructure was discussed and accepted, where the infrastructure operated by different community's should be interoperable with the services made available in PARTHENOS.

4.1 Technical workshop PARTHENOS community service integration

This workshop was organised by task 6.3 and the PARTHENOS CNR-ISTI partners to allow the technical experts from the different community infrastructures discussion with the D4science experts. The goals for this workshop were to:

1. Get a (better) grasp of what is entailed for integrating community services into a PARTHENOS VRE using the D4science platform.
2. Determine which modes of integration are available and would be best suited for specific community services also with respect to the necessary maintenance effort.
3. Investigate which community services would be eligible for integration, based on the nature of the service (scope, complexity) and the PARTHENOS project accessible expertise.
4. Make a start discussing the place and added value of the PARTHENOS VRE's in the research landscape.

The workshop attracted a large number of technical experts and partly based on the previous selection of community services (D6.2 Table 3 [11]) a new selection of services was created as a basis for an integration work plan.

4.2 Overview of tools and services

The original overview of tools and services available in D6.2 Table 3, gave a good overview of what the different PARTHENOS stakeholder communities have developed over the years and are providing to their users. However, in the context of their integration in the PARTHENOS LAB VRE and the D4science technology PARTHENOS uses, some further selection and changes were needed. For instance, the Nederlab tool proposed in D6.2 Table 3 is a highly specialised corpus exploitation tool for searching through very large textual corpora, going far beyond the capacity and scope of PARTHENOS. Also integrating the different specific community metadata catalogues in PARTHENOS was abandoned in favour of a central metadata search facility based on the PARTHENOS metadata mapping (See D5.5 chapter 3).

Table 3. A selection of services offered by research infrastructures

Name	Technology/Service Provider	RI	Task
Ariadne Visual Media Service	CNR/ISTI	ARIADNE	Visualisation
CSTlemma	University of Copenhagen	CLARIN	NLP
RTFandplaintexttokenizer	University of Copenhagen	CLARIN	NLP
Distanbol	OEAW	CLARIN/DARIAH	NLP, visualisation
NER Liner2 PL	Wroclaw University	CLARIN	NLP, NER
NER Spacy EN	Wroclaw University	CLARIN	NLP, NER
NER Spacy DE	Wroclaw University	CLARIN	NLP, NER
Parser Maltparser PL	Wroclaw University	CLARIN	NLP
Parser Spacy EN	Wroclaw University	CLARIN	NLP
Parser Spacy DE	Wroclaw University	CLARIN	NLP
Distanbol	OEAW	CLARIN/DARIAH	NLP, visualization
Stanbolwrapper	OEAW	CLARIN/DARIAH	NLP
NLPchain	OEAW	CLARIN/DARIAH	NLP
Tagger Nltk EN	Wroclaw University	CLARIN	NLP, POS tagging
Tagger Spacy DE	Wroclaw University	CLARIN	NLP, POS tagging
Tagger Wcrtf2	Wroclaw University	CLARIN	NLP, POS tagging
RUBRICA[10]	SISMEL	E-RIHS	KE,
B2SHARE	EUDAT	EUDAT	infrastructure, Data Hosting
NLP Hub	CNR/ISTI	PARTHENOS	NLP

[10] PARTHENOS VRE unique service

Language Resource Switchboard	EKUT	CLARIN	infrastructure, service brokering
Gate Cloud	Sheffield University	none	NLP, various services

There is an abundance of services provided by the RIs and their supporting organisations. Often these are specialised, mainly of interest to experts in the domain of origin, but can often also be of interest to the other (sub-)domains in the SSH, e.g. the use of Natural Language Processing (NLP) technology to analyse large historical text corpora to obtain information on historical events described in those corpora.

Not all services provided by the RIs are interesting to offer through the PARTHENOS VRE since they would be overlapping with the new, to be built functions, being provided by PARTHENOS such as discovery services for data and services, that most RI already provide. PARTHENOS will provide a consolidated discovery service for data and services covering the whole SSH. Through this reasoning it was, for instance, decided to largely drop all metadata and discovery services from the services listed in D6.2 Table 3.

The high incidence of NLP services in Table 3 should also be seen in the context of their usefulness of such services for the broad Humanities and their large availability in many different versions, but also of our desire to test the PARTHENOS platform technology (D4science) as being an appropriate integration platform for the type of RI experts that are available to maintain the services once integrated. A major advantage of the integration of the different NLP services is the sharing of resources in the PARTHENOS Dataminer workspace, allowing users to create pipe-lines of NLP services using the Python Notebook feature.11

We experimented both with services that were completely running within the D4science environment, as well as with services running at remote sites accessed through a proxy service in the D4science Dataminer. We have two screencasts made by community software developers that illustrate both methods.

[11] The functionality is available as trial (thanks to EGI) in the PARTHENOS Lab VRE.

Further selection criteria were the availability of resources but especially of experts in the consortium. In many cases, expert knowledge of code and/or service API is required to integrate a service successfully.

The ARIADNE Visual Media Service (AVMS) is an excellent demonstration of how an application that needs a complex user-interface can be integrated into the PARTHENOS platform. Although such an integration goes beyond what a community expert can do using the automatic facilities available in the SAI/DataMiner integration mode (see section 4.5 Modes of Integration) that is used for most of the NLP service integrations. AVMS manages its own workspace, and resource selection mechanisms. Remaining important advantages of the AVMS integration are the use of the D4science integration of Authentication that is compatible with the EOSChub12 and the monitoring and resource scaling.

The NLP Hub also offers a comprehensive user interface, going beyond the standard facilities of specifying an input file and setting a number of parameters as standard available for the services integrated in the data-miner. Such integration is beyond the standard data-miner type of service integration and as in the case of AVMS, requires intervention from the D4science development team, but obviously can be more effective and useful for the end-user.

We included the integration of two ‘infrastructure’ type of services. The first is the “Language Resource Switchboard” that can broker the invocation of applicable services on specific types of resources. It allows users to select services from its large database of CLARIN language technology services, beyond those that are integrated in PARTHENOS. The second one “B2SHARE” which allows users to share and publish data files from PARTHENOS via the EOSC-hub B2SHARE service. Both services are part of the EOSC-hub portfolio and next to their direct usability by end-users, their integration demonstrates the integration of the PARTHENOS platform with other e-Infrastructures and RIs.

[12] https://marketplace.eosc-portal.eu/services/c/security-operations

4.3 User evaluation

The integrated services were evaluated by a team from WP2 and compared with the original use-cases mentioned in D6.2 chapter 1.8 (see D2.4 chapter 8). According to the evaluation the integrated specialised services were largely meeting the use-cases drawn-up by WP2, and some criticism was voiced about available user guidance and documentation within the PARTHENOS VRE for those services. In collaboration with the WP2 evaluation team additional documentation was added to the PARTHENOS platform.

4.4 Service Integration - Stakeholder User stories

In the previous version of this deliverable D6.2 [11], we referred to use cases provided by WP2. These introduced high level abstract actions from the user-perspective. In D6.4 we also add to those user stories the perspective of communities' efforts in operating and integrating services in PARTHENOS.

The PARTHENOS VRE offered the different stakeholder organisations, e.g. the community infrastructures and their constituent organisations, the opportunity to learn about the possibilities of integrating services using the D4science technology and evaluate its usefulness to them, weighing the effort of integration versus the advantages of sharing a common platform that provides users access to specialised cross-community services and potentially also access to general infrastructure resources.

We distinguish a number of use-cases and considerations from the perspective of the community infrastructure as a 'user' of the PARTHENOS platform:

1. In most cases, specialised community services are already operated and available from the community infrastructure itself. Added value from integration in PARTHENOS needs to come from being able to invoke specialised services in a common context e.g. using each others services from a single environment and sharing input and output data. All the integrated NLP services are within this class, End-users can experiment chaining different NLP services in pipe-lines. For some of the available NLP pipelines the DataMiner are quite specific e.g. “Twitter Opinion Mining” and “Brexit Analyzer Pipeline” and results can be used directly without further processing.

2. In a number of cases, the community infrastructure does not have the facility and resources to run specialised technology services. In this case, the PARTHENOS platform offers of course an excellent way of hosting such services e.g. RUBRICA (See D2.4 Chapter 8.4.1) in offering users access to other integrated specialised and general services.
3. Gateway to other infrastructures e.g. CLARIN, EOSC-hub, semi-public service offers as the Gate Cloud NLP services.

With respect to the effort involved, an increased proficiency was seen, of course, of the community experts in integrating their specialised services. The effort of integrating a service decreased from months (incidental) and several weeks to days. This was also closely related to being able to contact the right D4science experts, which went smoothly in the end.

The current PARTHENOS platform provides examples of all three use-cases, and gave the stakeholders ample experience in the incurred effort and required expertise versus the benefits for their end-users. The PARTHENOS stakeholders should make an evaluation about the added value of the PARTHENOS VREs compared to their own (potential) infrastructures.

4.5 Modes of integration

Services/tools are launched from a hosting environment. In gCube terms, two types of hosting node typologies are distinguished, gCube-based and SmartGears-based. With gCube-based nodes, services are launched and operated on nodes of the D4Science infrastructure. An advantage of this approach is that it provides the possibility to automatically scale up, whereas the SmartGears option requires installation.

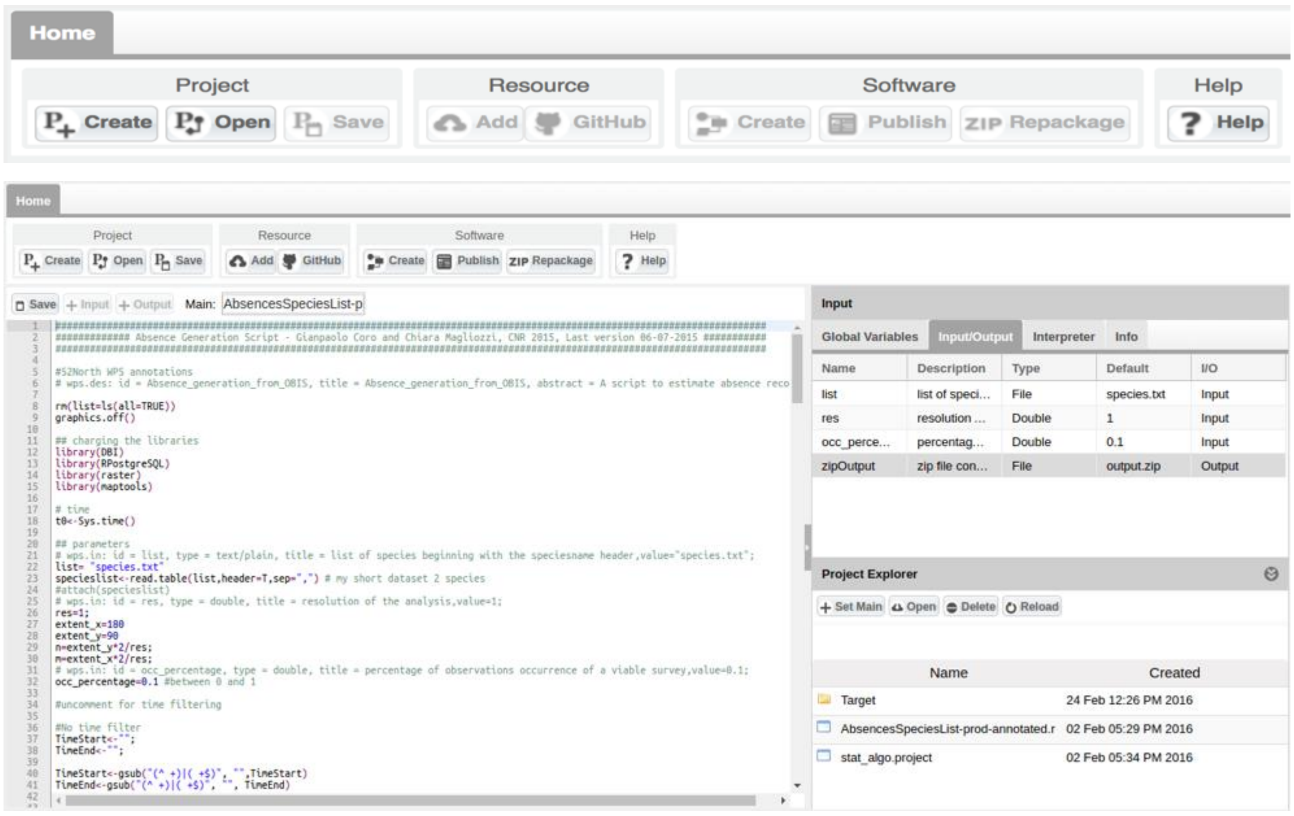
gCube’s enabling layer distinguishes between two types of hosting node typology: Software-as-Resource (SaR) and Container-as-Resource (CaR). In the SaR approach, a piece of software is considered a gCube resource if it can be managed in the gCube infrastructure. The CaR approach focuses on software that can be used over the network. They are typically deployed within containers, e.g. Tomcat, and may be registered to become gCube Hosting Nodes (gHN).

This flexible setup allows service providers to either keep offering their services on their own servers, adapted to become gHN, or to transfer the software to the facilities of the D4Science infrastructure. With this second option, the D4Science infrastructure is responsible for ensuring the availability of server-side capacities (storage, computing power, network bandwidth). Starting initially with the gHN approach for integration and testing and then moving to the central infrastructure for production is a very feasible and sensible scenario.

For most services delivered through the RIs, the CaR option appears to provide the most viable option as this allows for deployment and maintenance of the services from within the participating RIs. Participation of a container requires installation of SmartGears on the host, registration of the host with the infrastructure and acquisition of authorization tokens.

In order for hosted services to operate in the PARTHENOS infrastructure, the easiest approach appears to be Web Processing Service (WPS) compliant. In almost all cases, this requires a proxy to be deployed alongside the service or a wrapper encapsulating the service, with the benefit of not having to change the service (or the underlying software). To facilitate the importing process of services in the PARTHENOS infrastructure, a new Importer Tool named SAI was made available. SAI complements the Gold integration pattern (achieved by “wrapping” the method as a WPS method) by providing scientists with an interface that allows them to easily and quickly import Java, Python or R scripts onto the general-purpose gCube-based data analytics engine. Additionally, it allows scientists to update their scripts without following long software re-deployment procedures previously necessary.

The SAI Web application interface (Figure 13) resembles the R Studio environment, a popular IDE for R scripts, in order to make it friendly to script providers.



```
##########################################################################################################
############## Absence Generation Script - Gianpaolo Coro and Chiara Magliozzi, CNR 2015, Last version 06-07-2015 ###########
##########################################################################################################

#52North WPS annotations
# wps.des: id = Absence_generation_from_OBIS, title = Absence_generation_from_OBIS, abstract = A script to estimate absence reco

rm(list=ls(all=TRUE))
graphics.off()

## charging the libraries
library(DBI)
library(RPostgreSQL)
library(raster)
library(maptools)

# time
t0<-Sys.time()

## parameters
# wps.in: id = list, type = text/plain, title = list of species beginning with the speciesname header,value="species.txt";
list= "species.txt"
specieslist<-read.table(list,header=T,sep=",") # my short dataset 2 species
#attach(specieslist)
# wps.in: id = res, type = double, title = resolution of the analysis,value=1;
res=1;
extent_x=180
extent_y=90
n=extent_y*2/res;
m=extent_x*2/res;
# wps.in: id = occ_percentage, type = double, title = percentage of observations occurrence of a viable survey,value=0.1;
occ_percentage=0.1 #between 0 and 1

#uncomment for time filtering

#No time filter
TimeStart<-"";
TimeEnd<-"";

TimeStart<-gsub("(^ +)|( +$)", "",TimeStart)
TimeEnd<-gsub("(^ +)|( +$)", "", TimeEnd)
```

Name	Description	Type	Default	I/O
list	list of speci...	File	species.txt	Input
res	resolution ...	Double	1	Input
occ_perce...	percentag...	Double	0.1	Input
zipOutput	zip file con...	File	output.zip	Output

Name	Created
Target	24 Feb 12:26 PM 2016
AbsencesSpeciesList-prod-annotated.r	02 Feb 05:29 PM 2016
stat_algo.project	02 Feb 05:34 PM 2016

Figure 13. SAI Interface to import new services in gCube-based data analytics engine.

In all the integration scenarios, the comprehensive documentation accompanying the gCube framework represents invaluable input.

In summary, the following options are available for service integration:

- Local gCube hosting node - register service, allows monitoring, authentication/authorisation.
- Service as servlet (Tomcat or another servlet container)
- Not a servlet (requires a wrapper)
- Access cloud storage
- Read
- Write
- Run service hosted on the nodes of D4Science infrastructure (with the possibility to scale out)

From user’s perspective, the integrated services become part of a custom on-demand VRE, as a main organizing unit for putting together users, resources and services around a specific research question or task. Still under discussion is whether PARTHENOS will work towards a small number of bigger VREs with a broad function spectrum or multiple smaller, more specialized, ones. The NERLIX VRE (see Section 6) combines the strategy of service integration (‘How do I integrate my own service’) with the end user perspective as expressed through the D2.1 Use cases and requirements [7].

5. Resource discovery tools (Task 6.5)

In the overall architecture of PARTHENOS, resource discovery tools are located at the end of a complex workflow which involves harvesting of the source metadata from the providers, mapping to common semantic model, cleaning and further transformation and ingestion into different persistence layers. These offer different means to query and explore the data in the vast dataspace of PARTHENOS.

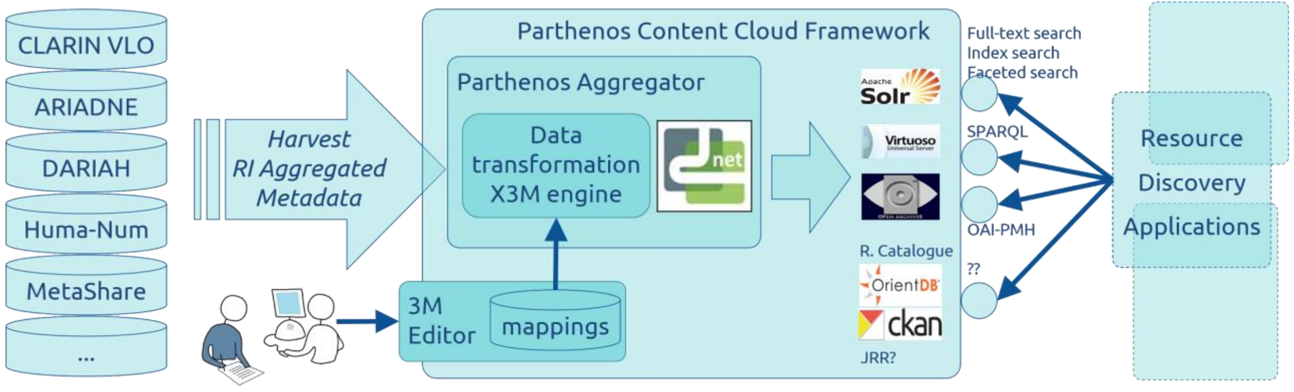


Figure 14. Overall aggregation and discovery workflow

After an extensive evaluation (c.f. section 5.1), we decided to use the solution Metaphactory as the base system to customize for the needs of resource discovery in the PARTHENOS data space. The resulting solution is called PARTHENOS Discovery. It is to be regarded also in tandem with the Joint Resource Registry (JRR). While JRR provides basic functionality for (faceted) search & browsing through the data, PARTHENOS Discovery (PD) offers a much richer means to query the data and visualize the results. This plethora of features, however, comes with a certain penalty on ease of use. In addition, in the process of exploring the data space, we have developed a custom scripting tool, called SparqLaborer, to allow for automatized processing of sets of SPARQL queries in response to the need to systematically oversee and evaluate various aspects of PARTHENOS' data.

5.1 Evaluation of possible solutions

In Task 6.5, an extensive evaluation of existing solutions for resource discovery has been conducted in order to identify most promising candidates for integration with PARTHENOS. Due to the available capacity in the project and the host of existing solutions, development from scratch was ruled out. In the evaluation process we considered a broad range of tools based on three different basic principles:

- Faceted search
- SPARQL-based
- Graph-based.

The evaluation was conducted against a set of requirements, formulated in D6.2:

- Combined faceted / index-based full-text search (allowing for browsing, drilling down, as well as specific string-based queries).
- Full-text, Content-first search (autocomplete/suggestions, ideally typed and quantified) well known from Google (or your browser).
- Complex queries / Advanced search (allowing to query specific indexes, applying boolean operators, fuzzy search, regular expressions, joins).
- Metadata & content search
- Semantic search (especially Named entities)
- Handle arbitrarily deep hierarchies
- Handle arbitrary relations
- Pointers to original location [of data & metadata] (ideally all locations/identifiers)
- Inform about available indexes
- Invoke processing services (on resources and result sets)
- Harmonized access/API to heterogeneous sources
- Personalized Workspace
- Harvesting/Download
- Feedback

Table 4 contains the evaluated tools with brief summary of the evaluation findings:

Table 4: Summary of the evaluation of discovery tools

Tool	Description and Pros	Supported requirements	Cons and major problems/Issues
FactForge	Based on GraphDB Great visualisation, easy to navigate, relatively scalable, owl:sameAs and inferences (on and off)	Fully/Mostly 5,8 Partially 1,2,3,4,6,7,11,13 No support 9, 10,12,14	Only pre-defined data sets; not meant for custom datasets or local installation
Neo4j	Native graph database Many modern visualization tools available on top of the datastore, wide graph-community support behind it, future potential for expansion	Fully/Mostly 3,6,7 Partially 1,2,4,5,8,11,12,13 No support 9, 10,11,14	Neo4J's data model does not support RDF natively, requiring transformation of data from RDF, adding unnecessary complexity Performance/ Scalability issues
Virtual Language Observatory / PARTHENOS instance (Pavlov)	VLO is CLARIN metadata catalogue based on solr offering facetted search over aggregated records. Given the similarity of the setup, it was considered as base for one possible	Fully/Mostly: 1,2,3,8,13 Partially: 6,7,9,10 (with Switchboard)	Lack of key features such as semantic search Similar in functionality (faceted browsing) with Joint Resource Registry
QuePy	Inspirational for natural language question-answer system which generates SPARQL query	Fully/Mostly 4,5 Partially 1,2,3 No support	Only pre-defined data sets; not meant for custom datasets Insufficient for

	Possibility to build a SPARQL query without technical background, sample queries (which are working) are available as autocomplete, editing function for the generated SPARQL query, clarification/transparency of query building and process	6,7,8, 9,10,11,12,13,14	serious use: The generated query is suboptimal. It is difficult to create complex questions
YAGO2	Big semantic query engine with a couple of useful concepts: query builder and user feedback on data quality Simple query builder to avoid learning SPARQL queries, entities autocomplete and interesting properties list Time and Location dimension (as well as Subject-Predicate-Object) which can be plotted on GoogleMaps and SMILE Timeline, straightforward navigation for inbound and outbound properties	Fully/Mostly 5,8 Partially 1,2,3,6,7,11,13 No support 4,9,10,12,14	Only pre-defined data sets; not meant for custom datasets Old-fashioned interface, no visuals (images etc), difficult to understand what kind of data is returned only by looking at property names
MultiWiBi	Inspirational in terms of graph-based visualisation and result filtering Speedy visualisation	Fully/Mostly 5,8 Partially 1,2,3,6,7	Not meant for custom datasets

	Easy to expand the view with parameter sliders Could be interesting for inferences	No support 4,9,10,11,12,13,14	Old-fashioned interface and visualisation
Pelagios / Peripleo	Dedicated to collaborative annotation and linking of historic places Impressive spatio-temporal visualisation (Peripleo) Allows for intuitive spatio-temporal filtering of resources	Fully/Mostly 4,5,8 Partially 1,2,3,6,7,13,14 No support 9,10,11,12	Not meant for custom datasets Only for spatio-temporal data
Palladio	Generic Open Source visualisation tool to foster SPARQL query over any arbitrary endpoint, with table, graph, gallery, map, timeline views	Fully/Mostly 5,8 Partially 1,2,3,6,7,13 No support 4,9,10,11,12,14	Only capable to operate on a fixed limited dataset, does not allow to interactively query a large triple store
LodLive	Generic Open Source visualisation tool which allows to navigate the RDF graph (explore the neighbourhood of any arbitrary node) Graph structure is the very basis of RDF, so presenting it in such a way provides authentic insight into the data, lightweight and smooth in any browser, connect to public SPARQL endpoints	Fully/Mostly 5,8 Partially 1,2,3,4,6,7,13 No support 9, 10,11,12,14	Project seems dead since May 2017, has crashed a few times, displaying each entity as separate node in a graph, the layout of which keeps on changing, is very inefficient and problematic from usability point of view.

As seen in the table, there are several interesting open source solutions currently available, especially in the context of cultural heritage domain. However, the level of product readiness and the suitability for PARTHENOS varies, especially when evaluated against the backdrop of the feature list. A major showstopper in many cases is the fact that the given solution is just used to explore a specific predefined dataset, i.e. it is not applicable as a general solution for custom data as is required in the context of PARTHENOS. Nevertheless, most of the evaluated solutions feature functionalities that are interesting and very relevant for the task of resource discovery and thus can serve as source of inspiration.

5.1.1 Focus on visualisation tools

Next to general exploration tools, we additionally focused on tools specialized in visualisation. This is based on the assumption that visualisation is a critical aspect of resource discovery and that integration of visualisation components in a larger discovery framework is worth consideration. These tools can play a role in three distinct parts of the workflow:

- Support search - e.g. plotting available data points on a map, allowing to filter the dataset visually.
- Visualize aggregations - e.g. all kinds of quantitative statistics over the metadata of the result set.
- Explore a specific resource - for a complex resource, e.g. a graph representing the social network of Viennese society in 19th century, ideally a tool would offer dedicated means to browse and navigate through the underlying graph.

Based on the categorisation, we have tested and assessed several tools (see Table 5). The scope of our investigation was broad, taking into account not only ready-made tools, but also software libraries that would need to be integrated into any overall solution, but could provide higher flexibility and a richer feature-set. Probably the main motivation for the evaluation was the quest for inspiration for our solution.

Table 5. Visualisation tools and their functionalities for PARTHENOS workflow

Tools	Support search	Visualize aggregations	Explore a specific resource
Gephi Popular desktop client for graph analysis	Free search is not provided. The network can be filtered to select nodes and/or edges based on the network structure or data. An interactive user interface is used to filter the network in real-time. - Create complex filter query without scripting - Build new networks from the filtering result - Save your favourite queries	Tree /graph /network	Graph based real time visualization. Each resource corresponds to a node. The related data are displayed as table However, it is not possible to view and select only individual nodes and gradually explore their neighbourhood
Matplotlib	Python plotting library. Search parameters customized for each implementation according to the script configuration	Quantitative /statistical	Different plotting types are provided. Resources are visualized / explored according to the script configuration.
Bokeh	Python plotting library. Search parameters customized for each implementation according to the script configuration	Bar-chart, radio, pie-chart, sankey, boxplot,	Different plotting types are provided. Resources are visualized / explored according to the script configuration.

D3.js	Popular Javascript visualisation library, with many sample implementations. Search parameters can be customized for each implementation according to the script configuration	Bar-chart, radio, pie-chart, sankey, boxplot,	Through custom implementation, gradual navigation through the graph can be implemented. (Reference implementations available)
GeoBrowser	Free search / Geospatial / Timeline filtering	Geographical, tabular view, timeline	Resource(s) can be selected by the map view or by the tabular view. Each point of interest on the map shows the number of the resource(s) related to the point. More information are provided by the tabular view once the resource has been selected.
eLINDA	No search functionality	Graph, chart Visualisation of RDF triple store. Users can interactively navigate themselves statistical bar charts representing RDF classes, properties, and	Only aggregated bar chart representation. No view for individual resources

		instances. They can specify any public SPARQL endpoint. Easy filtering function.	

5.1.2 Metaphactory as most promising candidate

One of the evaluated tools was ResearchSpace used by the British Museum. ResearchSpace is a customisation of Metaphactory, a professional full-featured powerful user-friendly knowledge management platform. In particular, we were impressed by the query builder function, which enables the users to formulate complex queries by simply selecting top entities (e.g. Actor, Place, Time, Event, and Object) and relationships between them, e.g. “hasMet” or “isAbout”, the system generating appropriate SPARQL query and returning the result set. In addition, it is positively surprising that the system architecture relies exclusively on SPARQL for all aspects of data management, including ingestion, manipulation, publication, and visualisation. The software clearly demonstrates the state-of-the-art architecture and data management model as well as design and interface for the Semantic Web.

In addition, we considered as significant advantage that ResearchSpace uses CIDOC-CRM as a base ontology, although customised for their own purposes. It was foreseen that Research Space is ideal in terms of the adaptability of the PARTHENOS Entity Model which is based on CIDOC-CRM.

Taking the impressive maturity of the software and CIDOC-CRM applicability into consideration, we decided to use ResearchSpace (RS) as the base for the resource discovery tooling in the task 6.5. We started to experiment with the open source code of RS in a local instance. However, we encountered substantial difficulties in deploying the system, due to various configuration issues and undocumented and opaque customisations of the base system and were not able to establish efficient communication with the persons responsible. This led us to the decision to purchase one year of service support for Metaphactory. The extra cost spent is outweighed many times by the dramatic increase in efficiency and fluency in the adaptation and work with the platform. This allows investment of the limited resources in the task to concentrate on system integration, configuration, and customisation, offering the end-users the sophisticated design and functionalities of Metaphactory.

In the following sections, as a main part of this chapter, we detail the general setup of the discovery solution and the customisation based on Metaphactory.

Table 6: Metaphactory support against the defined requirements

Metaphactory	Requirement reference and comments
Fully or mostly supported	1,2,3,5,6,7,8,9,13
Partially supported	4: no content search so far 11: Metaphactory allows to connect to any external repository (triple store), and query even multiple repositories in one view. However only RDF-based repositories (triple stores) are supported
Functions exist, but not yet activated in our instance	14: there is an annotation functionality by default, but not fully explored and evaluated within the project
Under development	10: the integration with the Language Resource Switchboard (LRS) and DataMiner functionality of D4science is underway 12: user management system is not activated

With respect to visualisation, Table 7 illustrates which features are present within Metaphactory. The only missing element at the moment is the lack of interactivity for most of the views. In contrast, eLINDA is equipped with dynamic filtering of data space, specialising in a bar chart view, where the users can navigate through the network of RDF data such as classes, properties, and instances. However, it is possible to add interactivity to Metaphactory in the future, as it has a highly customisable templating function.

Table 7: Metaphactory visualisation features

Support search	Visualize results	Explore a specific resource
Full-text search Autocomplete based on full-index and/or imported vocabularies (typed entities) Query builder which uses simplified top-level entities ("Virtual materialisation" of relationships) Save queries SPARQL endpoint	interactive UI-components: - Table(list) view - Gallery view - Map view - Graph view - Chart view - Timeline view	Highly customisable detail view (based on templates for all classes and/or subclasses) All visualisation views can be integrated in the detail view as needed Links to related Instances, Classes, and Properties Image preview Pagination to explore a large dataset Automatic assignment of unique URI

5.2 PARTHENOS Discovery (PD) - Metaphactory customisation

Even though Metaphactory represents a highly complex application, by default it comes as a bare-bones platform that requires a fair amount of customisation to fit the data at hand and the specific needs of the user base.

It offers a rich selection of pre-defined widgets that can be easily combined into custom complex pages, working purely in web-forms in a browser. These components get dynamically filled with data solely through SPARQL requests. Altogether, this setup offers a clean, transparent, reproducible way of building a custom application on top of Metaphactory.

5.2.1 Overall structure

PD consists of four main parts (further described in the following subsections):

Homepage - main entry point, offering first overview of the dataspace

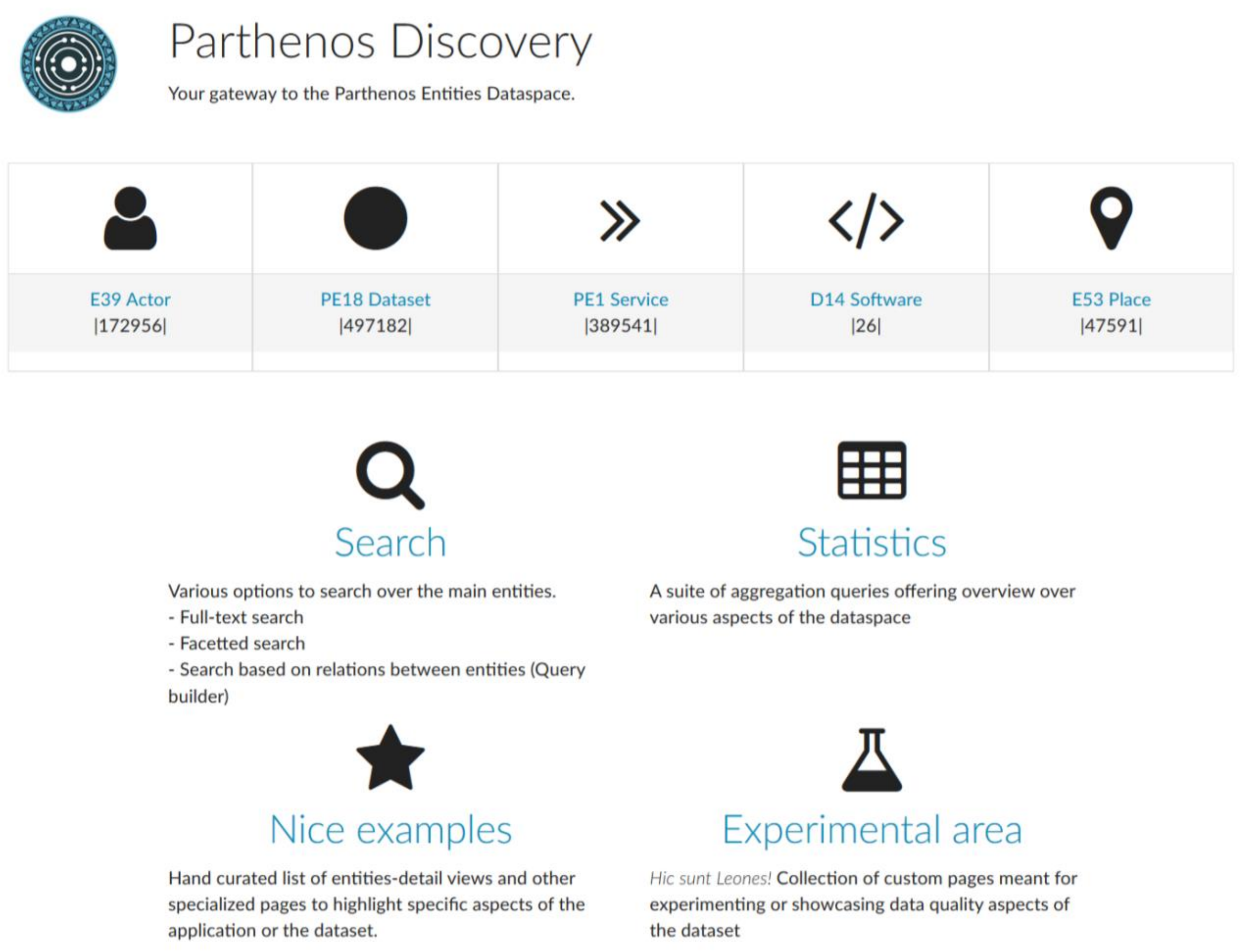
Search - simple keyword search supported by faceting, as well as more complex search based on relations between entities with custom display of results

Statistics - tables and charts with statistics about various aspects of the data space

Detail view - pages displaying information about individual entity - its properties and relations to other entities

5.2.2 Homepage

The Homepage is the default entry point that dispatches user to the main areas and gives a first basic overview of the dataset, division by main entities, including counts of instances per entity and a direct link to a type-constraint search allowing to search, e.g. only for Actors, or Datasets. Although it is possible (and in fact easier) to search over all entities, the motivation here is to give a first intuitive entry point to the vast data space of PARTHENOS, similar to the frontpage of the Joint Resource Registry. Below the main entities banner, short teasers point to the main parts of the applications.



Parthenos Discovery

Your gateway to the Parthenos Entities Dataspace.

E39 Actor	PE18 Dataset	PE1 Service	D14 Software	E53 Place
|172956|	|497182|	|389541|	|26|	|47591|

Search

Various options to search over the main entities.
- Full-text search
- Facetted search
- Search based on relations between entities (Query builder)

Statistics

A suite of aggregation queries offering overview over various aspects of the dataspace

Nice examples

Hand curated list of entities-detail views and other specialized pages to highlight specific aspects of the application or the dataset.

Experimental area

Hic sunt Leones! Collection of custom pages meant for experimenting or showcasing data quality aspects of the dataset

Figure 15. Starting page of PARTHENOS Discovery

5.2.3 Search

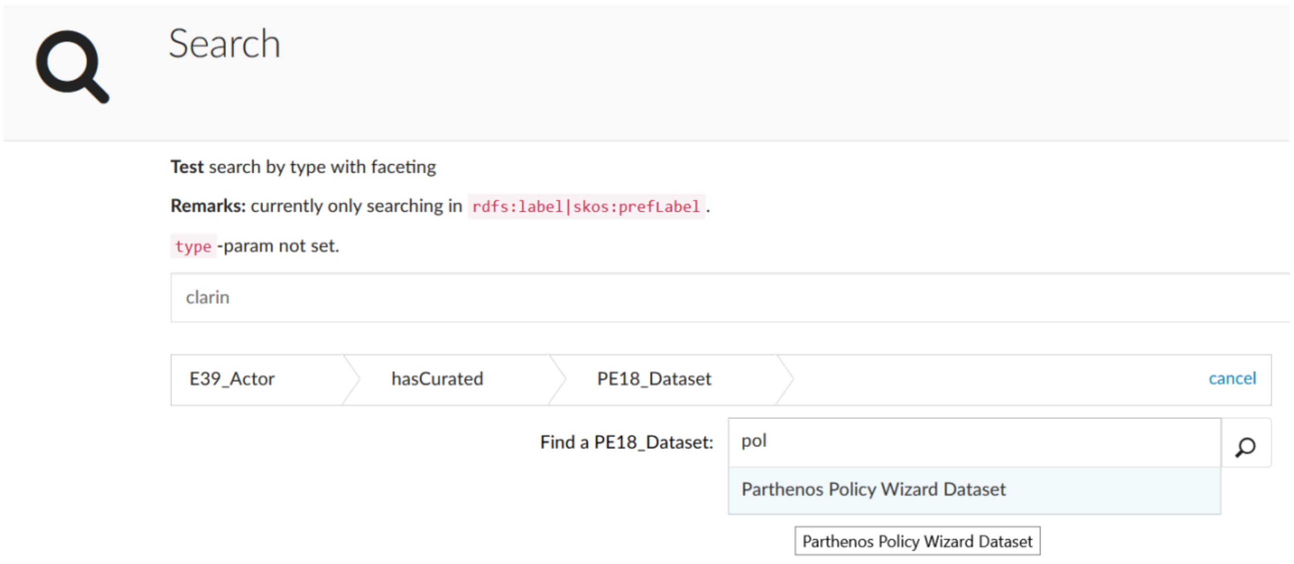
Metaphactory offers a coherent set of UX-components for querying the data. Next to the simple keyword search there is the so-called query-builder, a UX-component that guides the user through formulating more complex queries by offering relevant categories and relations between them to pick from.

Customisation:

- Adapt the query builder to offer the main categories (entity types) and relations of the PARTHENOS Entities Model (ongoing)
- Restrict (simple) search results to relevant (top-level) classes (filter out classes like Appellation, (Creation) Events, etc.)
- Custom result-view merging instances with multi-instantiation into one row (collapsing multiple classes into one field)

- Search button - by default, the search components fire requests automatically upon user typing. There is an alternative widget that offers the user a traditional Search button.

5.2.3.1 Query builder


Search

Test search by type with faceting

Remarks: currently only searching in `rdfs:label|skos:prefLabel`.

`type` -param not set.

clarin

E39_Actor hasCurated PE18_Dataset cancel

Find a PE18_Dataset: pol

Parthenos Policy Wizard Dataset

Parthenos Policy Wizard Dataset

Figure 16. Metaphactory query builder customized to PE model

5.2.4 Statistics

To get an overview of such a complex and vast data space as the one offered by PARTHENOS, it is indispensable to provide the user with various aggregations over the data space. In PD, we implemented a whole set of pages dedicated to such overviews, starting from the Statistics overview page. This is envisaged as an extensible suite of statistical summaries, currently covering following dimensions:

- Data Providers - how much data is coming from each provider?
- Classes - how many instances of which class are present?
- Properties - what properties are used?
- Types - PEM makes extensive use of the CIDOC-CRM typing construct p02_hasType -> E55 Type
  A dedicated page shows which types are most frequent and also distinguished use of types per Class, e.g. the CIDOC-CRM class crm:E51_Contact_Point may be further categorized with E55 Type as “email”, “phone” or “fax”

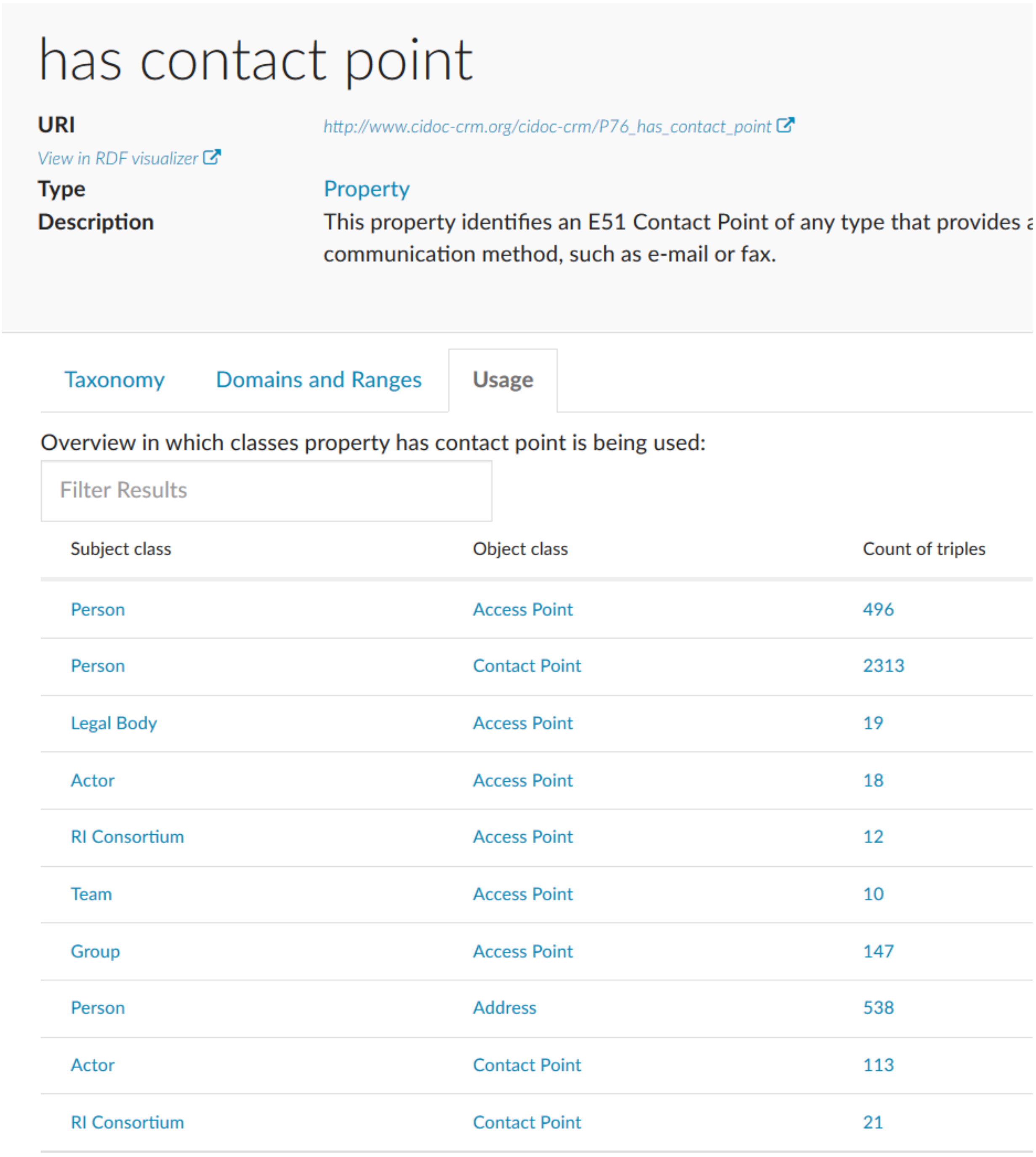
- Aboutness - Another crucial property in CIDOC-CRM is crm:P129_is_about which links a Dataset to a Concept (or in fact a crm:E89_Propositional_Object to any crm:E1_CRM_Entity).

Sample results for the aggregations mentioned above are included in the Data quality section 6.3 of D5.6.

A major advantage of the consequent use of templating mechanism in combination with SPARQL queries becomes apparent, when looking on the meta or ontology level. Classes and properties defined in the PEM (or in fact any other ontology) are equally available for inspection in custom views as the data itself. In fact, Metaphactory comes with pre-built generic templates for classes and properties. These have been further customized to combine the ontology information with the instance information. Under the tab [Usage] in both Class and Properties templates, the various usage patterns for every class or property13 and their frequency are listed. Figure 17 illustrates such a customized view on the example of the property *P76 has contact point*.

One down-side of the flexible powerful templating system is that all SPARQL queries are computed upon user’s request. This poses an unnecessary, prohibitive burden on the underlying triple store, forcing it to run resource-heavy statistical queries over the whole dataset every time a user requests a site. As remedy, measures are taken to pre-compute and cache the aggregation results, or add additional inferred triples upon data ingest, allowing for faster querying.

[13] I.e. with which combination of object types given property appears how often


has contact point

URI http://www.cidoc-crm.org/cidoc-crm/P76_has_contact_point

View in RDF visualizer

Type Property

Description This property identifies an E51 Contact Point of any type that provides a communication method, such as e-mail or fax.

Overview in which classes property has contact point is being used:

Subject class	Object class	Count of triples
Person	Access Point	496
Person	Contact Point	2313
Legal Body	Access Point	19
Actor	Access Point	18
RI Consortium	Access Point	12
Team	Access Point	10
Group	Access Point	147
Person	Address	538
Actor	Contact Point	113
RI Consortium	Contact Point	21

Figure 17. Sample of a customized view for the property, on the example of the property *P76 has contact point*

5.2.5 Detail View

While navigation, search and statistics are necessary for the user to find his/her way through the data space, in the end the most relevant information is the detailed description of individual resources. Metaphactory allows the presentation of data in manifold ways, based on a powerful templating mechanism in combination with possibility to define specific fields for giving information on relations and attributes. The following describes both the templating mechanism and the fields and details the implementation of the PARTHENOS Discovery solution, by means of customising the underlying Metaphactory system.

5.2.6 Templates

Templates are a kind of master documents describing how to render certain kind of data. The more generic a template, the less it may be capable to cover special cases. Therefore, the decision on the granularity of templates needs to be balanced with respect to the conceptual data model in use.

In Metaphactory framework, the logic which template is used for rendering of which entity is governed by the type (Class) of the corresponding entity, respecting also the class hierarchy. This means that a template for an instance of a given class is applied to all instances of all subclasses as long as there is no more specific template for one of the subclasses.
In PD currently five templates for the most relevant entities of PEM were created:

- E39 Actor
- PE35 Project
- PE1 Service
- E70 Thing (mainly geared towards PE18 Dataset)
- E53 Place.

The following figure illustrates the templating mechanism:

  - The user requests a representation of some resource.
  - The templating engine retrieves the information that this resource is of type PE25 RI Consortium , and consulting the underlying ontology it determines the super-classes of PE25 RI Consortium: PE34 Team, E74 Group, E39 Actor, etc. and checks if there is a template defined for given class.
  - Template for E39 Actor is present.
  - It will be used to generate a web view for the requested resource.

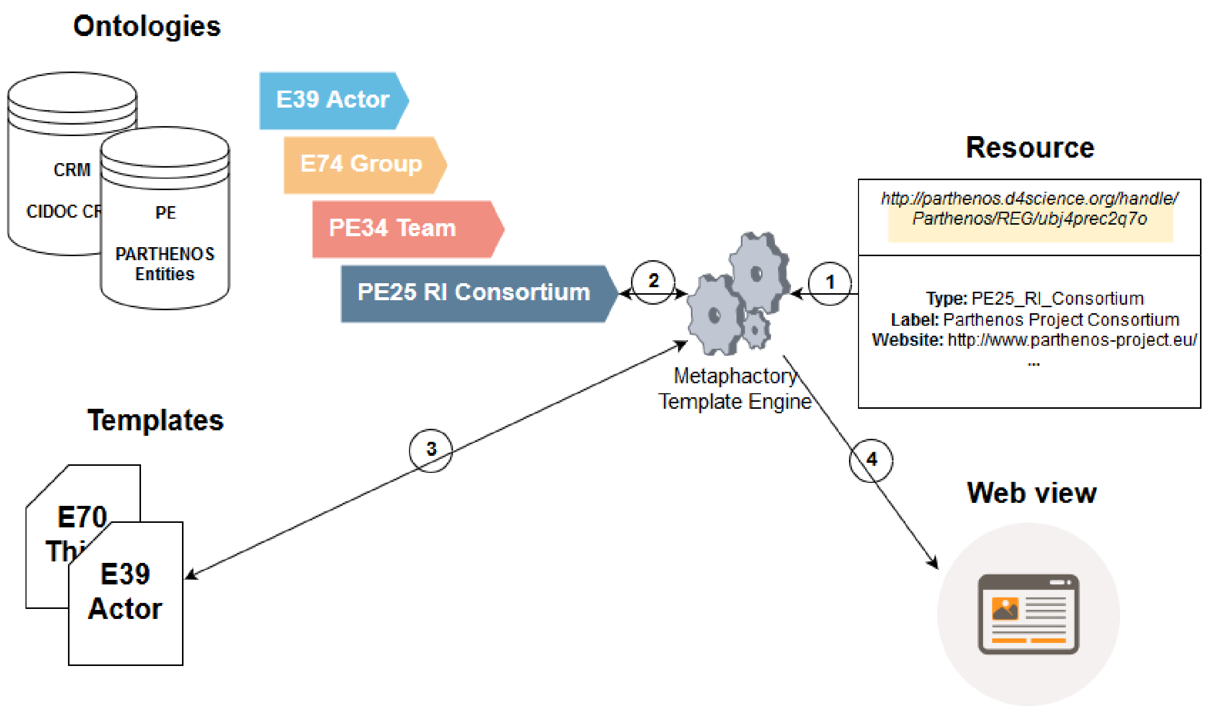
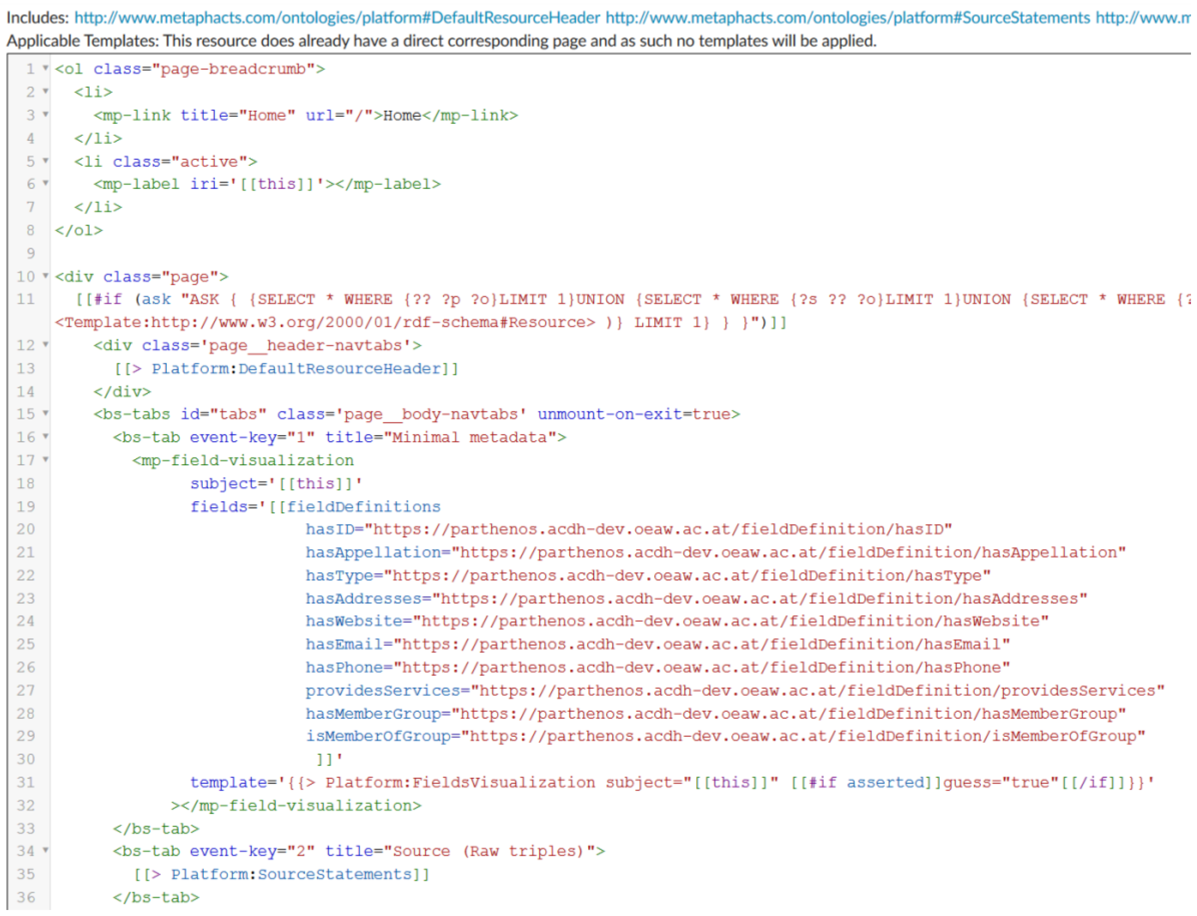

Ontologies
CRM
CIDOC CR
PE
PARTHENOS Entities
E39 Actor
E74 Group
PE34 Team
PE25 RI Consortium
Resource
http://parthenos.d4science.org/handle/Parthenos/REG/ubj4prec2q7o
Type: PE25_RI_Consortium
Label: Parthenos Project Consortium
Website: http://www.parthenos-project.eu/
...
1
2
3
4
Metaphactory Template Engine
Templates
E70 Thi
E39 Actor
Web view

Figure 18. Metaphactory Templating mechanism

As a result of this templating process, the individual records or entities are rendered in a human-readable way and are displayed via the web browser. The templating mechanism, taking into account class hierarchy, offers a generic yet flexible way of handling complex data models with many classes. It allows even an existing template to be applied to an instance of a new class, if the conceptual model is extended accordingly.

There are entities that do not have a custom-tailored template defined. Nevertheless, they can be inspected, as Metaphactory offers a default generic detail view rendering applicable to all entities. If deemed necessary, a custom template for any given class will be introduced.

Templates are defined using a combination of plain HTML elements, custom HTML5 Web Components, and a minimal set of markers/commands for controlling the processing by the templating engine. The next figure displays parts of the E39 Actor template in the template editor of the Metaphactory framework.



Includes: http://www.metaphacts.com/ontologies/platform#DefaultResourceHeader http://www.metaphacts.com/ontologies/platform#SourceStatements http://www.m

Applicable Templates: This resource does already have a direct corresponding page and as such no templates will be applied.

```
<ol class="page-breadcrumb">
  <li>
    <mp-link title="Home" url="/">Home</mp-link>
  </li>
  <li class="active">
    <mp-label iri='[[this]]'></mp-label>
  </li>
</ol>

<div class="page">
  [[#if (ask "ASK { {SELECT * WHERE {?? ?p ?o}LIMIT 1}UNION {SELECT * WHERE {?s ?? ?o}LIMIT 1}UNION {SELECT * WHERE {?
<Template:http://www.w3.org/2000/01/rdf-schema#Resource> )} LIMIT 1} } }")]]
    <div class='page__header-navtabs'>
      [[> Platform:DefaultResourceHeader]]
    </div>
    <bs-tabs id="tabs" class='page__body-navtabs' unmount-on-exit=true>
      <bs-tab event-key="1" title="Minimal metadata">
        <mp-field-visualization
              subject='[[this]]'
              fields='[[fieldDefinitions
                          hasID="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasID"
                          hasAppellation="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasAppellation"
                          hasType="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasType"
                          hasAddresses="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasAddresses"
                          hasWebsite="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasWebsite"
                          hasEmail="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasEmail"
                          hasPhone="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasPhone"
                          providesServices="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/providesServices"
                          hasMemberGroup="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasMemberGroup"
                          isMemberOfGroup="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/isMemberOfGroup"
                           ]]'
              template='{{> Platform:FieldsVisualization subject="[[this]]" [[#if asserted]]guess="true"[[/if]]}}'
            ></mp-field-visualization>
      </bs-tab>
      <bs-tab event-key="2" title="Source (Raw triples)">
        [[> Platform:SourceStatements]]
      </bs-tab>
```

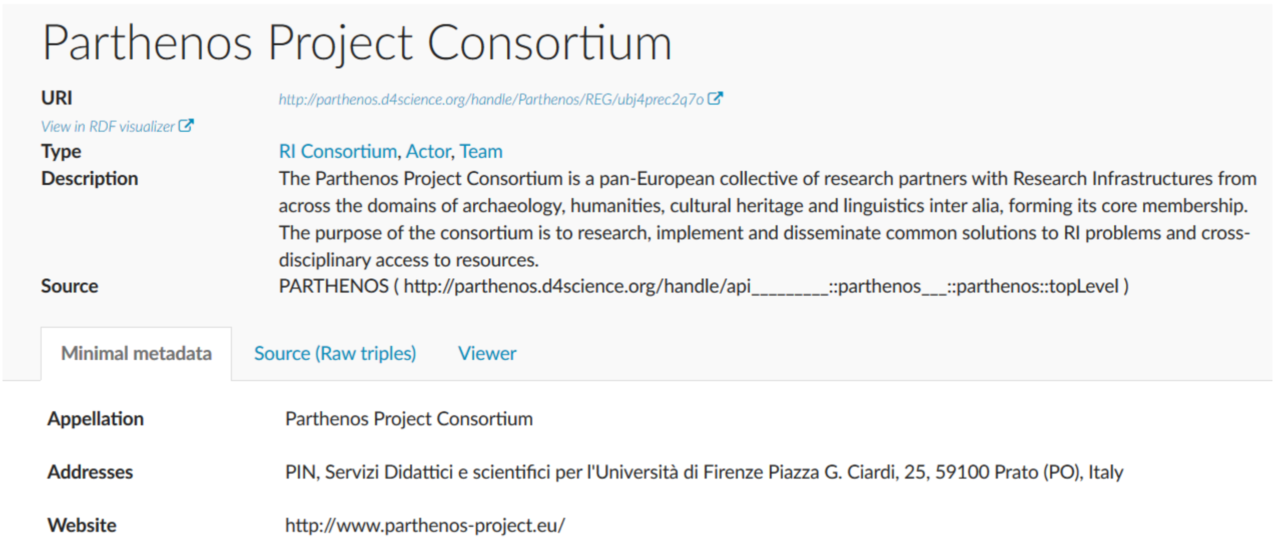
Figure 19. Template for E39 Actor in the Metaphactory template editor

The template uses normal HTML elements to organize the content (e.g. <ol>, <li>, <div>). There are additional tags that call the Metaphactory components (e.g. <mp-field-visualization>), integrating specific functions for handling data from the underlying data space. It is also possible to call other templates (using “[[> {Name of template}]]”), allowing for a modularisation of the template code.

The templates defined in PD consist of following generic components:

Header	This shows information that all of the entities have in common, e.g. URI, Type, Description
Minimal metadata	A defined set of information that is specific to a template
Source (Raw triples)	This shows all the incoming and outcoming statements of the inspected resource; mainly useful for verbatim investigation of the underlying RDF data for expert users.
Viewer	This calls the RDFViewer, showing the RDF of the dataset in an easy to browse view.

When calling a resource, usually all of these four views of the data are shown to the user, the three latter ones as individual tabs. The following figure illustrates this on the example resource “PARTHENOS Project Consortium” of type PE25 RI Consortium:



Parthenos Project Consortium

URI View in RDF visualizer	http://parthenos.d4science.org/handle/Parthenos/REG/ubj4prec2q7o
Type	RI Consortium, Actor, Team
Description	The Parthenos Project Consortium is a pan-European collective of research partners with Research Infrastructures from across the domains of archaeology, humanities, cultural heritage and linguistics inter alia, forming its core membership. The purpose of the consortium is to research, implement and disseminate common solutions to RI problems and cross-disciplinary access to resources.
Source	PARTHENOS (http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel)

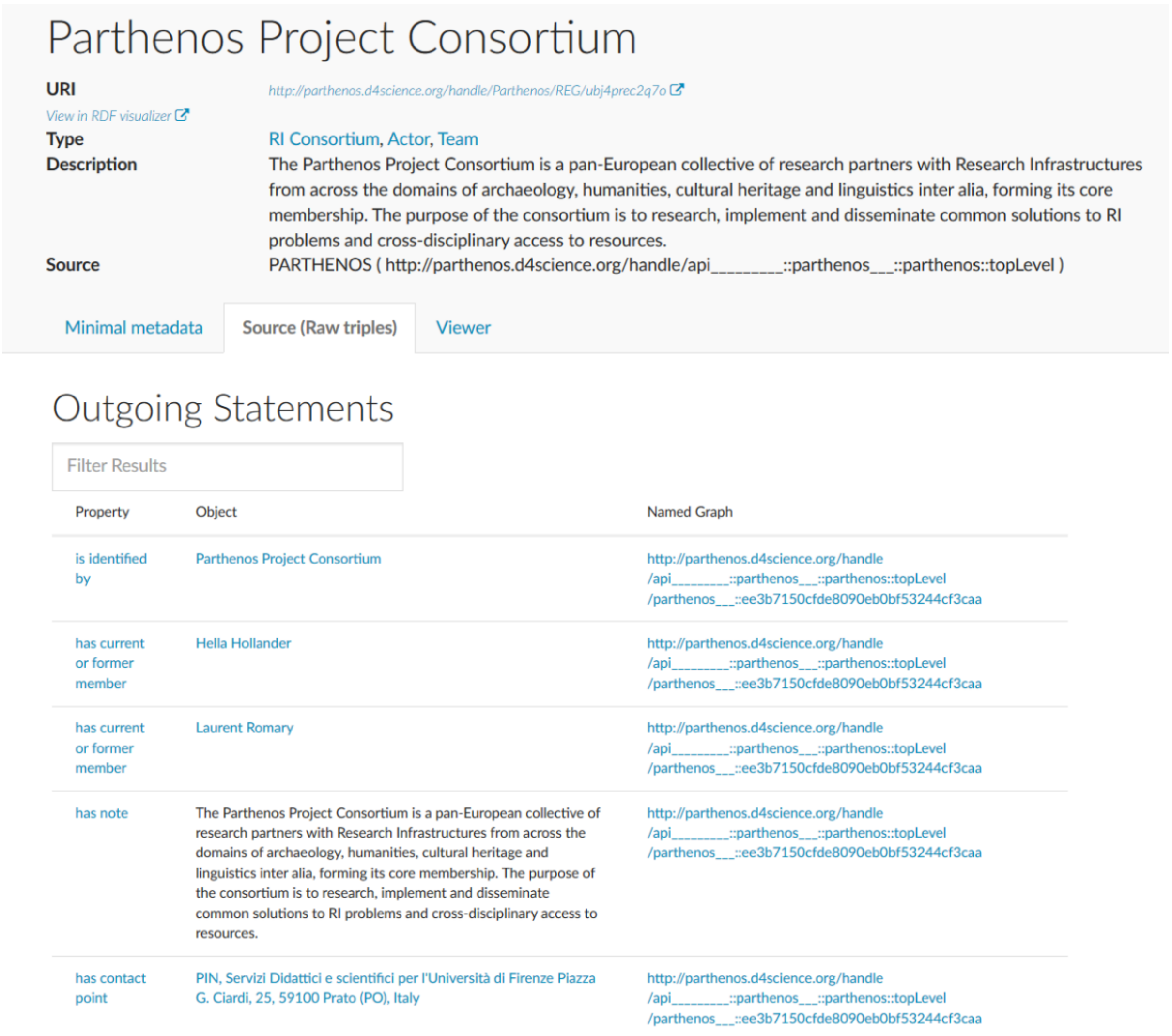
Appellation	Parthenos Project Consortium
Addresses	PIN, Servizi Didattici e scientifici per l'Università di Firenze Piazza G. Ciardi, 25, 59100 Prato (PO), Italy
Website	http://www.parthenos-project.eu/

Figure 20. Detail view with “Minimal metadata” tab active for the resource “PARTHENOS Project Consortium”

The header displays basic generic information about a resource, most importantly its URI, i.e. the identifier of the resource. It also informs about the type (class) of a resource (can be also more than one). The source field informs about the data provider of this resource.

The Minimal metadata is the default tab that is shown. It lists the most important information about a resource in a condensed manner, hiding details irrelevant for the normal user. It is governed by the definition of fields (see next subsection).

As there can be also more information about a resource beyond the minimal metadata, the tab “Source (Raw triples)” shows all outgoing and incoming statement (triples):



Parthenos Project Consortium

URI	http://parthenos.d4science.org/handle/Parthenos/REG/ubj4prec2q7o
View in RDF visualizer	
Type	RI Consortium, Actor, Team
Description	The Parthenos Project Consortium is a pan-European collective of research partners with Research Infrastructures from across the domains of archaeology, humanities, cultural heritage and linguistics inter alia, forming its core membership. The purpose of the consortium is to research, implement and disseminate common solutions to RI problems and cross-disciplinary access to resources.
Source	PARTHENOS (http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel)

Outgoing Statements

Property	Object	Named Graph
is identified by	Parthenos Project Consortium	http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel/parthenos___::ee3b7150cfde8090eb0bf53244cf3caa
has current or former member	Hella Hollander	http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel/parthenos___::ee3b7150cfde8090eb0bf53244cf3caa
has current or former member	Laurent Romary	http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel/parthenos___::ee3b7150cfde8090eb0bf53244cf3caa
has note	The Parthenos Project Consortium is a pan-European collective of research partners with Research Infrastructures from across the domains of archaeology, humanities, cultural heritage and linguistics inter alia, forming its core membership. The purpose of the consortium is to research, implement and disseminate common solutions to RI problems and cross-disciplinary access to resources.	http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel/parthenos___::ee3b7150cfde8090eb0bf53244cf3caa
has contact point	PIN, Servizi Didattici e scientifici per l'Università di Firenze Piazza G. Ciardi, 25, 59100 Prato (PO), Italy	http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel/parthenos___::ee3b7150cfde8090eb0bf53244cf3caa

Figure 21. Detail view with “Source” tab active

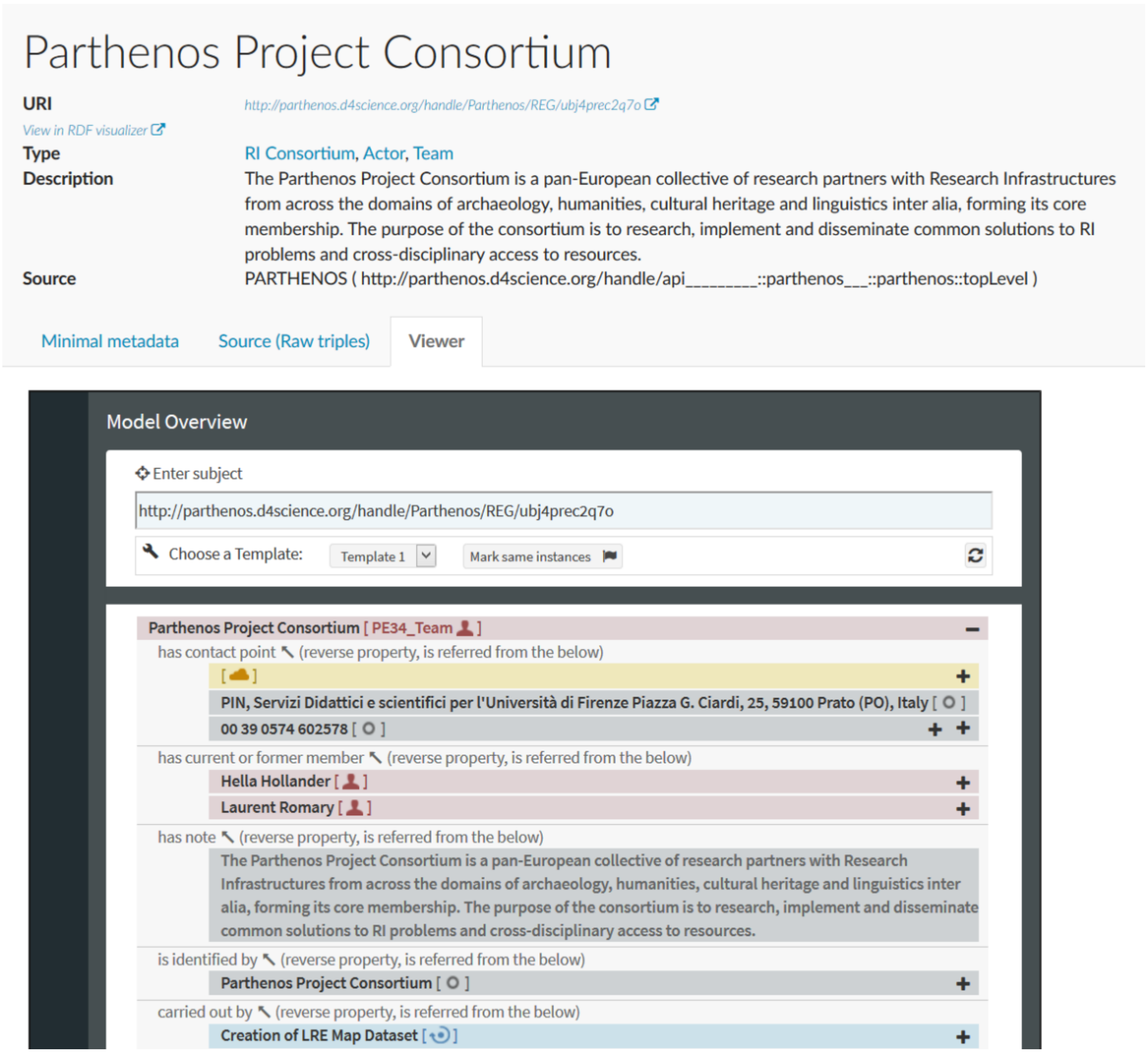
5.2.7 Integration of RDF Visualizer into PD Detail View

As a combination of the two abovementioned views, the viewer tab integrates the RDFViewer (cf. section 2.2 for more information on RDFViewer) that displays all the available information, however in a more condensed and more readable manner:

The RDF Visualizer is a tool for generic displaying and browsing of RDF data. It offers a highly condensed, flexible, configurable, detailed view of an RDF dataset, designed with human-readability in mind, aimed to overcome the drawbacks of the existing RDF data visualization methods/tools, most notably graph-based solutions.

As such it is a natural addition to the functionality of PD, and has been integrated as an alternative view on information about any given resource. It has been included via iframe in one of the generic templates, with following adaptations:

- Entities links in RDF Visualizer open back in PARTHENOS Discovery, allowing seamless browsing between related entities,
- Refactoring/optimisation of the SPARQL queries,
- Activation of the reverse properties functionality,
- Elimination of the header and sidebar.



Parthenos Project Consortium

URI http://parthenos.d4science.org/handle/Parthenos/REG/ubj4prec2q7o
View in RDF visualizer
Type RI Consortium, Actor, Team
Description The Parthenos Project Consortium is a pan-European collective of research partners with Research Infrastructures from across the domains of archaeology, humanities, cultural heritage and linguistics inter alia, forming its core membership. The purpose of the consortium is to research, implement and disseminate common solutions to RI problems and cross-disciplinary access to resources.
Source PARTHENOS (http://parthenos.d4science.org/handle/api________::parthenos___::parthenos::topLevel)

http://parthenos.d4science.org/handle/Parthenos/REG/ubj4prec2q7o

Parthenos Project Consortium [PE34_Team]
has contact point (reverse property, is referred from the below)
[ ]
PIN, Servizi Didattici e scientifici per l'Università di Firenze Piazza G. Ciardi, 25, 59100 Prato (PO), Italy [ ]
00 39 0574 602578 [ ]
has current or former member (reverse property, is referred from the below)
Hella Hollander [ ]
Laurent Romary [ ]
has note (reverse property, is referred from the below)
The Parthenos Project Consortium is a pan-European collective of research partners with Research Infrastructures from across the domains of archaeology, humanities, cultural heritage and linguistics inter alia, forming its core membership. The purpose of the consortium is to research, implement and disseminate common solutions to RI problems and cross-disciplinary access to resources.
is identified by (reverse property, is referred from the below)
Parthenos Project Consortium [ ]
carried out by (reverse property, is referred from the below)
Creation of LRE Map Dataset [ ]

Figure 22. Detail view of the tab "Viewer" for the resource "PARTHENOS Project Consortium" with integrated RDF Viewer

In summary, templates are used in PD to allow different perspectives on the data. Normal users will usually start by looking at the minimal metadata whereas expert users have the possibility to explore the verbatim information as stored in the individual triples. The template engine allows very specific approaches to show the data. For PD some generic templates were defined, that are used for similar data. That is based on the hierarchy of the conceptual model.

5.2.8 Fields

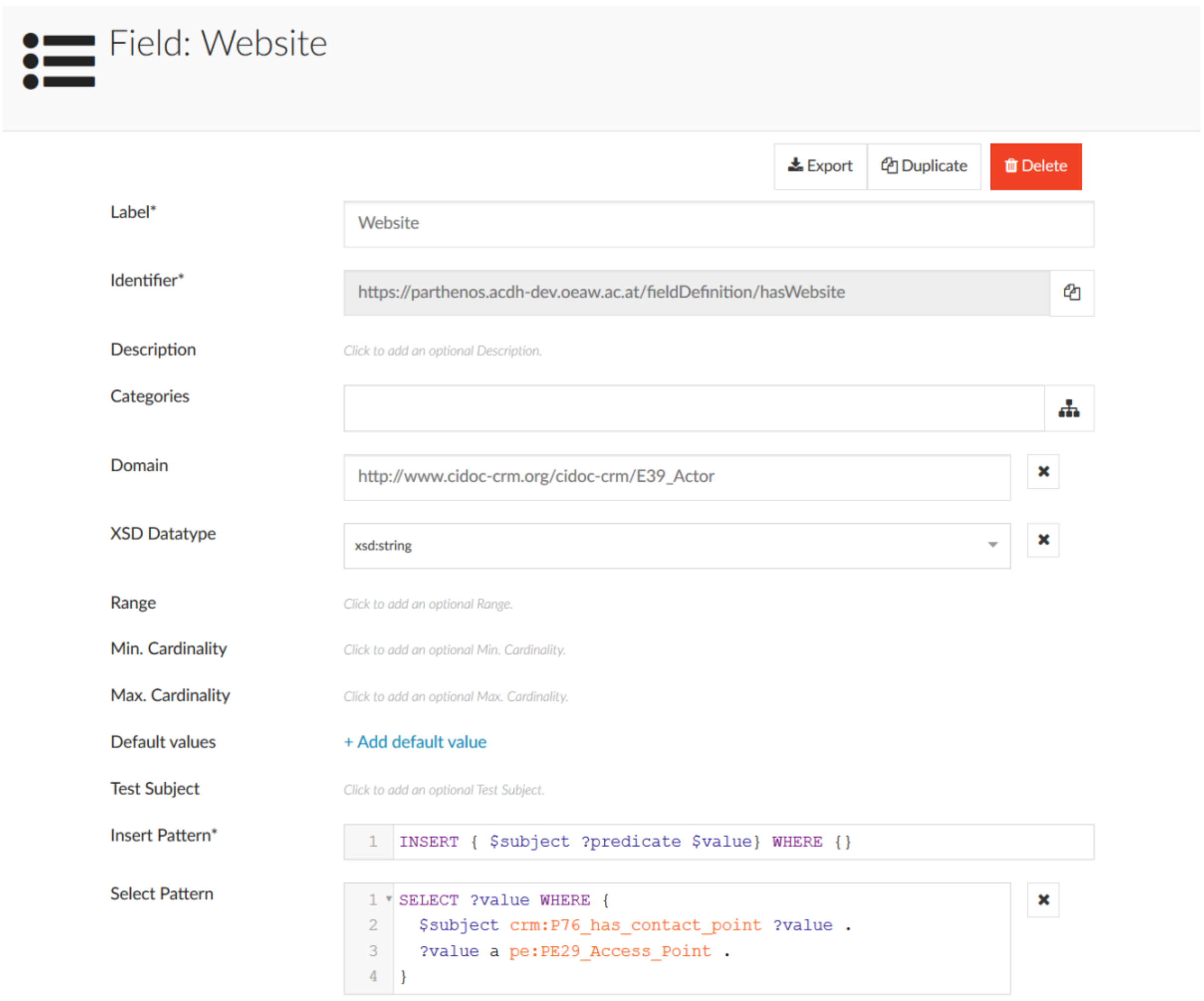
The templating mechanism of Metaphactory is accompanied by so-called 'Fields' that can be then integrated into templates. The interplay of templates and Fields controls the output. Fields are a mechanism introduced in Metaphactory to hide the internal complexity of the underlying data model behind structures known and understandable by the normal user. Conceptually these are based on the idea of Fundamental Categories and Relationships for querying CIDOC CRM based repositories, introduced by Tzompanaki and Doerr (2012). In summary, the fields offer a transparent abstraction layer over the underlying data.

This principle is illustrated in the following example:
An important piece of information regarding a person or an institution is the website. In the underlying conceptual model, this is defined as relation of E39 Actor:
E39 Actor - P76 has contact point -> PE29 Access Point
In PD, we defined a field "hasWebsite" with the SELECT pattern:

```
SELECT ?value WHERE {
  $subject crm:P76_has_contact_point ?value .
  ?value a pe:PE29_Access_Point .
}
```

The field can then be used in templates (in this case the E39 Actor template), its population and rendering being handled automatically by the Templating engine of Metaphactory by processing the SPARQL query defined in the field parametrizing it with the current context resource.

Metaphactory offers the means to define and manage custom fields by giving them a user-friendly name and formulating the underlying, potentially complex SPARQL queries to read the value of the field from the underlying resource graph, but optionally also to write new information into the graph based on user input.



Field: Website

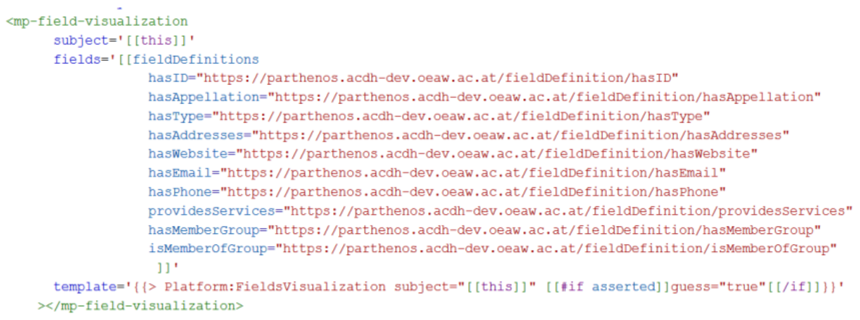
Label*	Website
Identifier*	https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasWebsite
Description	Click to add an optional Description.
Categories	
Domain	http://www.cidoc-crm.org/cidoc-crm/E39_Actor
XSD Datatype	xsd:string
Range	Click to add an optional Range.
Min. Cardinality	Click to add an optional Min. Cardinality.
Max. Cardinality	Click to add an optional Max. Cardinality.
Default values	+ Add default value
Test Subject	Click to add an optional Test Subject.
Insert Pattern*	INSERT { $subject ?predicate $value} WHERE {}

Select Pattern

```
SELECT ?value WHERE {
  $subject crm:P76_has_contact_point ?value .
  ?value a pe:PE29_Access_Point .
}
```

Figure 23. Metaphactory field editor

The field itself needs also to be integrated into a template. This is done by the HTML5 component mp-field-visualization, that is part of the Metaphactory system.



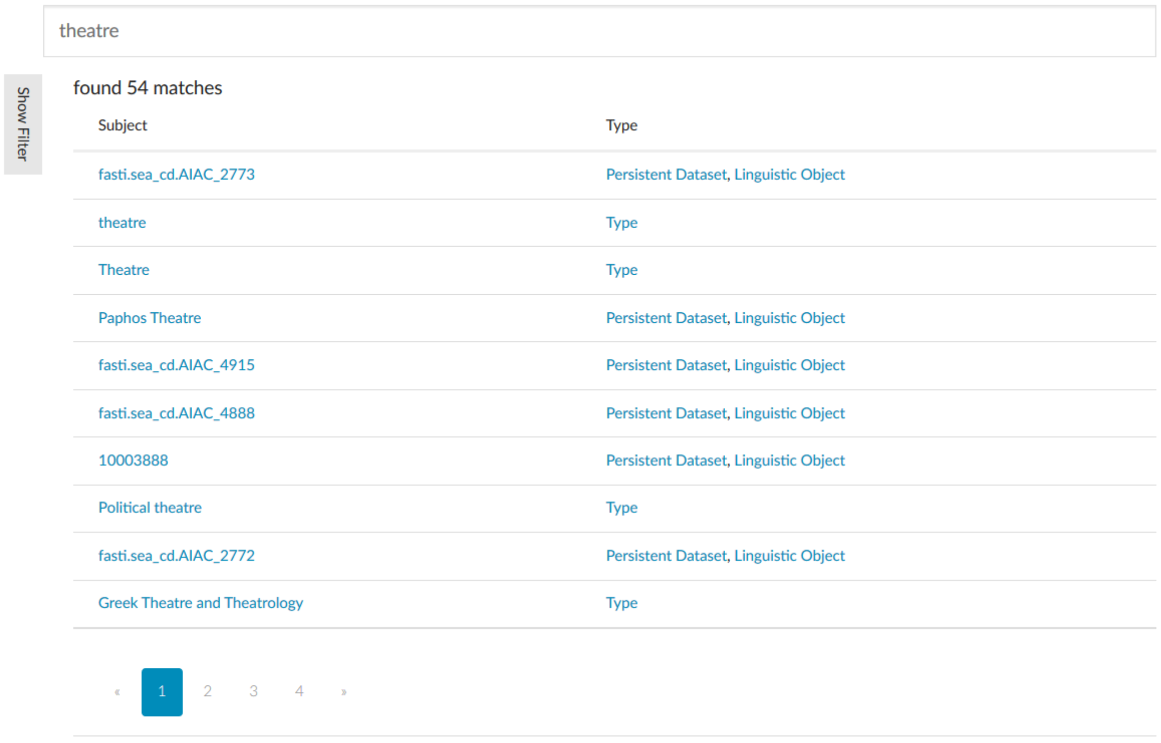
```
<mp-field-visualization
      subject='[[this]]'
      fields='[[fieldDefinitions
                  hasID="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasID"
                  hasAppellation="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasAppellation"
                  hasType="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasType"
                  hasAddresses="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasAddresses"
                  hasWebsite="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasWebsite"
                  hasEmail="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasEmail"
                  hasPhone="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasPhone"
                  providesServices="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/providesServices"
                  hasMemberGroup="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/hasMemberGroup"
                  isMemberOfGroup="https://parthenos.acdh-dev.oeaw.ac.at/fieldDefinition/isMemberOfGroup"
                   ]]'
      template='{{> Platform:FieldsVisualization subject="[[this]]" [[#if asserted]]guess="true"[[/if]]}}'
    ></mp-field-visualization>
```

Figure 24. Integration of fields into a template by means of custom element <mp-field-visualization>

5.3 Exploration

Exploration is a way of non-targeted browsing through the dataspace. Having a starting point, users like to get an impression of the connected data and like to inspect the relations to other datasets. In doing so, new insights can be expected. It is also a way to get an understanding which kind of data there is. This can be the basis for more detailed, result-oriented analysis. PD readily supports this scenario, by resolving all links to some human-readable representation for given entity. By implementing the minimal metadata view, the most relevant relations between entities are highlighted offering salient/sensible results.

Let’s imagine a use case, where a researcher wants to get information on theatre buildings, that is available in the PARTHENOS dataspace. The starting point is a search for a “theatre”:



theatre

Show Filter

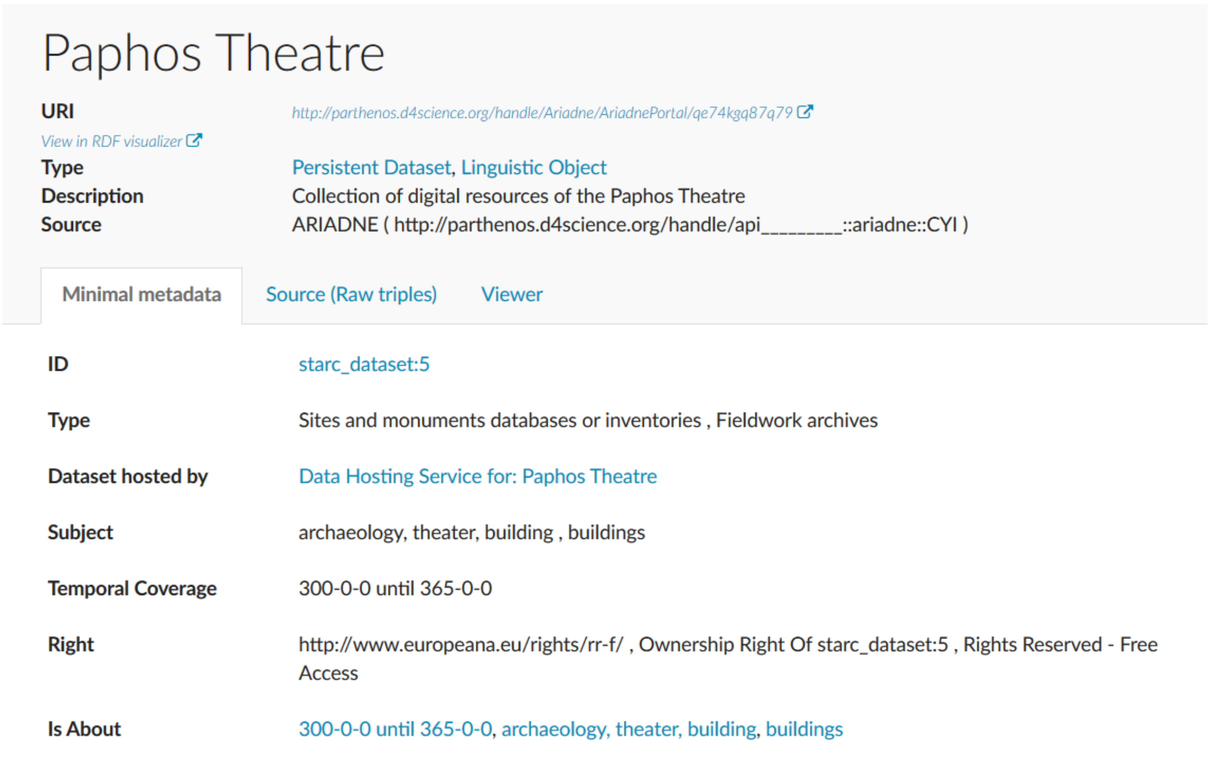
found 54 matches

Subject	Type
fasti.sea_cd.AIAC_2773	Persistent Dataset, Linguistic Object
theatre	Type
Theatre	Type
Paphos Theatre	Persistent Dataset, Linguistic Object
fasti.sea_cd.AIAC_4915	Persistent Dataset, Linguistic Object
fasti.sea_cd.AIAC_4888	Persistent Dataset, Linguistic Object
10003888	Persistent Dataset, Linguistic Object
Political theatre	Type
fasti.sea_cd.AIAC_2772	Persistent Dataset, Linguistic Object
Greek Theatre and Theatrology	Type

« 1 2 3 4 »

Figure 25. Search in PD for “theatre”

It seems that “Paphos Theatre” could be an attractive candidate for further exploration:


Paphos Theatre

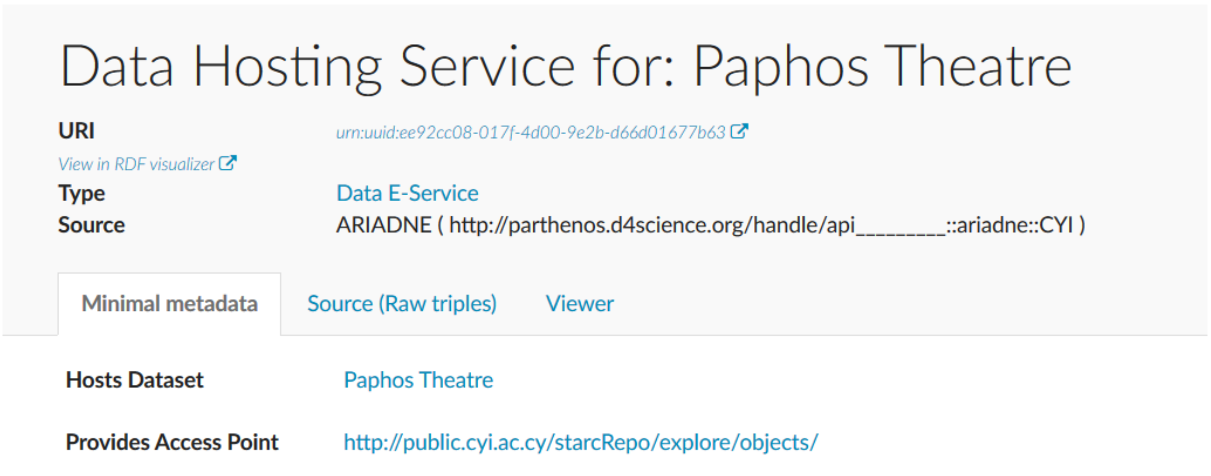
URI	http://parthenos.d4science.org/handle/Ariadne/AriadnePortal/qe74kgq87q79
View in RDF visualizer	
Type	Persistent Dataset, Linguistic Object
Description	Collection of digital resources of the Paphos Theatre
Source	ARIADNE (http://parthenos.d4science.org/handle/api________::ariadne::CYI)

Minimal metadata | Source (Raw triples) | Viewer

ID	starc_dataset:5
Type	Sites and monuments databases or inventories , Fieldwork archives
Dataset hosted by	Data Hosting Service for: Paphos Theatre
Subject	archaeology, theater, building , buildings
Temporal Coverage	300-0-0 until 365-0-0
Right	http://www.europeana.eu/rights/rr-f/ , Ownership Right Of starc_dataset:5 , Rights Reserved - Free Access
Is About	300-0-0 until 365-0-0, archaeology, theater, building, buildings

Figure 26. Detail view of the resource "Paphos Theatre"

The detail view provides more information about the Paphos Theatre as was provided by ARIADNE. Especially, one might be interested where the resource is hosted. Field "Dataset hosted by" leads to the "Data Hosting Service for: Paphos Theatre" with the corresponding Access Point (URL):


Data Hosting Service for: Paphos Theatre

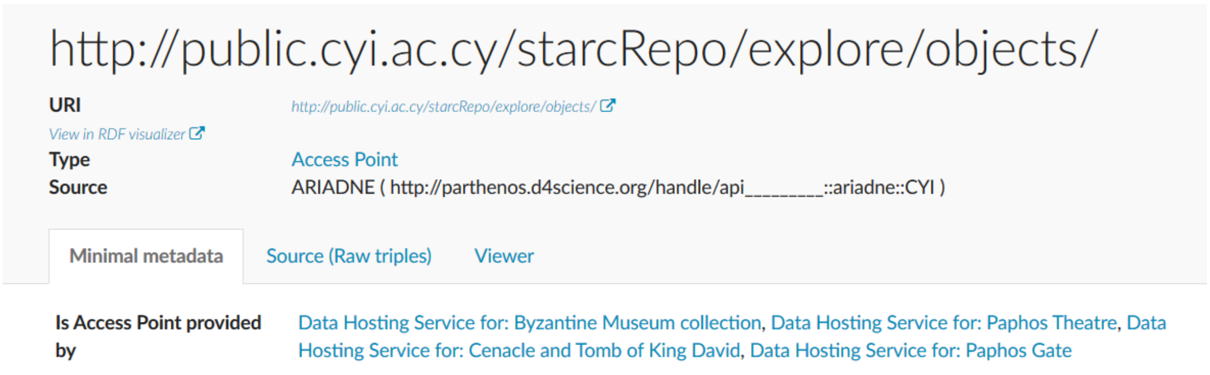
URI	urn:uuid:ee92cc08-017f-4d00-9e2b-d66d01677b63
View in RDF visualizer	
Type	Data E-Service
Source	ARIADNE (http://parthenos.d4science.org/handle/api________::ariadne::CYI)

Minimal metadata | Source (Raw triples) | Viewer

Hosts Dataset	Paphos Theatre
Provides Access Point	http://public.cyi.ac.cy/starcRepo/explore/objects/

Figure 27. Detail view of the Data hosting Service for Paphos Theatre

The user might follow the external link to get to the original location of the data. Alternatively, the user might be interested in other data sets available under given access point. Following the link within PD reveals that the Byzantine Museum collection, the Tomb of King David etc. are also available under given access point.


http://public.cyi.ac.cy/starcRepo/explore/objects/

URI	http://public.cyi.ac.cy/starcRepo/explore/objects/
View in RDF visualizer	
Type	Access Point
Source	ARIADNE (http://parthenos.d4science.org/handle/api________::ariadne::CYI)

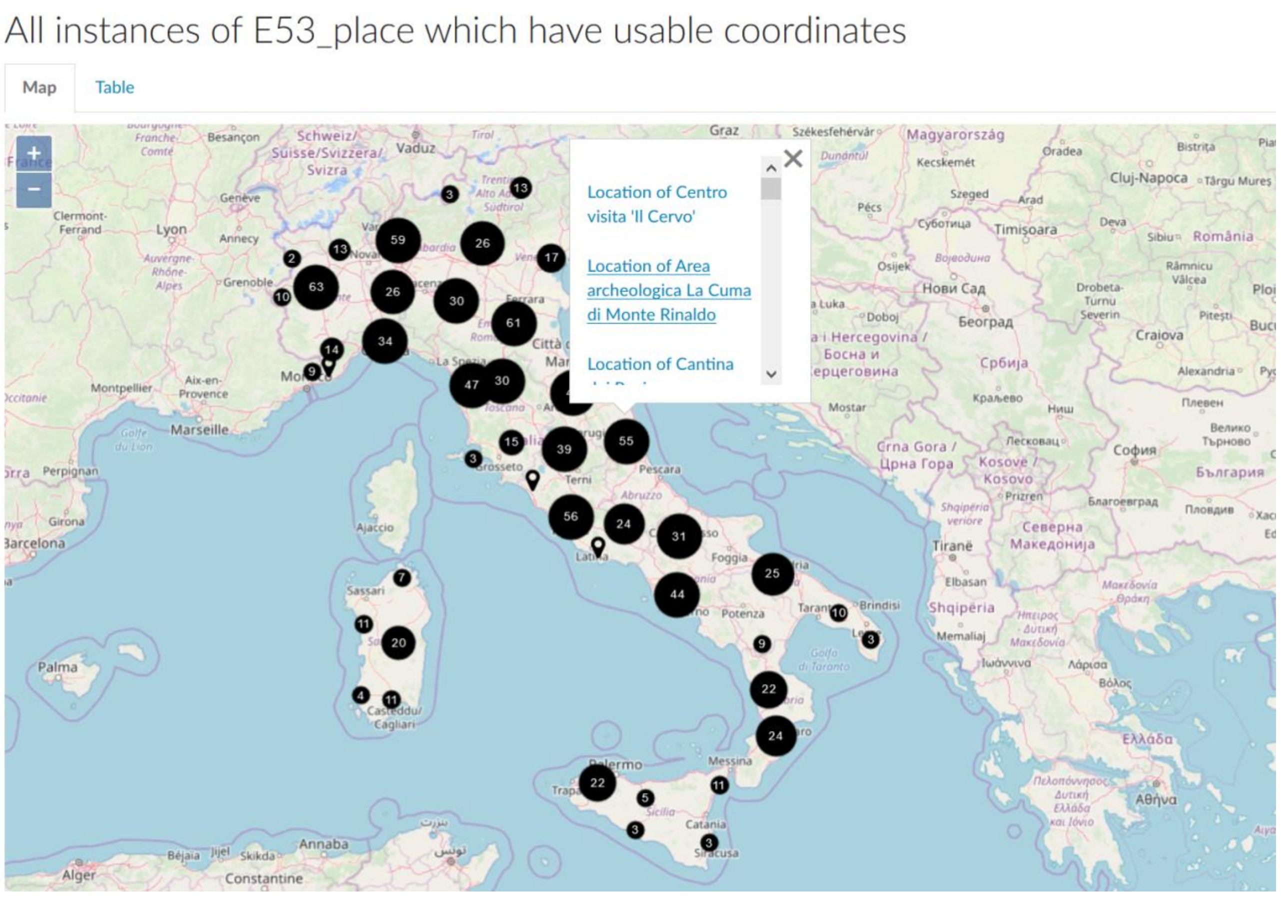
Is Access Point provided by	Data Hosting Service for: Byzantine Museum collection, Data Hosting Service for: Paphos Theatre, Data Hosting Service for: Cenacle and Tomb of King David, Data Hosting Service for: Paphos Gate

Figure 28. Detail view of the Access Point offering Data on "Paphos Theatre"

5.4 In-depth analysis and custom views

The previous sections described only the basic setup of PD. The powerful flexible platform lends itself ideally for further in-depth analyses, by creating new pages and templates custom-tailored to the questions/investigations at hand. The most prominent candidate for such further analysis is data curation and data quality assessment. Custom queries and templates can give hints on the need of more elaborate mappings, but can also point to particular data quality issues.

There is equally large potential with respect to exploration of the data. To demonstrate, we introduce an example with geospatial data: Metaphacts offers out-of-the-box a UI-component for displaying georeferenced data. Unfortunately, only a limited set of the Places in the original metadata features geocoordinates information. Figure 29 demonstrates the potential of map view, when/if geographical coordinates exists in the metadata:


All instances of E53_place which have usable coordinates

Location of Centro visita 'Il Cervo'

Location of Area archeologica La Cuma di Monte Rinaldo

Location of Cantina

Figure 29. Plotting of information on a map UI-component in PD

5.5 Integrate PD with D4science

Even though PD can be used as a stand-alone tool, work is ongoing to also integrate it with the D4science platform. The integration includes following distinct aspects:

- **Synchronize dataspace** - making sure that PD operates on the same dataset as Joint Resource Registry. Due to certain technical limitations on the side of Virtuoso, PD uses a separate blazegraph instance run by ACDH-OEAW as underlying triplestore, instead of the default Virtuoso instance run in D4science. Therefore, a procedure has been set up to regularly automatically synchronize the datasets: Every time Virtuoso has been repopulated with new aggregated data, a full data dump is extracted and stored under an agreed upon location. Subsequently this dump is fetched and used to refresh the data in blazegraph.
- **Make PD visual part of the D4science portal** - It is crucial for smooth user experience, to have all the tools available on one spot, the PARTHENOS VRE. In

minimal setup this means, embedding the remote applications (i.e. PD in this case) via iframe in the main portal application.

- **Make PD part of D4science security context** - Most of PD's functionality is available to the anonymous user. However, defined user context (logged-in users) allows for two important functional enhancements:
  a) personalisation - the user could for example store queries for future use, or even adapt certain aspects of the application;
  b) service invocation - the user could call one of the Data Miner algorithms offered in the PARTHENOS VRE with resources found in PD as payload to be processed.
  The work to enable federated login for PD is still ongoing, a few technical discrepancies have been identified and are being worked on in a tri-lateral investigation involving technicians from D4science, Metaphactory and ACDH-OEAW.

5.6. SparqLaborer - Automatic Querying

5.6.1 Requirements

Ingesting vast quantities of heterogeneous data and transforming them inevitably brings along challenges regarding the resulting data quality. In order to mitigate arising errors not only a structural and coordinated effort to control the data is needed, but also supporting tools which help the data experts at hand. Querying the resulting data via individual and disconnected SPARQL queries while also keeping an overview over the query results in an orderly fashion is a demanding and error-prone task. Thus, the requirements for a supporting tool consist of the following:

Base technical requirements:

- Compiling multiple SPARQL queries in one 'query collection' which can be executed together in one batch processing.
- Writing the results of the multiple queries as well as a summary into a target file / directory.
- Being able to be called programmatically and on a periodic basis.

Logical handling requirements:

- Being able to have multiple values injected at once so that multiple queries can be run from a single base query with several inputs defined.
- Providing a method to enable data experts in writing arbitrary code between queries which should be aimed at reusing individual query results in order to programmatically generate new queries.
- Providing a method to enable data experts in implementing their own custom post processing on individual query collections.

Since time and resources are limited and the production of such a tool is only a small side project aimed at improving the overall ingestion process within PARTHENOS, any non-technical and non-logical functionalities are disregarded, foremost being the creation of a graphical user interface. As such a simple console-based application was sufficient and its programming language python was chosen due to its fast prototyping capabilities and a wide range of libraries supporting the fulfilment of the stated requirements.

5.6.2. Implementation

5.6.2.1. Base technical requirements

This section elaborates on the minimal technical requirements outlined before.

- Compiling multiple SPARQL queries in one “query collection”

Next to being able to compile multiple SPARQL queries in one file, SparqLaborer also provides the possibility of organising and documenting such a set of queries by attaching titles and descriptions to individual queries as well as to the overall collection as a whole.

- Writing the results of the multiple queries as well as a summary into a target file / directory

Three options were implemented:

- Writing the results into a local directory

  This includes a summary of the execution which contains the SPARQL queries themselves, number of their results, duration of execution, error messages, etc. This summary is written into an xlsx file while the raw result data of the query results is written with the format specified in the collection file of which there could be: xml, rdf, json, csv, tsv.

- Writing the results into one xlsx format
  While including the summary as described above, this option includes the query results into the xlsx file as well, so that everything is encapsulated in a single xlsx file.
- Writing the results to a google sheet.
  The data written into the google sheet file is the same as the data written to a single xlsx file: summary and the query results. Sharing access of the google sheet is set to be editable by everyone by default.
- Being able to be called programmatically and on a periodic basis
  Since SparqLaborer was designed from the beginning on a minimalistic basis in respect to the perspective of the user interaction, the resulting mode of interaction is console based. Thus, calling SparqLaborer from any script, manually or automatically, is trivial. An instance of SparqLaborer was set up effortlessly on the ACDH servers which runs a set of query collections on a periodic basis, to enable the tracking of changes in the data quality over time on a consistent basis.

5.6.2.2. Logical handling requirements

In order to enable reuse of queries and avoiding redundant creation of several highly similar queries, a bit of meta logic was added to SparqLaborer, as well as a few software hooks where data experts using SparqLaborer could implement some additional logic on their own.

- Being able to have multiple values injected at once
  Often when analysing data from a certain perspective or research question, the queries to visualize results to these questions are often similar to each since their topic is similar. Hence, in order to prevent redundant workloads and errors due to code duplication, the functionality to inject arbitrary values into queries was needed. This way SparqLaborer can be used to run multiple queries from a single base query but expanded with the help of value injections.
- Providing a method to enable data experts in writing arbitrary code between queries
  As another layer of meta logic, sometimes programmatic work flows are required in between the execution of individual queries. This possibility was

implemented by attaching custom meta functions to each individual query which are defined by the data experts and are called after their corresponding query was executed. This enables for example to feed in results from one query into another, fulfilling a more automated and complexity-driven analysis need.

- Providing a method to enable data experts in implementing their own custom pre/post processing

In some cases, it was necessary to run arbitrary code before the queries are executed, e.g. when their values need to be read from somewhere else automatically. This can be done by implementing code in the designated custom pre-processing hook. Additionally to pre-processing, there also exists the option of running post-processing code, e.g. for custom statistical evaluation. This possibility was implemented by creating a hook where data experts could write arbitrary custom post processing code reusing the query results in any way.

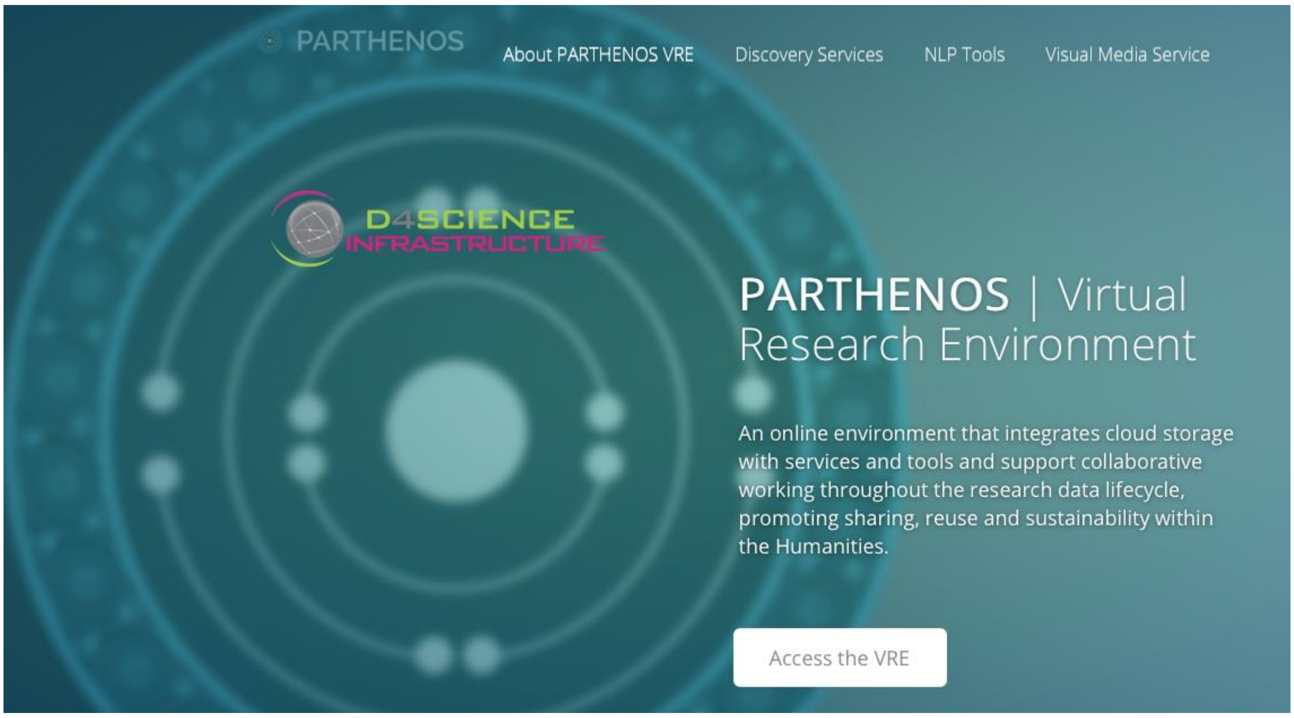
5.6.3. Conclusion

The development of SparqLaborer was done to meet the requirements ad-hoc, resulting from the need for a more systematic and automated way to explore the PARTHENOS data space. As such the emphasis was put on pure functionality and extendibility to save time and to avoid draining resources from the overall PARTHENOS project, while giving the data experts a quick possibility to analyse the conglomerated data in a more structured and automatic way.

The code can be found on: https://github.com/acdh-oeaw/SparqLaborer

6. Findability and accessibility of services

In order to implement findability of services and data, a plan to make the PARTHENOS Community provided services entered in the PARTHENOS Joint Resource Registry has been discussed and is in progress at the time of writing of this document. To ease and implement accessibility of services instead, the creation of a new specific VRE named "The PARTHENOS VRE" is in progress. The plan for this VRE is to have an introductory page and a set of dedicated pages per each service offering typology, namely:



PARTHENOS | Virtual Research Environment

An online environment that integrates cloud storage with services and tools and support collaborative working throughout the research data lifecycle, promoting sharing, reuse and sustainability within the Humanities.

Figure 30. The PARTHENOS VRE home page

- Discovery;
- Text and NLP Tools;
- Visual Media.

The Discovery services will make accessible to users the aggregated RIs metadata compliant with the PARTHENOS Entity Model (PEM), providing faceted and advanced search capabilities as well as the possibility to browse search. The Text and NLP services will make accessible and provide users with a rich set of Natural Language Processing services (e.g. Part-of-speech tagging, Parsers, NER, Opinion Mining etc.) as well as services for reference resources integration/merging.

The Visual Media service will make accessible to users an easy-to-use way for publishing advanced multimedia content on the Web, to upload visual media files on a server and to automatically transform them into an efficient web format, making them ready for web-based visualisation.

7. Use cases

In order to test, and prototypically synthesize the various technical aspects of the PARTHENOS endeavour - resource aggregation & discovery, metadata mapping, integration of processing and visualisation services integration - a number of use cases have been developed and implemented, namely the Reference Resource Integration and Language Resource Switchboard. In the following we provide the details about the selected use cases along with their descriptions.

7.1 Reference Resources Integration

This use case describes the need of a tool to merge specialised reference resources, developed according to specific research purposes, without performing repetitive tasks on each resource. The main actor of this use case is the RUBRICA Service [8], which aims to foster the interoperability and integration of various reference resources used in different disciplines. Starting from trusted knowledge bases (i.e.: databases, thesauri, authority lists etc.) researchers can create, merge, edit and reuse specialized reference resources, developed according to specific research purposes, without performing repetitive tasks on each resource.

Actors:

- RUBRICA Service
- Member (Registered User of the Infrastructure)
- Joint Resource Registry (JRR) (the Registry containing the aggregated RI metadata according to the PARTHENOS Entity Model (PEM))
- Workspace (Infrastructure integrated service for cloud storage)

Triggers:

- The user indicates that he/she wants to merge a number of reference resources found in the JRR.
- Preconditions:
- The JRR contains at least two reference resources.
- The Workspace contains at least two reference resources files that the user previously stored.

Post-conditions:

- The result of the merging operation is saved in the user’s Workspace area.

Basic Flow:

- The user will indicate that he/she wants to merge a number of reference resources found in the *JRR.*
- The *Workspace* will present the reference resources files that the user previously stored.
- The user will indicate that he/she wants to merge two references resource among the ones previously stored in the *Workspace*.
- The user will indicate the parameters necessary to the merging operation.
- The user will confirm that the parameters information is accurate.
- The user will indicate that the merge operation should be executed.
- The *RUBRICA Service* will confirm that the merge operation is executed.
- The *RUBRICA Service* will confirm that the result of the operation is saved in the *Workspace*.
- The user will exit the system.

7.2 Language Resource Switchboard

This use case describes the need for a tool that allows users to easily process PARTHENOS Infrastructure specific types of resources by means of services beyond those that are integrated in PARTHENOS. The main actor of this use case is the Language Resource Switchboard Service (cf. Section 3), which can broker the invocation of applicable services on specific types of resources. It allows users to select services from its large database of CLARIN language technology services, beyond those that are integrated in PARTHENOS.

Actors:

- Language Resource Switchboard Service
- Member (Registered User of the Infrastructure)
- Joint Resource Registry (JRR) (the Registry containing the aggregated RI metadata according to the PARTHENOS Entity Model (PEM))
- Workspace (Infrastructure integrated service for cloud storage)

Triggers:

- The user indicates that he/she wants to process a textual resource found through the JRR.

Preconditions:

- The JRR metadata contains pointers to textual resources.
- The Workspace contains at least one textual resource file that the user previously stored.

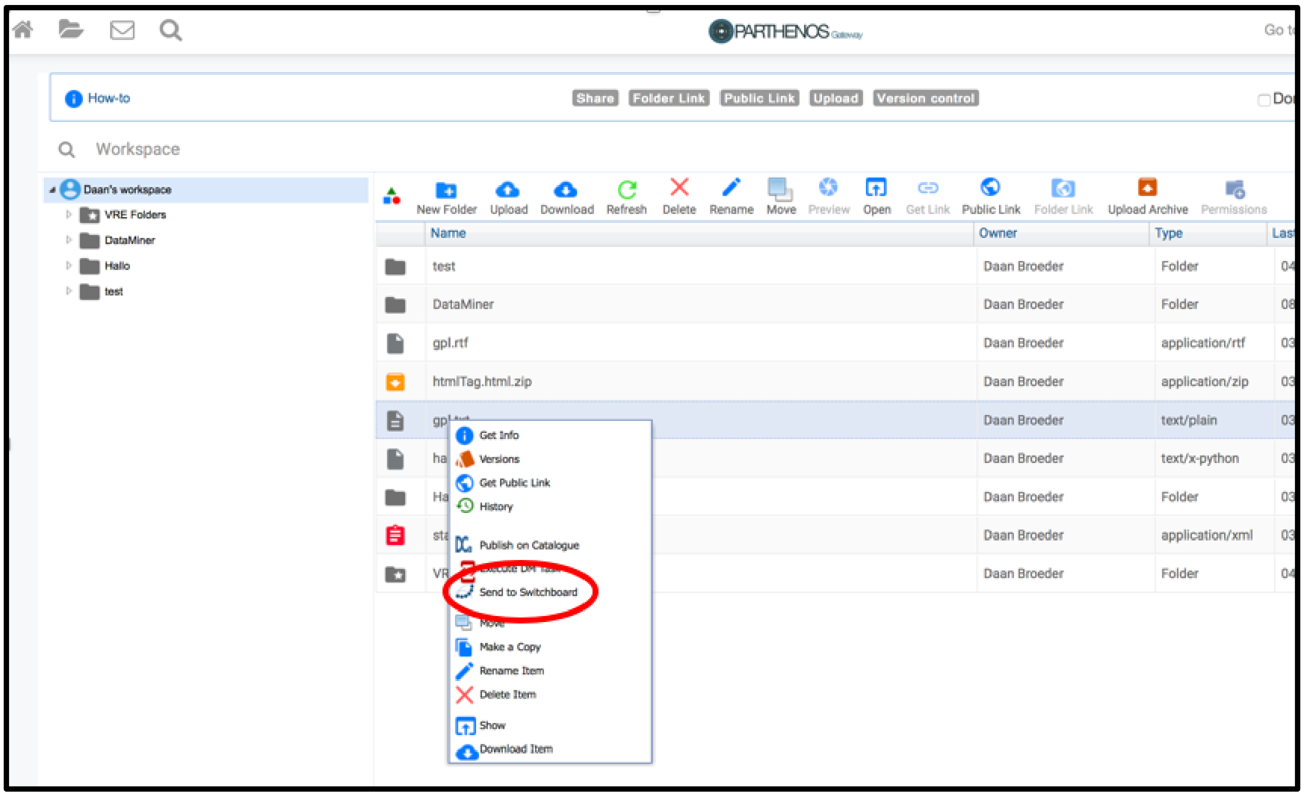
Post-conditions:

- Language Resource Switchboard Service is capable of executing the textual resource coming from the PARTHENOS Infrastructure.

Basic Flow:

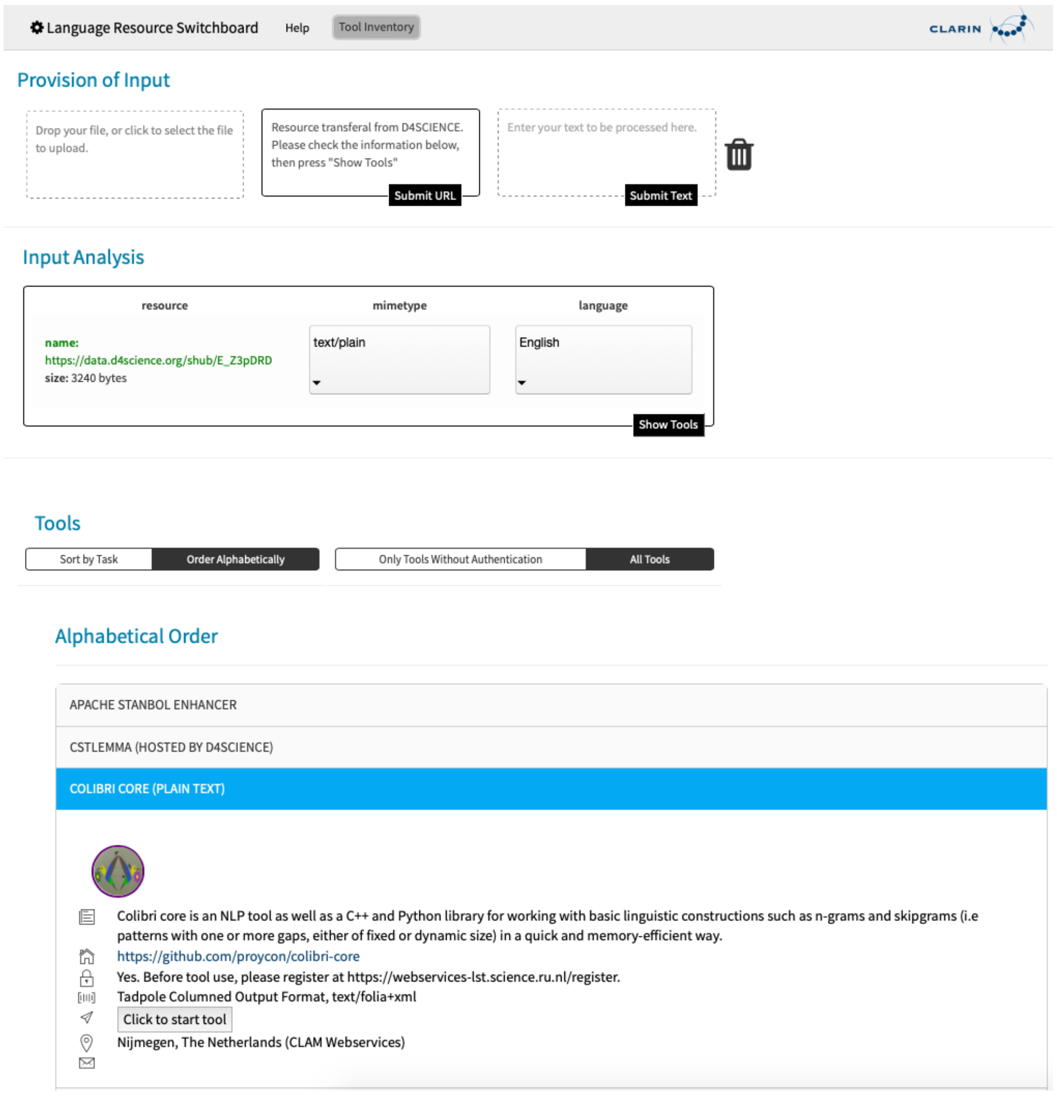
- The user will indicate that he/she wants to process a textual resource found through the JRR.
- The *Workspace* will present the textual resource files that the user previously stored.
- The user will indicate that he/she wants to process one textual file resource among the ones previously stored in the *Workspace.*
- The user will indicate that the processing operation should be executed.
- The *Language Resource Switchboard* will confirm that the operation is executed.
- The user will exit the system.

Figure 31 shows an example where a textual resource from the workspace is selected and sent to the Switchboard with just one single click.



Name	Owner	Type	Last
test	Daan Broeder	Folder	04
DataMiner	Daan Broeder	Folder	08
gpl.rtf	Daan Broeder	application/rtf	03
htmlTag.html.zip	Daan Broeder	application/zip	03
gpl.txt	Daan Broeder	text/plain	03
ha	Daan Broeder	text/x-python	03
Ha	Daan Broeder	Folder	03
sta	Daan Broeder	application/xml	03
VR	Daan Broeder	Folder	04

Figure 31. From PARTHENOS to Language Resource Switchboard - Workspace textual resource selection


Language Resource Switchboard Help Tool Inventory CLARIN

Provision of Input

Drop your file, or click to select the file to upload.

Resource transferal from D4SCIENCE. Please check the information below, then press "Show Tools"

Submit URL

Enter your text to be processed here.

Submit Text

Input Analysis

resource	mimetype	language
name: https://data.d4science.org/shub/E_Z3pDRD size: 3240 bytes	text/plain	English

Show Tools

Tools

Sort by Task | Order Alphabetically | Only Tools Without Authentication | All Tools

Alphabetical Order

APACHE STANBOL ENHANCER

CSTLEMMA (HOSTED BY D4SCIENCE)

COLIBRI CORE (PLAIN TEXT)

Colibri core is an NLP tool as well as a C++ and Python library for working with basic linguistic constructions such as n-grams and skipgrams (i.e patterns with one or more gaps, either of fixed or dynamic size) in a quick and memory-efficient way.

https://github.com/proycon/colibri-core

Yes. Before tool use, please register at https://webservices-lst.science.ru.nl/register.

Tadpole Columned Output Format, text/folia+xml

Click to start tool

Nijmegen, The Netherlands (CLAM Webservices)

Figure 32. From PARTHENOS to Language Resource Switchboard - Textual resource selection loaded automatically in the Language Resource Switchboard

Figure 32 shows the textual resource selected from the workspace loaded automatically by the Language Resource Switchboard which, in turn, allows the end user to select from a broad list of services external to the PARTHENOS Infrastructure (COLIBRI CORE service in the example).

8. References

[1] Pagano, P. Candela, L., Assante, M., Frosini, L., Manghi, P., Bardi, A. & Sinibaldi, F. (2016). “PARTHENOS Cloud Infrastructure”. Deliverable D6.1.

[2] Manghi, P., Artini, M, Atzori, C., Bardi, A., Mannocci, A., La Bruzzo, S., Candela, L., Castelli, D. & Pagano, P. (2014). "The D-NET software toolkit: A framework for the realization, maintenance, and operation of aggregative infrastructures", Program, Vol. 48 Issue: 4, pp.322-354, doi:10.1108/PROG-08-2013-0045

[3] King, M., Ostojic, D., Ďurčo, M., & Sugimoto, G. (2016). Variability of the Facet Values in the VLO – a Case for Metadata Curation. In Selected Papers from the CLARIN Annual Conference 2015, pp. 25–44. Linköping University Electronic Press. Retrieved from http://www.ep.liu.se/ecp/123/003/ecp15123003.pdf

[4] Ostojic, D., Sugimoto, G., & Ďurčo, M. (2016). Curation module in action-preliminary findings on VLO metadata quality. Retrieved from https://www.clarin.eu/sites/default/files/ostojic-etal-CLARIN2016_paper_22.pdf

[5] Shneiderman, B. (1996). The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In Proceedings of the IEEE Symposium on Visual Languages, pp. 336-343, Washington: IEEE Computer Society Press. Retrieved from: http://citeseer.ist.psu.edu/409647.html

[6] Marketakis, Y., Minadakis, N., Kondylakis, H. et al. (2016). X3ML mapping framework for information integration in cultural heritage and beyond. Int J Digit Libr (pp 1-19). doi:10.1007/s00799-016-0179-1

[7] Drude, S., Di Giorgio, S., Ronzino, P., Links, P., Degl’Innocenti, E., Oltersdorf, J. & Stiller, J. (2016). “Report on User Requirements”. Deliverable D2.1.

[8] Degl'Innocenti et al. "Report on the Assessment of Interoperability, Services and Tools" (2018). Deliverable D2.4. https://www.parthenos-project.eu/Download/Deliverables/D2.4_Assessment_Interoperability_Services_and_Tools.pdf (last visited 18/02/2019)

[9] Bruseker G. et al. "Report on the Integration of Reference Resources" (2017). Deliverable 5.4. http://data.d4science.org/RWkvOGxoWkVTUjlMZzYydnA2NXFzbGgzUHJZK1BWS3JHbWJQNStIS0N6Yz0 (last visited 25/02/2019)

[10] Tzompanaki K. and Doerr M. (2012) Fundamental Categories and Relationships for querying CIDOC CRM based repositories. Technical Report ICS-FORTH/TR-429, April 2012. http://www.cidoc-crm.org/sites/default/files/TechnicalReport429_April2012.pdf

[11] Bardi A., Bruseker G., Durco M., Kemps-Snijders M., Lorenzini M., Theodoridou M., (2017). "Report on services and tools (interim)". Deliverable 6.2.