DELIVERABLE

Project Acronym: DM2E

Grant Agreement number: ICT-PSP-297274

Project Title: Digitised Manuscripts to Europeana

D2.3 – Final Version of the Interoperability Infrastructure

Revision: Final 1.0

Authors:

Kai Eckert (UMA)
Evelyn Dröge (UBER)
Konstantin Baierer (EL)
Julia Iwanowa (UBER)
Jorge Urzúa (MPIWG)
Klaus Thoden (MPIWG)
Dominique Ritze (UMA)
Nasos Drosopoulos (NTUA)
Arne Stabenau (NTUA)
Timm-Martin Siewert (EL)
Kilian Schmidtner (SBB)

Project co-funded by the European Commission within the ICT Policy Support Programme		
Dissemination Level		
PU	Public	X

Revision history and statement of originality

Revision	Date	Author	Organisation	Description
0.1	22.1.14	Kai Eckert	UMA	Initial Draft Version
0.2		All	---	Collaborative work on the document
0.3	06.02.14	Kai Eckert, Evelyn Dröge	UMA, UBER	Editing, Draft Finalisation
0.4	07.02.14	Kai Eckert	UMA	Export to Word, further editing
0.5	10.02.14	Violeta Trkulja	UBER	Final revision
1.0	14.02.14	Violeta Trkulja	UBER	Approval of Final 1.0

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise.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

Contents

List of Tables

List of Figures

List of Abbreviations

ABO	Austrian Books Online
API	Application Programming Interface
BBAW	Berlin-Brandenburgische Akademie der Wissenschaften (Berlin-Brandenburg Academy of Sciences and Humanities)
CHO	Cultural Heritage Object
EAD	Encoded Archival Description
EDM	Europeana Data Model
EL	Ex Libris Germany
DM2E	Digitised Manuscripts to Europeana
DM2E model	Specialisation of the EDM made by DM2E
DTA	Deutsches Textarchiv
GUI	Graphical User Interface
JAAS	Java Authentication and Authorization Services
MPIWG	Max-Planck-Institut für Wissenschaftsgeschichte (Max Planck Institute for the History of Science)
NTUA	National Technical University of Athens
OAI-PMH	Open Archives Initiative - Protocol for Metadata Harvesting
OWL	Web Ontology Language
ONB	Österreichische Nationalbibliothek (Austrian National Library)
PNX	Normalised Primo XML format
TEI	Text Encoding Initiative
UBER	Humboldt-Universität zu Berlin (Humboldt University Berlin)
UBFFM	Universitätsbibliothek JCS Frankfurt am Main (University library JCS Frankfurt)
UIB	Universitetet i Bergen (University of Bergen)
UMA	University of Mannheim
VM	Virtual Machine
WAB	Wittgenstein Archives at the University of Bergen
WP	Work Package
XSLT	Language for XML-to-XML transformations

Please note that translations of institution names may be unofficial and do only serve a better understanding of the corresponding abbreviation.

1 Role and scope of this deliverable

This deliverable presents the current state of the interoperability infrastructure being developed in WP2. It is a continuation of deliverable D2.1 ("Initial Version of the Interoperability Infrastructure") and D2.2 ("Intermediate Version of the Interoperability Infrastructure").

Since month 6, a prototype of the infrastructure has been developed and presented within the project. Based on the prototype, the actual interoperability infrastructure has been planned and developed, and has been released in an intermediate version in July 2013.

The intermediate version was used to transform and ingest metadata provided by WP1 and to publish the data as Linked Data to be consumed by WP3 (see D3.2 "Prototyping Platform Implemented").

Based on the experiences gained from the data ingestions and in cooperation with the development of the scholarly environment in WP3, we continued the development of the final version.

The final version consists of the DM2E model describing the digitised manuscripts, a workflow ontology describing the transformation of the metadata, as well as the metadata provenance, a workflow execution engine with a graphical user interface (OmNom), the MINT mapping tool to support the graphical generation of mappings, a Silk-based contextualisation service for the generation of links, a Linked Data API that exposes the generated RDF data to the tools developed in WP3, an OAI-PMH service to provide the transformed data as EDM data to Europeana, and finally a search-and-browse interface as main access point for the users.

In this deliverable, we give description and status overview of all components of the infrastructure, as well as its documentation and deployment.

2 Overview

The project Digitised Manuscripts to Europeana (DM2E) has two primary goals:

1. The transformation of various metadata and content formats describing and representing digital cultural heritage objects (CHOs) in the realm of digitised manuscripts from as many providers as possible into the Europeana Data Model (EDM) to get it into Europeana.
2. The stable provision of the data as Linked Data and the creation of tools and services to reuse the data in the Digital Humanities, i.e., to support the so-called "Scholarly Primitives" (Unsworth, 2000). The basis is the possibility to annotate the data, to link the data, and to share the results as new data.

All metadata in DM2E stems from various cultural heritage institutions across Europe and is maintained and described in various formats, among others MARC 21, MAB2, TEI, and METS-MODS.

These formats provide enough degrees of freedom to reflect specific requirements of the providers, hence it has to be expected that for each format and each provider, a different transformation has to be created and maintained. Changes in the original metadata and adaptions of the transformation process lead to new versions of the published data, for which the provenance has to be stored and provided.

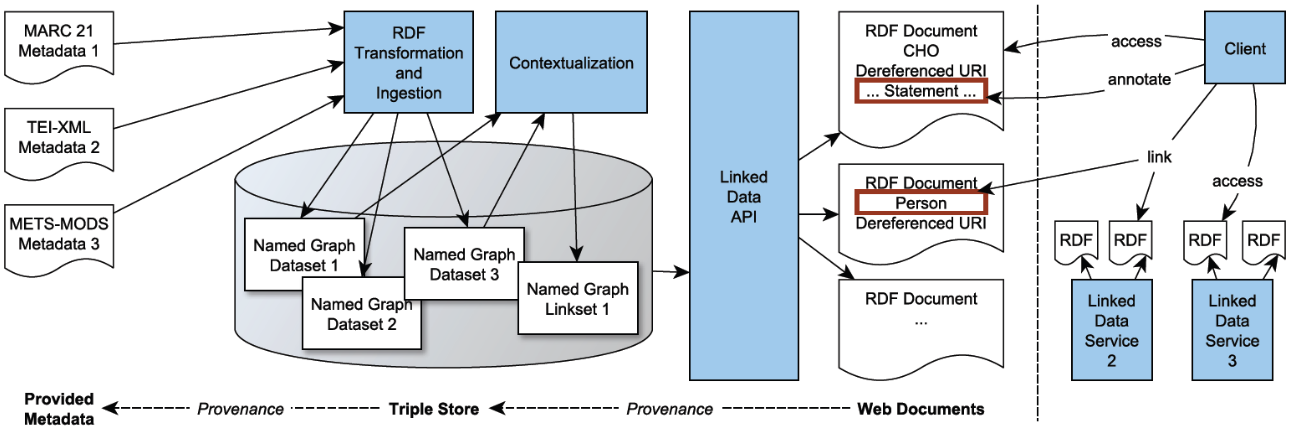


Figure 1: Architecture of the Interoperability Platform.

Figure 1 shows the general architecture of the interoperability infrastructure developed in DM2E. Different transformation and ingestion workflows are used to transform metadata from various input sources to RDF and to ingest the RDF into a triple store. Within the triple store, the RDF data is organised using Named Graphs (Carroll, Bizer et al., 2005), whereby a Named Graph is created for each ingestion. The data is accessed via a Linked Data API where all URIs used in the data are dereferenced and subsets of the RDF data describing the requested resource are returned as single Web documents. For each Web document, provenance information must be provided so that it can be related to a specific ingestion in the triple store and subsequently to the originally provided metadata. In Section 3, we describe our Web-based workflow engine that ensures that all our data can be traced back to the generating workflow and to original data. The Linked Data API exposing our data to the scholarly environment developed in WP3 is described in Section 4.

Recent work on the DM2E model and pointers to its documentation on the Web can be found in Section 5, followed by a Section exemplifying the creation of mappings from original metadata to our DM2E model, using both MINT and XSLT.

In Section 7, the contextualisation is described, i.e., the generation of additional links and statements based on the ingested data using Silk. The contextualisation is shown in Figure 1 as an additional process, leading to linksets that are kept separately from the ingested datasets. This ensures that the user can decide which linksets should be used when working with the data, as different approaches with varying qualities will be used for contextualisation.

Section 8 describes the export of DM2E data to Europeana by means of an OAI-PMH interface and a simplifying mapping of the DM2E data to plain EDM data.

Originally, Europeana was seen as the main entry point to the DM2E data. In the meantime, it turned out that this is not sufficient. For one, a constant ingestion of DM2E data to Europeana already during the project is not feasible - especially as the data still constantly changes as a result of adjustments in the original data, the data model, new requirements by the scholarly users or simply new insights gained during the ingestions. Secondly, our scholars need a dedicated entry point to find all annotatable resources provided via DM2E. Therefore, we decided to provide a search and browse interface, as described in Section 9.

Finally, the status of the documentation of the whole infrastructure, as well as details to its deployment status are provide in Section 10 and 11, respectively.

3 Web-based Workflow Engine

In DM2E, we decided to implement a Web-based workflow system and explore the possibilities, but also the challenges arising from such an approach that combines the Linked Data principles (Berners-Lee, 2006) with a RESTful Web-architecture (Fielding, 2000). The idea is to make not only data available on the Web, but also the processes used to create and transform data, the configuration parameters used within these processes, as well as the original data and possible intermediate results within a longer workflow chain.

In this Section, we describe the framework that we created in order to support our idea of such an open workflow system. As we want to follow the Linked Data as well as the REST principles, we have to define two parts accordingly: on the one hand a formal ontology, the OmNom vocabulary,[1] that describes and relates all resources involved and that can be used to create the Linked Data representations of all resources. On the other hand, the RESTful API has to be specified that puts the ontology into action and allows the execution of workflows on the Web.

3.1 Workflow Ontology

Many ontologies exist to describe Web services, workflows, data, and provenance. In DM2E, we created another one that liberally reuses ideas that have been formalised before -- e.g., that a Web service has some inputs and outputs, that a workflow consists of several connected services, and so on. We do not reuse an ontology here as no existing ontology completely provides what we need and at the same time too many adapted ontologies complicate the matter unnecessarily at this point in time, where we want to present and shape the idea of a consistent open workflow ontology. Mappings to existing and actually applied ontologies are of course crucial to simplify the reuse of existing resources and certainly part of our future work. An exception to this is the W3C PROV ontology which we use directly as a basis for our work.

A workflow is the composition of several steps to generate some output by processing some input. In the context of the Web, the single processing steps are Web services. From the workflow perspective, a Web service is seen as an atomic activity whose internals are hidden from the workflow system - and the rest of the world.[2]

Based on these preliminary considerations, we can structure the whole workflow ontology into three different areas:

1. **Specification**: First, the workflow components - the Web services - have to be specified in order to be used within a workflow. A Web service specification mainly consists of the definition of the input parameters and the output.
2. **Composition**: The composition relates the single Web services within a workflow. Therefore, Web services have to be assigned to positions within the workflow and outputs have to be connected to inputs of subsequent services.
3. **Execution**: When a workflow is executed, further resources are needed, for instance the job that represents the execution of a Web service, the actual parameter values and custom configurations that have been applied.

[1] OmNom vocabulary: http://onto.dm2e.eu/omnom/ (04.02.2014).

[2] This implies that transparency can only be reached up to a certain granularity. The granularity of Web services is deliberately chosen to be consistent with the WWW architecture where internals of a server are generally not visible.

Notation: We use a slightly extended version of PROV-N (Moreau & Missier, 2013) for the notation of the examples. `WebService(w1)` defines `w1` as an instance of the class `Web Service`. Relations between resources can be specified explicitly, e.g., `wasGeneratedBy (e1,j1)` indicates that entity `e1` was generated by job `j1`. Alternatively, they are provided as attribute-value pair during instantiation: `entity(e1,[wasGeneratedBy='j1'])`. In the attribute-value pairs, single quotes (`'`) denote URI references, double quotes (") literals. As an extension to PROV-N, we allow multiple comma-separated values for one attribute, e.g., `WebService (w1,[inputParam={'in1','in2'}])` means that Web service `w1` has two input parameters `in1` and `in2`.

A simple workflow: Figure 2 shows a simple example workflow. The Wittgenstein Archives at the University of Bergen (UIB) provide a TEI-XML file containing metadata and transcription of a digitised manuscript by Ludwig Wittgenstein (MS 114: X. Philosophische Grammatik). Additionally, an XSLT mapping has been created that translates the TEI data into the DM2E model, more specifically, TEI-XML into RDF-XML. The first stage in the workflow is the XSLT processor that uses the mapping and the original data to create the intermediate RDF representation (Ms-114-RDF-001, a stable, versioned resource). This intermediate representation is then consumed in the next stage by the Ingestion service, where additional metadata and links to prior versions are added. The result is stored in the central triple store, identified by a stable URI indicating this specific ingestion (Ms-114-2013-10-03).

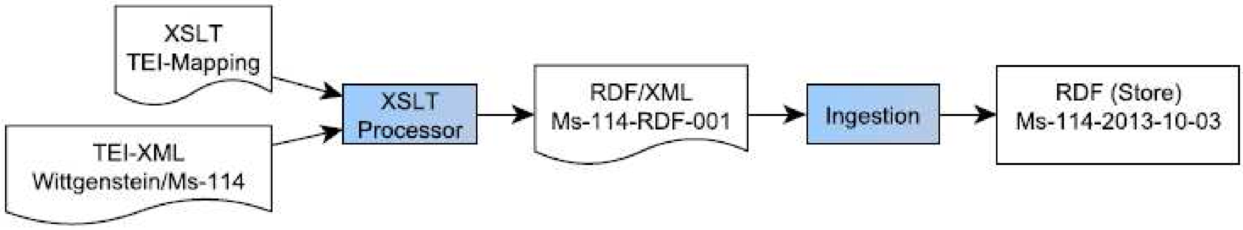


Figure 2: Example DM2E Workflow.

3.1.1 Specification

Regarding the specification of the Web services, the ontology is straightforward; each Web service is mainly characterised by its input and output parameters. In the following, we use the example Web service as described above. Its specification looks like this:

```
WebService(w1, [label="XSLT Processor", inputParam={'xml','xslt'},
outputParam='out'])
Parameter(xml, [ofWebService='w1', label="XML", comment="XML data",
isRequired="true", parameterType='xmlType'])
Parameter(xslt, [ofWebService='w1', label="XSLT", comment="XSLT Mapping",
isRequired="true", parameterType='xsltType'])
Parameter(out, [ofWebService='w1', label="Result", comment="Transformed XML
data", parameterType='rdfXmlType'])
```

Figure 3 shows the relation between the WebService class and the Parameter class. General attributes like label and comment are omitted. The parameter type refers to an XSD type (for literal parameter values) or a class from a formalised type ontology that allows inferring the compatibility of types for parameter values. For example, `rdfXmlType` is compatible with both generic types `xmlType` and `rdfType`, i.e., subsequent services that require either XML data (such as an XSLT transformer) or RDF data (such as an entity linking component) can use the output of an `rdfXmlType` parameter. The indication if a parameter is required is important for the configuration of a workflow in the workflow engine and a sensible default value makes configuration easier for the user.

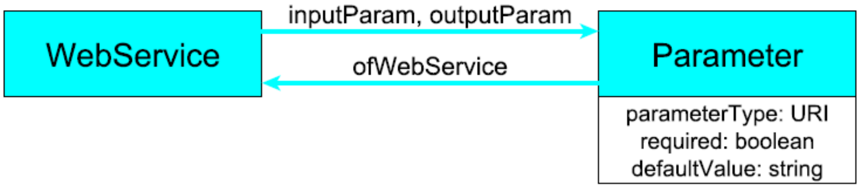


Figure 3: Web Service Specification.

3.1.2 Composition

Within a workflow, multiple Web services can be combined. An example workflow procedure is shown in Figure 4. Similar to Web services, each workflow has input and output parameters. In our example, the input parameters are `param1`, `param2` and the output parameter is `param3`. Within a workflow, each Web service is assigned to a position. The positions are needed to distinguish different invocations of the same Web service if a workflow should invoke a Web service more than once. Connectors (pink arrows in Figure 4) are used to connect resources to input parameters. These resources are typically provided as input parameters to the workflow or - for subsequent Web services in the workflow - are the output of an earlier Web service. For example, the output of `WS1` on `Position1` is the input of `WS2` on `Position2`. The implementation of the workflow engine uses the type information of the parameters to ensure that only compatible parameters get connected.

Although the positioning suggests a linear execution of the Web services, this is not necessarily the case. The parameter connectors define a partial order of the positions such that Web services that do not depend on each other can be executed in parallel. As soon as all input parameters of a Web service are available, it will be executed, which is particularly important for potentially long-running tasks like link discovery or data transformation in bulk.

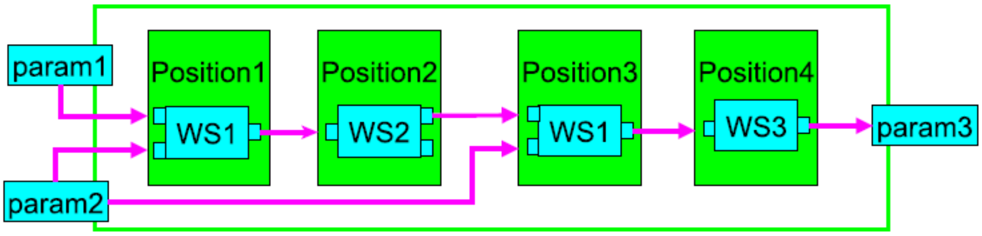


Figure 4: Example of a Workflow Procedure.

Figure 5 provides an overview on the relations between all the classes involved in the composition of Web services as a workflow. As indicated earlier, each workflow has input and output parameters. Additionally, a list of workflow positions is specified with each position pointing to a specific Web service. The ParameterConnector class is used to connect the parameters of all Web services included in the workflow.

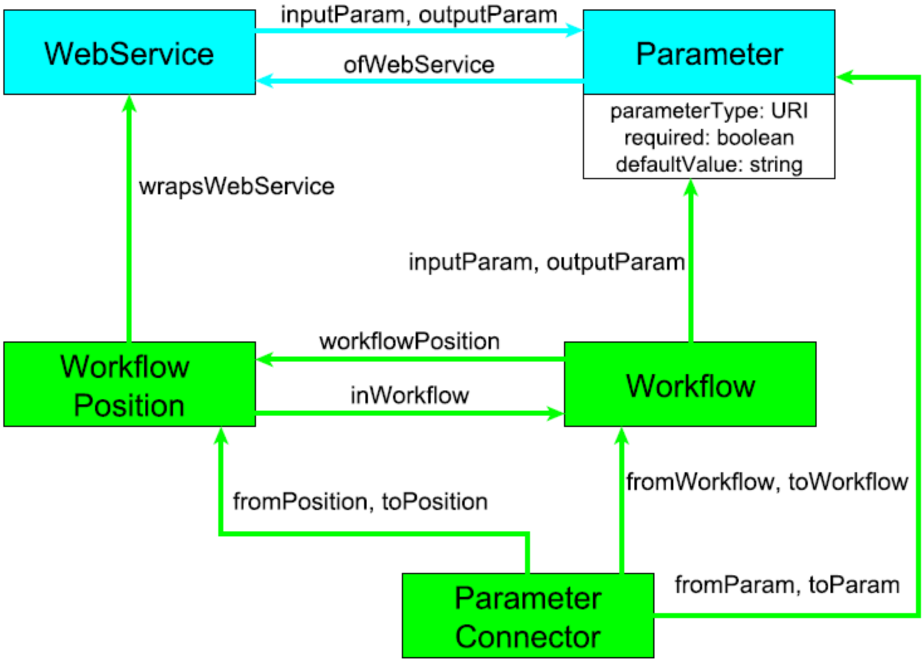


Figure 5: Composition of workflows.

3.1.3 Execution

Until now, the design of our ontology was more or less straightforward and in line with other workflow modeling approaches. Upon execution, however, we have to introduce a special resource that is usually omitted: the configuration. A configuration contains all assignments of values to parameters of a Web service or a workflow, respectively. There are two reasons why we model the configurations as independent resources. For one, this fits to the general idea of a transparent, resource-oriented workflow system. But for transparency, the information could be stored as well in the job descriptions. The main reason, is that we can use a more REST-style communication by posting simply and consistently the URI of a configuration in order to invoke a Web service instead of an RPC-style invocation with service-specific parameters.

Each configuration is persisted and exposed as Linked Data, including all information relevant to reproducing this specific execution: the identifier of the configuration, the Web service or workflow to be invoked, parameter assignments as well as provenance information such as the generation time and the workflow or the human user who generated the configuration. For example:

```
WebServiceConfig(c1, [assignment={'c1a1','c1a2'}, configuresWebService='w1',
modified="2013-09-12T14:40:55.361+02:00", wasGeneratedBy='workflow1'])
ParameterAssignment(c1a1, [forParam='xml', parameterValue='file1'])
ParameterAssignment(c1a2, [forParam='xslt', parameterValue='file2'])
```

The mentioned Web service configuration was used to configure the Web service *w1*. It has been created by *workflow1* and contains two assignments: *file1* was the XML-file which has been used and *file2* the according XSLT.

Finally, whenever a Web service or workflow is invoked, a job resource is created that serves as endpoint for the asynchronous communication, i.e., this resource contains information about the status of the job (NOT_STARTED, STARTED, FINISHED, FAILED). When finished, the job links also to the value assignment of the output parameter. Detailed log messages of the system can be represented as resources linked to the job. Note that the job resources are the only resources that can change their state, only after reaching either FINISHED or FAILED, they remain stable.

In Figure 6, an overview of the ontology is depicted, as presented so far. We only left out some minor classes and properties, that are not needed to understand the general approach.

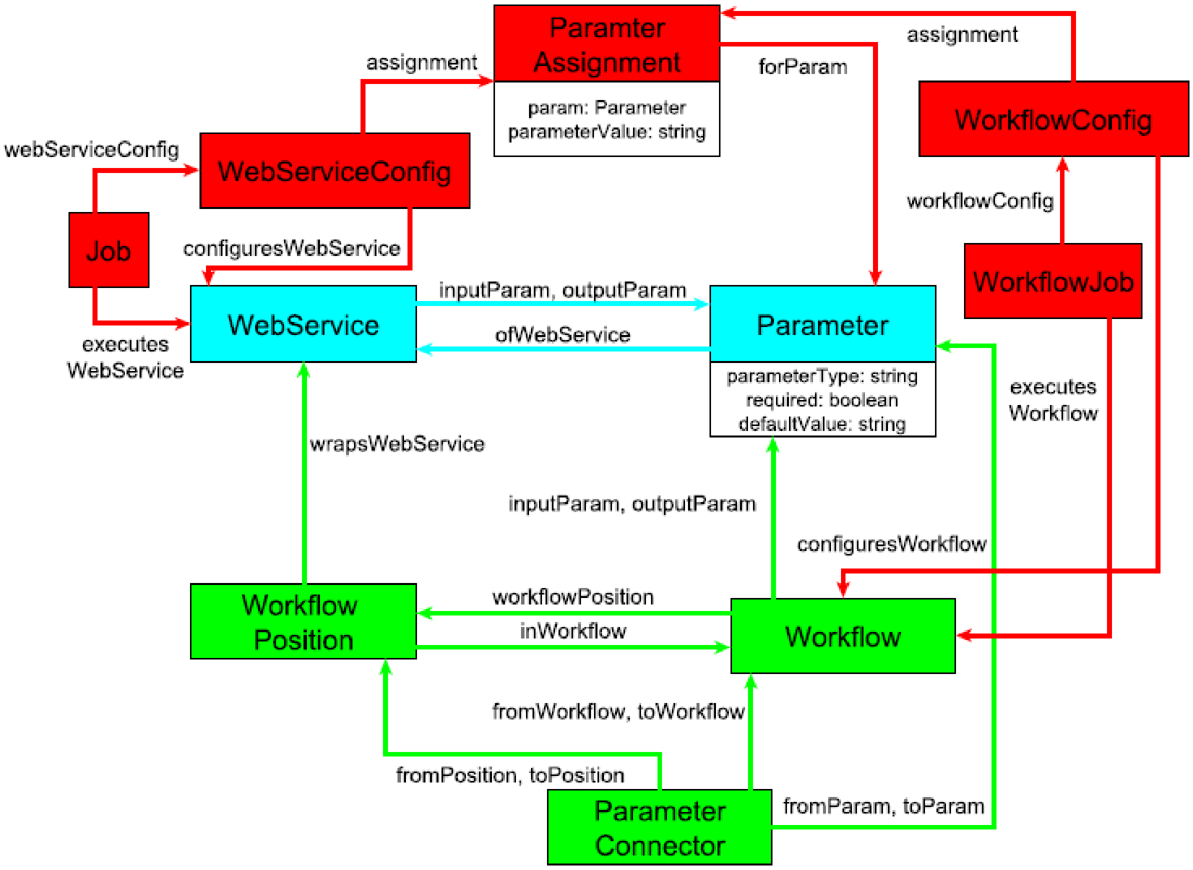


Figure 6: Execution of workflows.

Figure 7 exemplifies the ontology with a full workflow (without jobs and assignments to keep it preferably simple): the Web services are specified, assigned to their positions and the inputs and outputs are connected according to the needs of the workflow. During runtime, actual assignments to the parameters are provided via the configurations (shown in red).

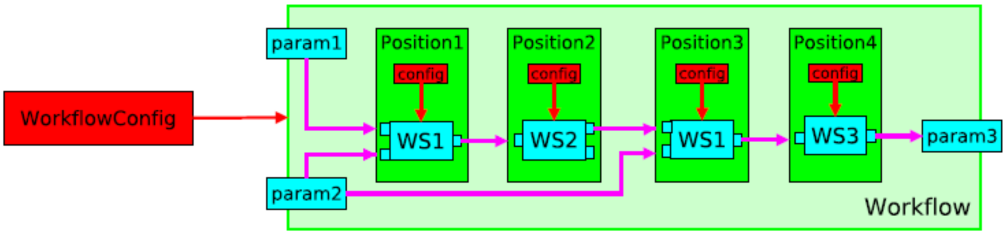


Figure 7: Full workflow with runtime information persisted as Web resources.

Our ontology is designed to represent each and every information and communication artefact that is available about workflows and services, as well as created during the execution. By linking all involved resources, it is possible to trace back all results to the original resources and to reveal all circumstances of its transformation.

3.2 REST API

In the following, we describe the REST API that exposes all resources on the Web and therefore completes our approach of an open workflow. RESTful APIs use resource URIs and allow an interaction with the resources using common HTTP requests, like GET to retrieve a representation of the resource, PUT (with existing URI) or PATCH to change a resource, PUT (with a new URI) or POST to create a new resource and DELETE to delete a resource. This usage of HTTP verbs is in accordance with the basic CRUD operations of a database and therefore especially suitable for data-centric applications.

The basis for our API are the Linked Data principles, i.e., each and every resource that is created according to our ontology can be dereferenced (using GET) to retrieve an RDF representation. This representation contains all available links to other resources according to our ontology, sometimes with additional statements. A configuration, for example, contains not only links to assignments, but also further statements, which values are actually assigned to which parameter, i.e., the full configuration as it is supposed to be consumed by a Web service or a workflow.

As workflows and Web services work very similarly in our system (workflows in fact are also Web services), we refer to them in the following as agents. There is also a third type of agent used in our system: Web services that store and provide resources like configurations or input files. To trigger actions in our system, messages have to be sent to an agent resource.

As every information is represented in resources, it is only necessary to point to such a resource in order to submit it. If a new resource is to be created, its representation is sent via PUT or POST.

The agents follow a combination of “follow your nose” and a few sensible defaults, i.e., careful deviations from hypertext-driven communication to reduce HTTP requests. In this vein, every GET on an agent returns its description, including its input and output parameters. By convention, appending */list* on the root URI of the Web service lists all resources created by this service. This is especially needed for the storage services. In these cases, new data is added by POSTing the RDF description of a blank node that is skolemised[3]

[3] Skolemization is the process of converting blank nodes into URI resources in such a way that their identity is preserved.

and assigned a stable URI in the process. If no RDF resource is to be created, as for a file upload, the actual file is sent in a multi-part request. Here we had to deal with an interesting issue of Linked Data applications: Usually, content-negotiation is used to determine, if a resource or its description is desired. This does not work in the case of RDF resources, as they contain RDF, just like their metadata. Therefore, we decided to interpret the existing accept header for RDF data always as an indicator to retrieve metadata. For the actual files, no accept header or a general one like application/octet-stream has to be set. This does not preclude setting the content type of responses correctly, so consumers know what media type they received. PUTting data to that newly-minted URI updates the data completely, PATCHing data non-destructively adds data to the graph denoted by the URI. If a resource is to be DELETEd, in the spirit of stable URIs and stable data, it is marked as deleted but subsequent requests will return a status of 410 GONE so as to degrade gracefully and prevent erroneous re-use of this URI.

The actual execution of Web services is then three-fold as depicted in Figure 8: First, the client creates a configuration containing parameter assignments, persists them by means of a POST request to the configuration service and receives the URI of this configuration. It then POSTs the configuration URI to the service in question which kicks off the asynchronous execution of the Web service in the background but instantly returns the URI of a job resource. The asynchronous worker updates the job using PUT or PATCH requests to the job URI while the client polls the current RDF description of the job using GET requests.

As jobs serve as communication artifacts between the asynchronous worker handling the execution of the Web service and the client, they provide references to all the resources responsible for its execution, as well as log messages, status updates and, on successful completion of the job, the output parameter assignments.

As said, workflows behave exactly as regular Web services: They require the URI of a persistent workflow configuration with input parameter assignments and return the URI of a workflow job resource upon a POST request. In the background, the workflow service dynamically creates configurations for the Web services according to the configuration of the workflow and the results of previous Web service runs, following the connections in the workflow description.

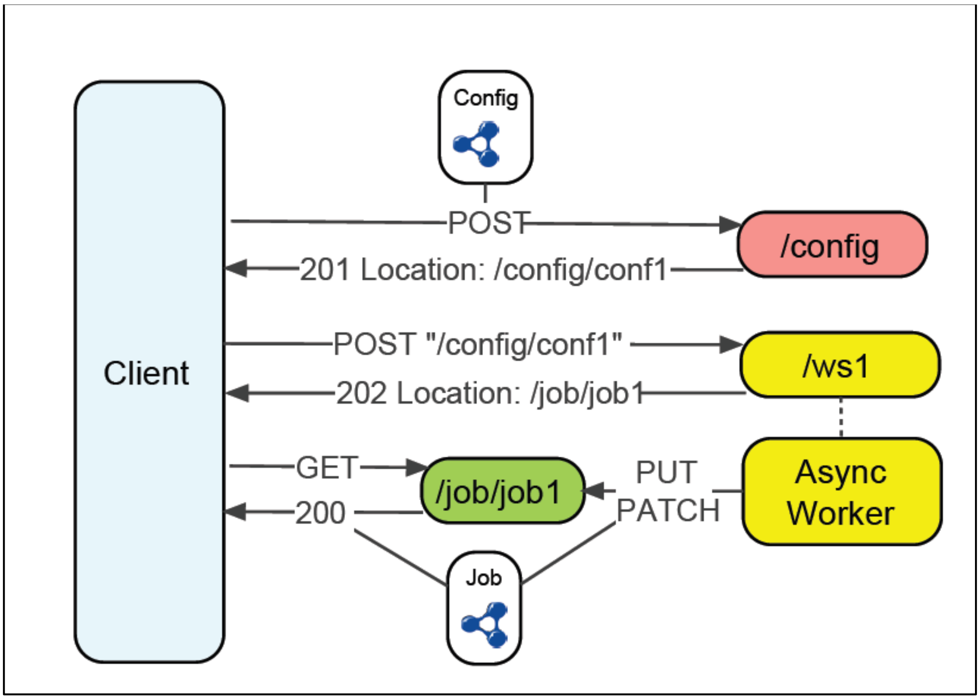


Figure 8: HTTP communication of a Web service execution.

3.3 Implementation

In the following, we give further information about technical details of our implementation. It consists of a lightweight browser-based user interface (called OmNom) and a backend that implements the workflow execution engine.

The bulk of the backend is written in Java,[4] using the Jersey implementation of the JAX-RS standard for RESTful Web services, for inter-service communication as well as with the user interface. Since the workflow execution relies so strongly on the exchange of RDF messages, we developed *Grafeo*,[5] an augmentation of the Jena RDF framework, that adds support, among other features, for exposing Java objects as RDF using Java annotations. Thus, all the classes of the ontology described above have a direct representation as Java classes, ready to be exported to or instantiated from various RDF serialisation formats as well as a JSON format similar to JSON-LD (JSON for Linking Data).[6]

In addition to Web services for creating, updating, retrieving and deleting the various objects that make up the ontology, an asynchronous Web service handles the queuing and running of workflows and orchestrates the correct execution of contained Web services. Task-specific Web services include XSLT transformers, a publishing component that handles the provenance-related aspects of ingestion, a file service offering an easy-to-use and easy-to-integrate storage solution, as well as a service that automates resource lookup and linking using the Silk Link Discovery platform (see Section 7 on contextualisation).

[4] Backend source code: https://github.com/DM2E/dm2e-ws (05.02.2014).
[5] Grafeo source code: https://github.com/DM2E/grafeo (05.02.2014).
[6] JSON-LD: http://json-ld.org (05.02.2014).

The user interface is based on Backbone.js[7] and Bootstrap,[8] allowing users to upload files, create workflows (cf. Figure 9), run jobs and monitor the resulting jobs.[9] Links to the HTML representation of the ingested data (see Section 4 below) for the review of the ingested data, as well as an integration with MINT for the creation of XSLT mappings (see Section 6 on the Creation of Mappings) are provided. The user interface strives for intuitivity and responsiveness, by allowing the use of Drag&Drop for workflow creation and parameter assignments, completely avoiding page reloads and offering both local and global filters to quickly find specific resources.

All user interface-related information is persisted and accessed using the same kind of RESTful RDF-based Web services as used in the workflows, further improving integration. New Web services can be registered in the system. The system reads the published self-description of the service and includes the service into the user interface.

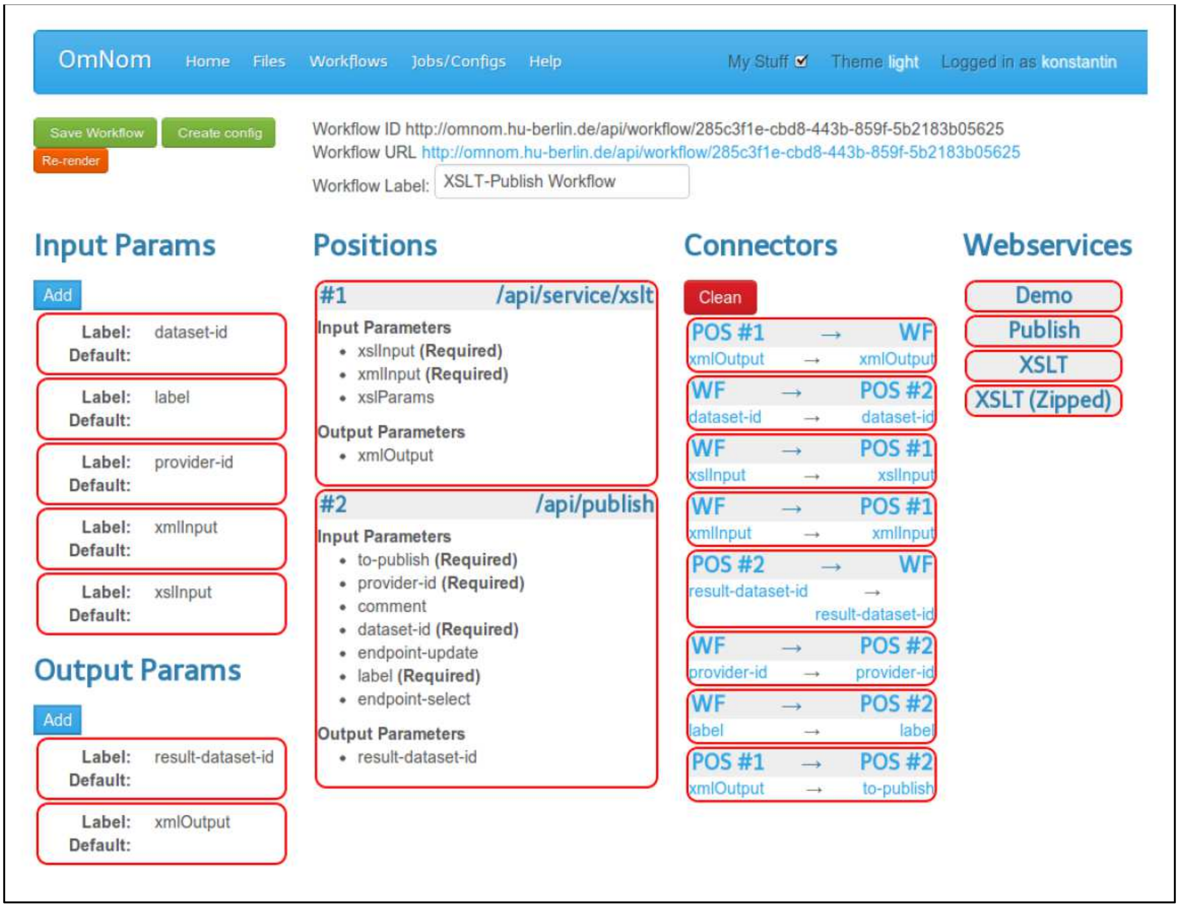
As part of the infrastructure, we also developed a central user management that is used to authenticate users of OmNom, but also of other tools used in DM2E, like MINT and Silk. This security service allows the centralised authentication and authorization to use every tool that belongs to the interoperability infrastructure. This service supports Single Sign-On, it means that the user logs in once and can access all the systems of the interoperability infrastructure.

The security service is based on the Java Open Single Sign-On (JOSSO) framework version 1.8.7[10]. JOSSO is based on several other standards like JAAS, Web Services/SOAP and Realm and also supports useful features like "remember me" and "password reset".

The deployment of this service needs two steps. The first one is the installation of the centralised security service (also called JOSSO gateway), and the second one is the configuration of a partner application (e.g. OmNom).

For further details please refer to the user- and developer documentation provided with this final version of the infrastructure (see Section 10).

[7] Backbone.js website: http://backbonejs.org/ (04.02.2014).
[8] Bootstrap website: http://getbootstrap.com (04.02.2014).
[9] OmNom source code: https://github.com/DM2E/dm2e-gui (04.02.2014).
[10] JOSSO website: http://www.josso.org (05.02.2014). The complete library can be downloaded from: http://sourceforge.net/projects/josso/files/JOSSO/JOSSO-1.8.7/ (05.02.2014).



Workflow ID http://omnom.hu-berlin.de/api/workflow/285c3f1e-cbd8-443b-859f-5b2183b05625
Workflow URL http://omnom.hu-berlin.de/api/workflow/285c3f1e-cbd8-443b-859f-5b2183b05625
Workflow Label: XSLT-Publish Workflow

Input Params

Label: dataset-id
Default:

Label: label
Default:

Label: provider-id
Default:

Label: xmlInput
Default:

Label: xslInput
Default:

Output Params

Label: result-dataset-id
Default:

Label: xmlOutput
Default:

Positions

#1 /api/service/xslt

Input Parameters

- xslInput **(Required)**
- xmlInput **(Required)**
- xslParams

Output Parameters

- xmlOutput

#2 /api/publish

Input Parameters

- to-publish **(Required)**
- provider-id **(Required)**
- comment
- dataset-id **(Required)**
- endpoint-update
- label **(Required)**
- endpoint-select

Output Parameters

- result-dataset-id

Connectors

POS #1 → WF
xmlOutput → xmlOutput

WF → POS #2
dataset-id → dataset-id

WF → POS #1
xslInput → xslInput

WF → POS #1
xmlInput → xmlInput

POS #2 → WF
result-dataset-id → result-dataset-id

WF → POS #2
provider-id → provider-id

WF → POS #2
label → label

POS #1 → POS #2
xmlOutput → to-publish

Webservices

Figure 9: Screenshot of the OmNom user interface.

4 Versioning and Linked Data API

The workflow engine is not DM2E specific and can be used in almost arbitrary contexts. Only few Web services are tailored to the DM2E project, for example the XSLT transformation and the contextualisation service. The most important service from the overall DM2E perspective, however, is the Publish service. This service, which can also be invoked directly, consumes a URI to arbitrary RDF data (in DM2E usually the result of a transformation or contextualisation) and ingests it into the central data store.

Besides the actual data ingestion, the metadata for the ingestion is created that links the newly ingested graph to prior versions (identified by the same provider and dataset IDs) and to the provenance chain provided by the workflow engine. As the full provenance chain is already published by design, only a link to the generating `Job` URI needs to be added.

For each dataset, a stateless (cf. Eckert 2013), general dataset URI is coined. All versions of a dataset are linked to this dataset URI by means of *prov:specializationOf*[11]. Additionally, a link to the prior version is added for convenience. This way, the metadata remains stable as well: the most recent version of a dataset can only be retrieved by dereferencing the general dataset URI where links to all versions are provided.

The ingested data in DM2E is accessed via a RESTful Linked Data API, following the Linked Data principles. Table 1 lists the most important URIs. All URIs, in particular the URIs of RDF data and datasets, represent resources described in the DM2E model (Section 5).

Furthermore, URIs of other resources within the data are dereferenced:
Timespans: .../**timespan**/[provider]/ [collection] /[identifier]
Agents: .../**agent**/[provider]/{[collection]}/[identifier]
Concepts and subject headings: .../**concept**/[provider]/{[collection]}/[identifier]
Places: .../**place**/[provider]/{[collection]}/[identifier]

The URIs are coined during the mapping process. It is possible for the providers to coin provider-specific URIs or collection-specific URIs. For further details, please refer to Goldfarb & Ritze (2013)[12].

The principle for the data delivery is the same for all URIs. The server determines the latest version of a dataset that contains information about the resource and delivers this data together with versioning information. This way, the provided RDF data remains stable as it is always related to a specific, versioned dataset or linkset. This is important to allow statements about the content of the metadata as well.

The content itself (scanned pages of the manuscripts, the textual transcriptions, images, etc.) is hosted by the providers and linked from the metadata. For the hosting of the content, a technical specification has been developed in DM2E (cf. Goldfarb, Morbidoni & Eckert, 2013[13]) to ensure that the content remains stable and is prepared for the consumption by applications like the scholarly environment developed in WP3.

[11] Provenance namespace: http://www.w3.org/ns/prov# (04.02.2014).

[12] The mapping recommendations by Goldfarb & Ritze (2013) are also part of deliverable D1.2.

[13] The annotatable content specification by Goldfarb, Morbidoni & Eckert (2013) is also part of deliverable D1.2.

Figure 10 shows a screenshot of the HTML representation of a resource map accessible via the Linked Data API. The API is implemented using Pubby, with many DM2E specific extensions and improvements.[14]

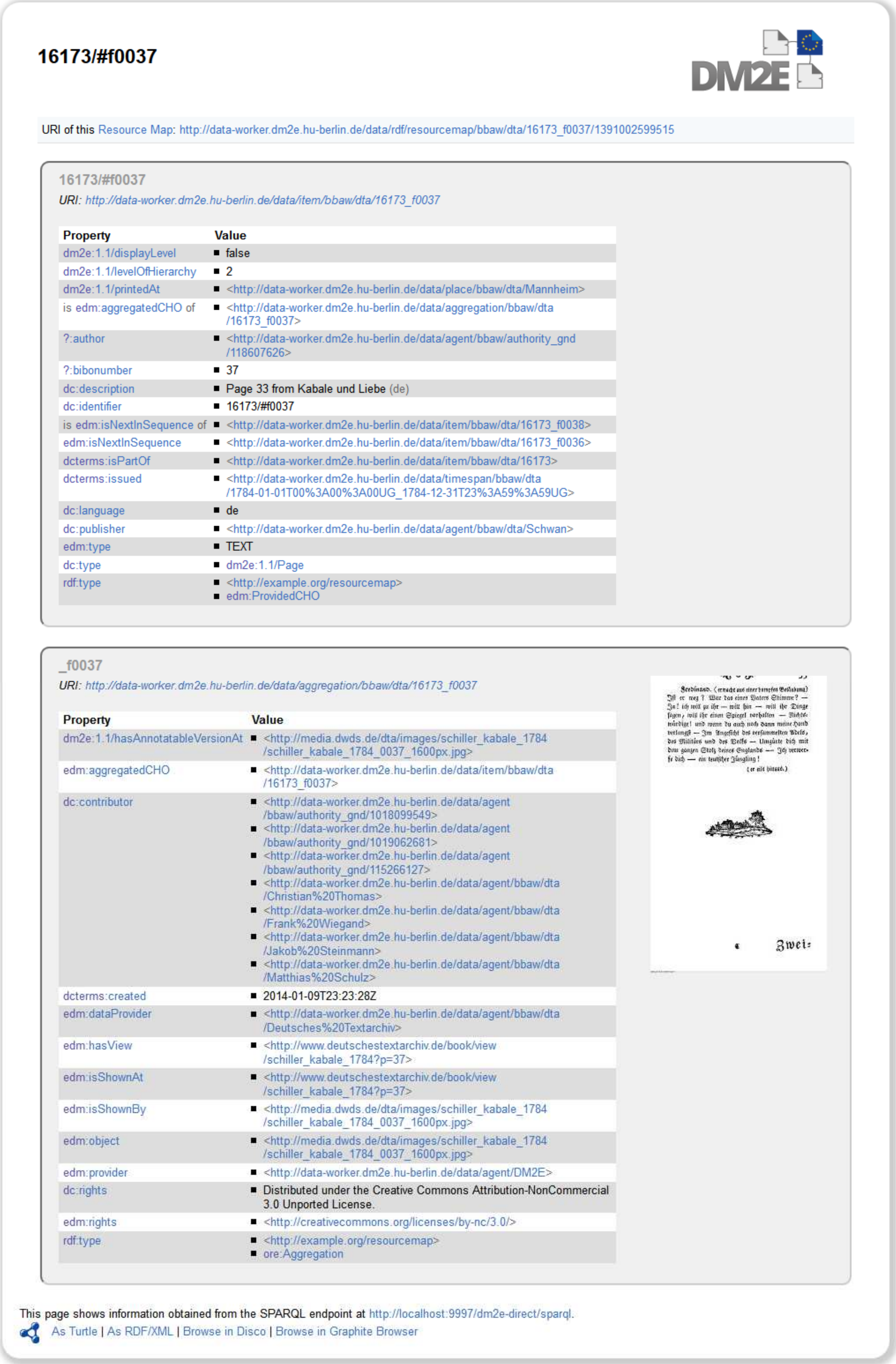
URI (http://data.dm2e.eu/data/...**)**	**Type**	**Content**
dataset/[provider]/[datasetID]/**[version]**	void:Dataset[15]	Description of the ingested dataset
item/[provider]/[datasetId]/[identifier]	edm:ProvidedCHO	303 to latest version of the resource map
aggregation/[provider]/[datasetID]/[identifier]	ore:Aggregation	303 to latest version of the resource map
rdf/resourcemap/[provider]/[datasetID]/[identifier]/**[version]**	ore:ResourceMap	RDF data, linked to a void:Dataset of the same version
linkset/[provider]/[linksetID]/**[version]**	void:Linkset	Description of a linkset
linkset/[provider]/[linksetID]/**[version]**/[provider]/[datasetID]/**resource**/[identifier]	dm2e:DataResource[16]	RDF data, links for a specific resource

Table 1: Linked Data API.

[14] Source Code: https://github.com/DM2E/pubby

[15] Void namespace: http://rdfs.org/ns/void# (04.02.2014).

[16] A *dm2e:DataResource* is a non-abstract information resource that provides RDF data. Therefore, it is a specialisation of a *foaf:Document*. In DM2E, every *dm2e:DataResource* is connected to a *void:Dataset* by means of *void:inDataset*.



16173/#f0037

URI of this Resource Map: http://data-worker.dm2e.hu-berlin.de/data/rdf/resourcemap/bbaw/dta/16173_f0037/1391002599515

16173/#f0037

URI: http://data-worker.dm2e.hu-berlin.de/data/item/bbaw/dta/16173_f0037

Property	Value
dm2e:1.1/displayLevel	■ false
dm2e:1.1/levelOfHierarchy	■ 2
dm2e:1.1/printedAt	■ <http://data-worker.dm2e.hu-berlin.de/data/place/bbaw/dta/Mannheim>
is edm:aggregatedCHO of	■ <http://data-worker.dm2e.hu-berlin.de/data/aggregation/bbaw/dta/16173_f0037>
?:author	■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/authority_gnd/118607626>
?:bibonumber	■ 37
dc:description	■ Page 33 from Kabale und Liebe (de)
dc:identifier	■ 16173/#f0037
is edm:isNextInSequence of	■ <http://data-worker.dm2e.hu-berlin.de/data/item/bbaw/dta/16173_f0038>
edm:isNextInSequence	■ <http://data-worker.dm2e.hu-berlin.de/data/item/bbaw/dta/16173_f0036>
dcterms:isPartOf	■ <http://data-worker.dm2e.hu-berlin.de/data/item/bbaw/dta/16173>
dcterms:issued	■ <http://data-worker.dm2e.hu-berlin.de/data/timespan/bbaw/dta/1784-01-01T00%3A00%3A00UG_1784-12-31T23%3A59%3A59UG>
dc:language	■ de
dc:publisher	■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/dta/Schwan>
edm:type	■ TEXT
dc:type	■ dm2e:1.1/Page
rdf:type	■ <http://example.org/resourcemap> ■ edm:ProvidedCHO

_f0037

URI: http://data-worker.dm2e.hu-berlin.de/data/aggregation/bbaw/dta/16173_f0037

Property	Value
dm2e:1.1/hasAnnotatableVersionAt	■ <http://media.dwds.de/dta/images/schiller_kabale_1784/schiller_kabale_1784_0037_1600px.jpg>
edm:aggregatedCHO	■ <http://data-worker.dm2e.hu-berlin.de/data/item/bbaw/dta/16173_f0037>
dc:contributor	■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/authority_gnd/1018099549> ■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/authority_gnd/1019062681> ■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/authority_gnd/115266127> ■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/dta/Christian%20Thomas> ■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/dta/Frank%20Wiegand> ■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/dta/Jakob%20Steinmann> ■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/dta/Matthias%20Schulz>
dcterms:created	■ 2014-01-09T23:23:28Z
edm:dataProvider	■ <http://data-worker.dm2e.hu-berlin.de/data/agent/bbaw/dta/Deutsches%20Textarchiv>
edm:hasView	■ <http://www.deutschestextarchiv.de/book/view/schiller_kabale_1784?p=37>
edm:isShownAt	■ <http://www.deutschestextarchiv.de/book/view/schiller_kabale_1784?p=37>
edm:isShownBy	■ <http://media.dwds.de/dta/images/schiller_kabale_1784/schiller_kabale_1784_0037_1600px.jpg>
edm:object	■ <http://media.dwds.de/dta/images/schiller_kabale_1784/schiller_kabale_1784_0037_1600px.jpg>
edm:provider	■ <http://data-worker.dm2e.hu-berlin.de/data/agent/DM2E>
dc:rights	■ Distributed under the Creative Commons Attribution-NonCommercial 3.0 Unported License.
edm:rights	■ <http://creativecommons.org/licenses/by-nc/3.0/>
rdf:type	■ <http://example.org/resourcemap> ■ ore:Aggregation

Ferdinand. (erwacht aus einer dumpfen Betäubung) Iſt er weg? War das eines Vaters Stimme? — Ja! ich will zu ihr — will hin — will ihr Dinge ſagen, will ihr einen Spiegel vorhalten — Nichtswürdige! und wenn du auch noch dann meine Hand verlangſt — Im Angeſicht des verſammelten Adels, des Militärs und des Volks — Umgürte dich mit dem ganzen Stolz deines Englands — Ich verwerfe dich — ein teutſcher Jüngling!

(er eilt hinaus.)

Zwei=

This page shows information obtained from the SPARQL endpoint at http://localhost:9997/dm2e-direct/sparql.

As Turtle | As RDF/XML | Browse in Disco | Browse in Graphite Browser

Figure 10: Pubby-based Linked Data API.

The main extension of Pubby is the support for custom SPARQL queries for different resources. Together with the new ability to publish information about more than one resource at once (as shown in Figure 10: A resource map, containing data about a CHO and an accompanying aggregation), this enables the publication of such diverse resources like versioned datasets, linksets, manuscripts, pages, agents, persons, or locations.

The Linked Data API is the only interface that is accessed by the scholarly environment, more precisely, by "Feed the Pundit"[17] (in the following referred to as "Feed"). Feed is a component developed in WP3 in order to provide an as-a-service access to Pundit. It provides a simple REST API to specifically interact with DM2E data, basing on the DM2E model and on Linked Data.

Feed can be called with arbitrary Linked Data resources using the DM2E model, by means of the following REST API URL:

http://feed.thepund.it/?dm2e={DM2E_Linked_Data_URL}&conf={Pundit_Configuration}

{DM2E_Linked_Data_URL} is the dereferenceable URL of some resource represented in RDF in conformance to the DM2E model, and {Pundit_Configuration} is one of the preloaded configurations of Pundit.

When the API is called, Feed gets the RDF representation of the resource and parses it to extract relevant metadata (e.g. authorship, dates and, in general, connections with other resources that might be of interest to display to the user) and, most importantly, it grabs the content to be annotated (e.g. an image, a text of simple HTML) from the actual content provider. This is done "on the fly" via a special RDF property defined in the DM2E model (*dm2e:hasAnnotatableVersionAt*).

An HTML representation of the resource is then created and loaded into Pundit, using the provided configuration.

At the moment two kind of resources are supported:

1. Manuscripts (or books): in this case a navigation bar is shown that allows users to go to a specific page;
2. Pages: in this case the content is shown, ready to be annotated, and a link to the entire manuscript is provided.

More details about Feed can be found in D3.3 "E-Learning Courses published" in the section "A3 - Consuming DM2E data in Feed".

[17] Feed the Pundit: http://feed.thepund.it/ (04.02.2014).

5 DM2E Model

The DM2E model specifies, how all resources provided via DM2E are described. Additionally, it links these resources to the ingestion process by means of links between resource maps, datasets and workflows. The DM2E model is a strict specialisation of the Europeana Data Model which allows all DM2E resources to be treated as Europeana resources.

5.1 Published Operational Version 1.1

Since the last deliverable D2.2, the DM2E model has been continuously updated to meet the further developed requirements from the data providers. The model is now in its final revisions and is aimed to be part of the final version of the interoperability infrastructure of the project. The latest version of the model, version 1.1, was published on May 2, 2013 and is currently under the twelfth revision. It is planned to be officially published as version 1.2 by the middle of March 2014.

Since the first publication of the DM2E model, a lot of modifications have been done on the data model specification. One of the most important modification was the update of all references to Europeana classes and properties regarding the latest specification of the Europeana Data Model, version 5.2.4, which was published on July 14, 2013. Apart from the synchronization of the DM2E model with the EDM, there have also been made some structural changes to the model by adding new required properties and by changing logical constraints of some class-properties-axioms. A new important property that was added is *dm2e:hasAnnotatableContentAt* (replaces *korbo:hasAnnotableVersionAt*) which is used with specific mime types that further specify the content to which this property refers to (e.g., "text/html"). This is needed for the WP3 tool Pundit in order to better work on content that is represented in the DM2E model. The detailed description of the impact on Pundit and the relations between the model specification and the annotation platform can be found in the technical specification "DM2E Annotatable Content Specification" (Goldfarb, Morbidoni & Eckert, 2013) which is part of deliverable D1.2 "Final Integration Report".

Other properties that were added are e.g. *dm2e:cover* to describe the manuscripts cover, *dm2e:genre* to add a genre (formerly done with *dc:subject* or *dc:type*) or *dcterms:medium* to describe the physical carrier or material of the object.

We had an important discussion with WP1 about the representation of uncertainty in time resources. This problem was solved by introducing *crm:P79F.beginning_is_qualified_by* and *crm:P80F.end_is_qualified_by* for indicating if the beginning resp. the end of a timespan is certain or not. For details, please refer to Goldfarb & Ritze (2013). If the definition of external resources fit with the definition of new introduced properties in the DM2E model, they were reused. An important change regarding model constraints is the new recommendation for language tags. Language tags are no longer mandatory but strongly recommended whenever the provider can set them. The recommendation has further be changed in now following the language representation standards RFC3066 and RFC5646 which strongly suggest using an ISO-639-1 two-character code and only if it does not exist to switch to a ISO-639-2 three-character code. The full change tracking issue can be reviewed on the revision history of the specification (Dröge, Iwanowa et al., 2014.

Revision 1.1 of the model version 1.1, which is the latest revision before the release of the model version 1.2 in March 2014, includes the requirements and last changes, that have been collected during the past All-WP-Meeting. The feedback from the data provider and from Europeana staff was incorporated into this specification of the model.

5.1.1 Impact of the Data Model on WP1 and WP3

The metalevel of the model described within deliverable D2.2 has remained stable and has not been changed. The focus of the newest development on the model was on the consistency and on more deeply exchange with the data providers regarding the implementation and usage of the model within the complete infrastructure. The improvement of the model was a joint effort between WP1 and WP2. To be able to get consistent data from all provided data formats, a document with mapping recommendations has been developed under the lead of WP1 (Goldfarb & Ritze, 2013). These recommendations are defined based on provided data and the requirements of the interoperability infrastructure. An exemplary recommendation is that all data provider should only use *xsd:dateTime,* which is the only data type for dates supported by the SPARQL query language, for the mapping of dates. Furthermore the URI schemas for the individual (data) resources have been updated in the current model version as proposed in Goldfarb & Ritze (2013).

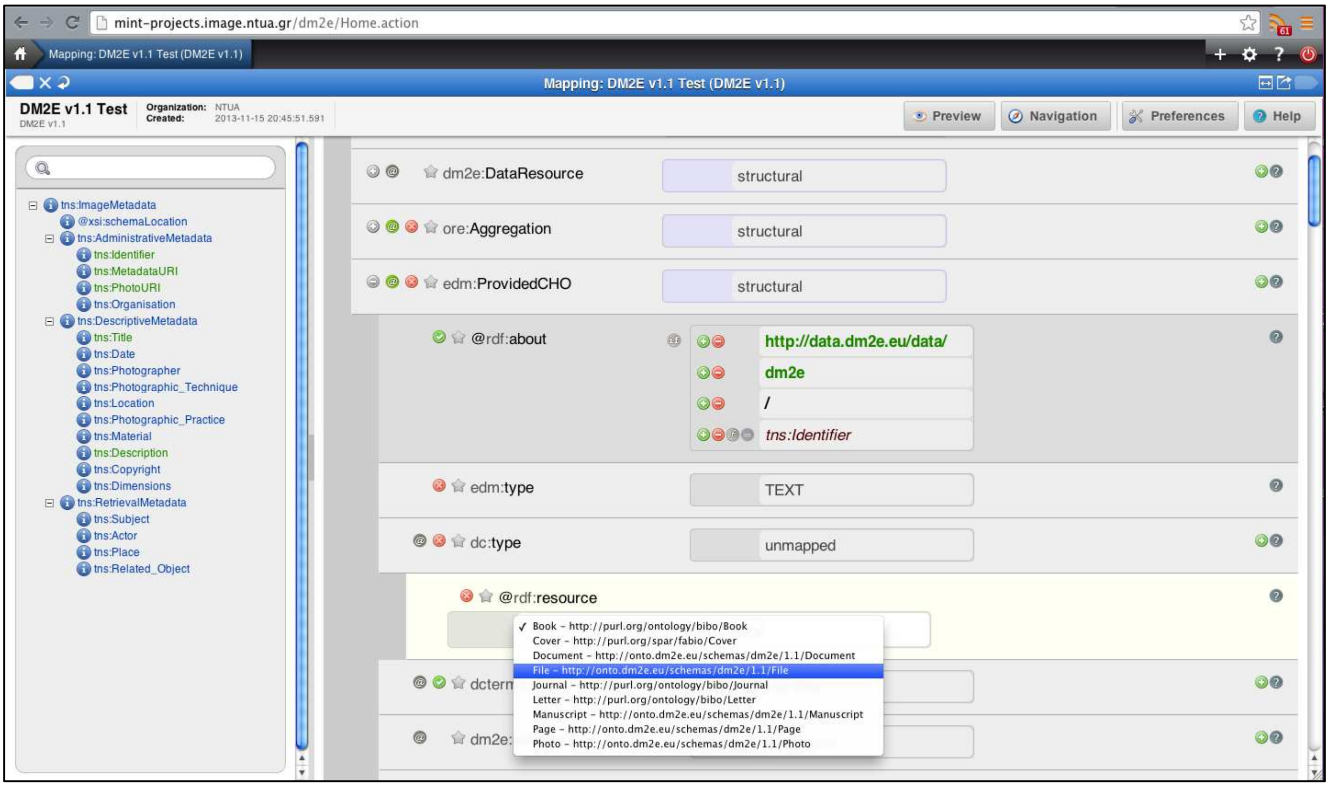
Another important step towards a user friendly interface for the DM2E infrastructure is the representation of hierarchical objects for browsing and searching functionalities. Therefore, it was necessary to define a generic, provider- and format-agnostic structure for representing hierarchical objects. The representation is based on the "Recommendations for the representation of hierarchical objects in Europeana" (2012): The properties *dcterms:hasPart* and *dcterms:isPartOf* are used for relations between different hierarchical layers of an object (e.g., a journal issue, an article and pages of an articles) whereas the property *edm:isNextInSequence* is used to represent sequences of documents (e.g. a sequence of pages) within one level. Two new properties, *dm2e:displayLevel* and *bibo:number,* were introduced and defined as mandatory. The first one is used to indicate the preferred level of the CHO for the search results. Each collected CHO has only one preferred display level that can be defined as such via the value "true". This part of the CHO serves as the entry point for the user to the whole resource. *bibo:number* can be used to indicate the position of each page within the interconnected sequence of the hierarchical object. This will allow the user to jump to any subpart of the CHO.

Since the latest revisions of the DM2E model, a lot of mappings have been done by the data providers which were quite helpful for a much improved understanding of the specific metadata. One of the results from these tests was the identification of weak points in the model based on the analysis of the data and mappings. The metadata records of the Austrian National Library, for example, include very large and heterogeneous subject headings. This kind of data is very interesting when used for internal and external contextualisation scenarios. For that purpose, the *dm2e:genre* property was established to collect such metadata as *skos:Concepts*. This is one of the possibilities how to bridge between the data model specification and the contextualisation approach.

6 Mapping Creation

The XSLT language is used for the implementation of formal crosswalks to transform providers' submitted XML or CSV metadata to the DM2E model. For this purpose, MINT (Drosopoulos, Tzouvaras et al., 2012) was originally deployed as a standalone tool implementing version 1.0 of the DM2E model in XSD and, providers were trained in its use in order to create a first draft of mappings for their data (D2.1 "Initial Version of the Interoperability Infrastructure").

MINT is a Web-based platform that facilitates aggregation initiatives for cultural heritage content and metadata. It is employed from the first steps of such workflows, corresponding to the ingestion, semantic alignment and aggregation of metadata records, and proceeds to implement a variety of publication approaches. MINT's mapping editor formalises the notion of a metadata crosswalk through a user-friendly graphical environment, where interoperability is achieved by guiding users in the creation of mappings between input and target elements (Figure 11). User imports are not required to include the respective schema declaration, while the records can be uploaded as XML or CSV files. User's mapping actions, ranging from simple drag and drop, data entry and selection from enumerated lists, to applying complex conditions and data manipulation functions, are serialised as XSLT style sheets. Those are stored and can be applied to imports, exported for further editing and, shared with other users to act as template for their mapping needs.



DM2E v1.1 Test
DM2E v1.1
Organization: NTUA
Created: 2013-11-15 20:45:51.591

dm2e:DataResource structural
ore:Aggregation structural
edm:ProvidedCHO structural
@rdf:about http://data.dm2e.eu/data/ dm2e / tns:Identifier
edm:type TEXT
dc:type unmapped
@rdf:resource
dcterm
dm2e:

✓ Book – http://purl.org/ontology/bibo/Book
Cover – http://purl.org/spar/fabio/Cover
Document – http://onto.dm2e.eu/schemas/dm2e/1.1/Document
File – http://onto.dm2e.eu/schemas/dm2e/1.1/File
Journal – http://purl.org/ontology/bibo/Journal
Letter – http://purl.org/ontology/bibo/Letter
Manuscript – http://onto.dm2e.eu/schemas/dm2e/1.1/Manuscript
Page – http://onto.dm2e.eu/schemas/dm2e/1.1/Page
Photo – http://onto.dm2e.eu/schemas/dm2e/1.1/Photo

Figure 11: Screenshot of an active mapping session in MINT.

MINT offered an alternative visualisation of the DM2E data model during its drafting period and was used to instruct content providers regarding the requirements and expectations for the creation of XSLT for their inhouse-metadata. A process was established in order to ensure the migration of existing mappings to the latest version of the DM2E model as this was updated by the respective working group of the project. A number of dedicated workshops were held during project meetings to provide training, support and hands-on examples for the data providers. Providers have since been creating mappings in MINT with the purpose of using them for the ingestion in OmNom, as is or after extending and adjusting them by hand. For the final version of the infrastructure, the user can access the mapping editor from OmNom and the mappings that have been created in MINT in order to use in workflows to transform the data to the DM2E model.

Task 1.3 "Testing the User Interface for Creating Mapping, Interlinking Heuristics and for Configuring the Workflow" dealt with evaluating the MINT platform for creating mappings. A questionnaire was designed for evaluating the functionality and the usability of MINT for mapping provider source data to the DM2E model. The results highlighted potential shortcomings when using specific XML schemas (e.g. MARC standards) as well as general features and requests from providers regarding the tool's usability and functionalities (for details, refer to deliverable D1.2). This resulted in a first update of MINT that introduced new functionality with respect to conditional mappings and the use of functions. Specifically, it allowed the use of negative closure for functional mappings and the use of "else" in conditions. Moreover, a wish-list has been drafted as a result of the questionnaire analysis that NTUA is going to address. This includes the following features:

- Allow the use of custom XPath expressions on top of the simple Drag&Drop XPath actions in the user input task before the creation of a mapping that defines the root and unique ID elements in the input metadata schema.
- Extend the use of functions and the ability to perform value mappings between the input and target schema elements, for element attributes as well.
- Fix a bug in custom functions that results in wrong escaping of certain characters and subsequent errors in the produced XSLT.
- Copy an element's mapping, along with any present conditional statements to another target schema element.

While the first feature, allowing for custom XPath expressions outside the mapping editor, cannot be implemented without significantly changing the inner workings of MINT, features 2 and 3 will be implemented in the normal course of maintenance releases. The last request involves the addition of new user interface elements and an update of the XSLT generation process making it a longer term target.

The MINT-approach is quite comfortable even for non-programmers to transform legacy data from local, mainly standalone applications to the Europeana Data Model (EDM) due to the graphical interface. It turned out, however, that it gets difficult to use for more complex mapping cases, especially when metadata encoding standards like MARC-21 or Encoded Archival Description (EAD) are used for the input data. Therefore, all data providers developed customised XSLT-scripts for transforming legacy data to DM2E, either from scratch, or based on a preliminary mapping created in MINT. Naturally, no graphical user interface can provide the same flexibility than a complex mapping language like XSLT, especially as it is based on XSLT and can never provide anything beyond it.

A further advantage of using customised XSLT mappings is the easier adaption of existing mappings to new data. This is commonly required for the above mentioned metadata encoding scheme, but also for Text Encoding Initiative (TEI) files, as they all provide a degree of freedom leading to various "dialects". Adjusting existing XSLT-scripts provides a higher flexibility to address these differences adequately.

7 Contextualisation

Contextualisation is the generation of additional links connecting Linked Data sources to one another, thus extending the Web of Data. The links are expressed by RDF triples, connecting the two data sources by URIs in the subject and object position with the predicate being one that expresses relations like similarity, e.g. *owl:sameAs*.

In the case of the DM2E project, we are faced with two different scenarios concerning contextualisation. One is the deduplication of internally used resources where identifiers are provided on collection, provider or global level. The other is the enrichment of data especially to external sources where no identifier is present and a URI is directly derived from a literal. Both scenarios are described in detail below.

As already described in deliverable D2.2 (“Intermediate Version of the Interoperability Infrastructure”), the system used for this task is Silk[18] (Volz, Bizer et al., 2009). It is included in the suite of DM2E tools and has direct access to the ingested data.

7.1 Use Cases for Contextualisation

Goldfarb & Ritze (2013) give instructions for the construction of the URIs during the mapping. Having generic rules for the URIs facilitates the rules that are to be written for the deduplication and also the enrichment with Silk. Regarding agents, places and concepts, the recommendations describe three different cases concerning the construction of URIs, depending on whether they are linked to external authorities, internal authorities or not linked at all.

In the first case, the generated URI contains the name of the authority used (e.g. GND, VIAF), followed by its respective identifier:

http://data.dm2e.eu/data/agent/onb/authority_gnd/118751069

In the second case, the generated URI contains a unique identifier from an internal authority either on the level of the provider or on the level of the collection:

http://data.dm2e.eu/data/agent/onb/123456

Thirdly, if there is no identifier at all present, the URI of a resource is to be generated on the level of the CHO, using the literal, preceded by a generated ID:

http://data.dm2e.eu/data/agent/onb/codices/AL00158123/Xyz123_Johannes_Rosos

Using the string manipulation tools (transformations) that are integrated in Silk, the literal is then stripped again and can be checked against external authority services.

The following two sections describe the two intended scenarios for the contextualisation of DM2E data in more detail.

7.1.1 Deduplication During Ingestion

The deduplication of agents/places/concepts can be performed on three levels: on the collection level, on the provider level and on the global level. A deduplication on the collection level tries to find duplicates only within the same collection. Whenever all

[18] Silk: http://wifo5-03.informatik.uni-mannheim.de/bizer/silk/ (04.02.2014).

agents/places/concepts are only derived from external authority files (case 1), no deduplication is necessary. If they are derived from internal authority files or controlled vocabularies (case 2), it can be checked whether instances from an external authority file are also contained in the same collection and can be mapped. This can especially be the case, if providers started to link to external authority files but the collection still contains local instances. As soon as only literals are available (case 3), detecting duplicates gets more complicated and error-prone. For example, places like "Frankfurt am Main", "Frankfurt a. M.", "Frankfurt a/M" etc. refer to the same instance but have different labels.

Within several collections of one provider, the same agents/places/concepts can occur more than once. Especially for searches on all collections of a provider, it is preferable to know which instances in different collections are indeed the same. Similar holds for the global level, e.g. to search for all CHOs where a certain agent has been involved.

In order to keep the data as it has been ingested and not to insert incorrect links in the data directly, we use linksets (cf. http://www.w3.org/TR/void/#linkset). Within a *void:Linkset*, the links indicate the relation between instances, in this case sameAs-links indicate that two instances are duplicates.

7.1.2 Enrichment

The second scenario in the contextualisation task is the enrichment of metadata. Here, the focus is on metadata that is not yet connected to external authority files. As was evident from the first assessment of the original data a year ago, there was already a considerable amount of metadata dealing with persons or corporate bodies that was connected to external resources like the GND. With most of the data being ingested now, the enrichment part of the contextualisation task is starting. As a first step, we analysed which external sources are worth to be linked. This includes first and foremost linking to subject headings, geospatial information and authority data on persons. In particular, extensive knowledge bases like DBpedia or Freebase are invaluable sources for the creation of linksets.

The sources we intend to use are shown in Table 2 below, sorted by the kind of information they provide. Not all of the sources provide public query endpoints[19], but in most of these cases, a data dump is available which can be used instead. For Silk, we can use them either as a dumped RDF file or via a SPARQL endpoint.

With these resources in place, rules are written that try and match the items of DM2E data with external resources to create new links. Similar to the deduplication, these links are again provided in linksets. The way the linking with Silk is performed will be explained in the next Section.

[19] OKFN maintains a website for monitoring availbilty and performance of public SPARQL endpoints: http://sparqles.okfn.org/ (04.02.2014).

Name	Contents	Public query endpoint
GND (https://portal.dnb.de)	Persons, corporate bodies, titles	None, data dump available at http://datendienst.dnb.de/cgi-bin/mabit.pl?userID=opendata&pass=opendata&cmd=login
VIAF (http://viaf.org)	Persons	None, data dump available at http://viaf.org/viaf/data/
DBpedia (http://dbpedia.org/About)	Wikipedia	SPARQL: http://dbpedia.org/sparql
Yago (http://www.mpi-inf.mpg.de/yago-naga/yago/	Wikipedia, WordNet, GeoNames	SPARQL: http://lod2.openlinksw.com/sparql
Freebase (http://www.freebase.com/)	People, places, things	MQL API: https://www.googleapis.com/freebase/v1/mqlread
LCSH (http://authorities.loc.gov/)	Subject headings	None, data dump available at http://id.loc.gov/download/
DDC (http://dewey.info)	Subject headings	SPARQL: http://dewey.info/sparql.php
Lobid (http://lobid.org/organisation)	Library information	SPARQL: http://lobid.org/sparql/
Geonames (http://www.geonames.org/)	Geospatial information	none, data dump available at http://www.geonames.org/ontology/documentation.html
Linked Geodata (http://linkedgeodata.org/About)	Geospatial information	SPARQL: http://linkedgeodata.org/sparql
InPho (https://inpho.cogs.indiana.edu/)	Philosophy	REST API: https://inpho.cogs.indiana.edu/docs/
Europeana (http://www.europeana.eu/)	Cultural Heritage	SPARQL: http://europeana.ontotext.com/sparql
JudaicaLink (http://judaicalink.org/)	Jewish Culture and History	Under development, SPARQL will be available.

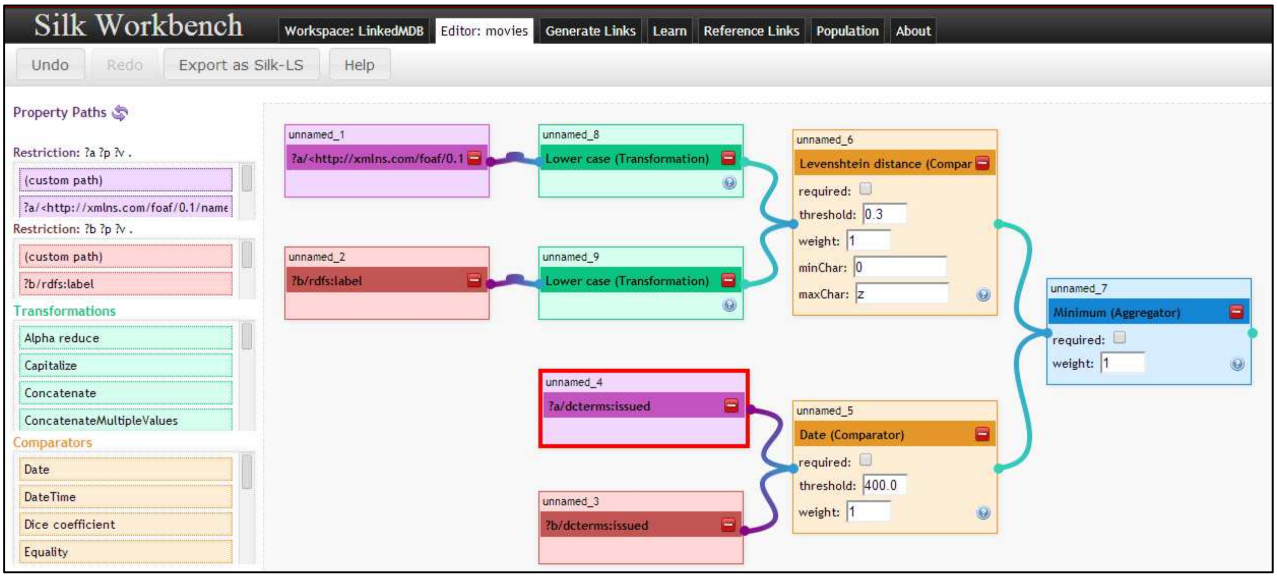
Table 2: Overview of the external datasets.

7.2 How to Work with the Silk Workbench

The tool which will be used for contextualisation is the Silk framework. It is able to detect relationships between linked data sources (Volz et al. 2009).

One component of the framework is the Silk workbench. For the DM2E project, there is one central instance of the workbench that can be found on http://context.dm2e.eu. The workbench is a graphical interface to generate linkage rules. Figure 12 shows an example of a linkage rule in the Silk workbench.

Linkage rules specify which conditions must hold to generate a link between instances of the data sources. Within a linkage rule, different operators can be applied like transformations, similarity measures and aggregations. Transformations take as input strings/numbers of attributes (purple and red boxes in Figure 12) and transform them in a certain way, e.g. lowercase all letters or split them up into single tokens (green boxes in Figure 12). This serves as a preprocessing step to better compare the attributes with similarity measures like Levensthein, Jaccard etc. (yellow boxes in Figure 12). Each comparison assigns a value to a pair of instances indicating how likely it is that a relation between this pair exists. Since elements often have several attributes, more than one comparison might be useful. In the end, the comparisons are combined in one aggregation (blue box in Figure 12). Within an aggregation, it is possible to weight the different comparisons if some are more important than others. Additionally, a threshold can be specified to indicate how high the resulting value of the aggregation or a single comparison must be such that a link is generated.



unnamed_6: required: □ threshold: 0.3 weight: 1 minChar: 0 maxChar: z

unnamed_7: required: □ weight: 1

unnamed_5: required: □ threshold: 400.0 weight: 1

Figure 12: Silk workbench.

An example is the following one: A dataset includes authors which are described by their name and their year of birth. This dataset should be linked to persons in DBpedia. Figure 13 shows the possible matching between instance a1 from the dataset (blue) and a2 from DBpedia (yellow). An exemplary linkage rule compares the year of a1 with the birthyear of a2 using the numerical distance comparator. Based on the Levenshtein distance, the nameof a1 is compared with the birthname of a2. If both comparisons discover a certain similarity (e.g. above 0.8), a link between a1 and a2 is created.

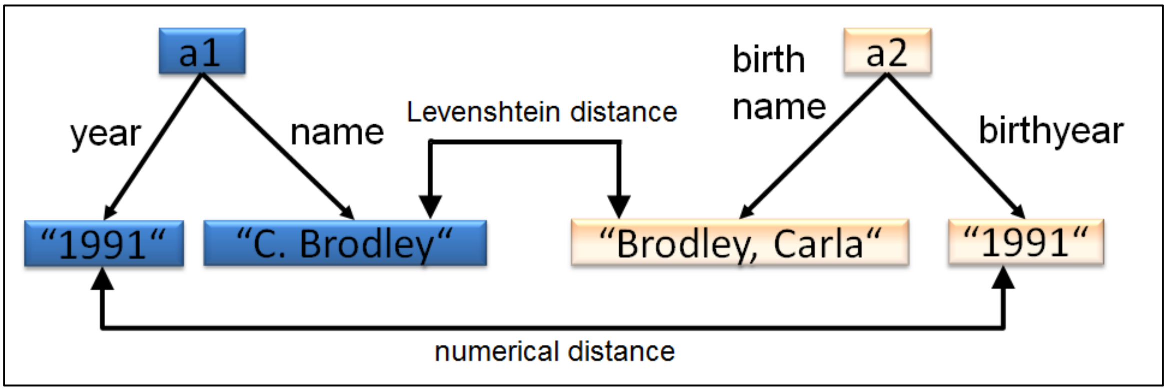


Figure 13: Example of different comparisons between instances a1 and a2.

In this case, it would be a sameAs-link. Within the project, we focus on sameAs-links but it is also possible to consider generating other links like hierarchical relations.

However, the quality of the links highly depends on the data and the amount of available information, e.g. if only a label is present, the link might not be very certain. Whenever it is important to only have correct links, links that only rely on a comparison of one attribute might be too unsure.

Since the manual creation of linkage rules can be very time-consuming and elaborating, the Silk Workbench provides the possibility to semi-automatically learn linkage rules. Therefore, an active learning approach using genetic programming is applied (Isele & Bizer, 2013). Based on the user's judgment if a link is correct or not, the algorithm learns the best possible linkage rule. An optimal rule achieves 100% F-measure. F-Measure is the harmonic mean between precision (the amount of correctly detected links) and recall (how many of all existing links are found). The active learning component tries to reduce the number of links that have to be provided by the user.

The created linkage rule can then be exported, made accessible via a URI and used within the Silk integration of OmNom (cf. the following Section, Technical developments).

Besides all the opportunities to generate links with Silk, there are also a few limitations. Without extending Silk, only the predefined transformations, comparisons and aggregations can be used. Further, Silk focuses on structured data and does not provide methods to consider unstructured data. Whenever longer parts of text are included in the data, they cannot properly be taken into account although the information in the text could be very useful. Additionally, the learning algorithm only considers attributes which are directly attached to the instances. For example, a publication can be connected to authors which are instances themselves. The attributes of the authors are not taken into account although especially comparing the names of the authors is very useful.

7.3 Technical Developments

To enable a continuous integration in the whole workflow system, we implemented a Silk Web service. This Web service acts like every other Web service in OmNom. It takes as input a URI to a Silk configuration file and generates the links based on this configuration. Therefore, we used the Silk Single Server Machine and built our implementation around it to fit our needs. Usually, Silk takes as input in the configuration either a local RDF dump or a SPARQL endpoint. Since the user cannot manually upload any files on our server, it is not possible to specify RDF dumps but not every resource has a SPARQL endpoint. Thus, we need to provide the possibility to specify URIs as input in the configuration. Whenever an URI is given, we download the content and write it into a local file on the server. Afterwards, we internally rewrite the configuration and replace it with the local file input.

With this workaround, we enable the usage of URIs as input without the need to directly adapt Silk.

Nevertheless, the input sources must either be available via a SPARQL endpoint or be provided as an RDF dump in the Web. Furthermore, the Silk configuration file must also be accessible via a URI.

As result of the Silk Web service, a linkset is created, containing the links between the instances. These linksets can then be published in the triple store similar to the ingested data. Using the Linked Data API, whole linksets can be displayed or only the links which have been generated for a certain instance. Thus, additional information about the instances is provided.

Using the linksets, it is obvious that the links have been generated by Silk and were not included in the ingested data itself. Moreover, also the provenance of the matching can be inspected to get to know which configuration has been used to create the according links.

The deployed Silk tool is the latest stable version 2.5.3 (from March 6, 2012)[20]. Documentation for the tool in general can be found in the Silk wiki at https://www.assembla.com/spaces/silk/wiki/Home. Information about the deployment in DM2E can be found in the developer documentation (see Section 10.2).

The software was deployed on a Tomcat 7 Web container configured to run with the options `JAVA_OPTS="-Djava.awt.headless=true -Xms2048m -Xmx8192m -XX:PermSize=1024m -XX:MaxPermSize=1024m`”, that is, a maximum use of 8GB of RAM. Depending on the use of the machine, the maximum amount of allocated RAM can still be raised later on, seeing the total of 16GB installed.

Deploying the latest release as is resulted in two drawbacks. Firstly, the working directory is called `.silk` (and thus hidden on Unix/Linux systems) and situated in the home directory of the user which in this case is the user that also runs the server. Subsequently, the working directory is situated at `/usr/share/tomcat7/.silk`. Secondly, the integration into the DM2E’s Single Sign-On (see next Section) cannot be completed without changes being made to the software itself. As Silk has no user management, a mechanism for logging in is not implemented, but would be necessary for that. However, the Silk service is integrated into the Single Sign-On with the exception that there is no possibility of logging out of the system while visiting Silk.

7.4 Potential Contextualisation Material in the Data Providers’ Content

The datasets provided by the content providers differ in their description of the manuscripts. They contain various instances like publications, persons, organisations, places and subjects. Following, we describe which instances are contained in the individual datasets, whether they are already linked and which challenges we face when matching them.

[20] Silk was directly downloaded from http://wifo5-03.informatik.uni-mannheim.de/bizer/silk/releases/ (05.02.2014).

7.4.1 ABO (ONB)

Within the ABO (Austrian Books Online) collection, provided by the Austrian National Library (ONB), publications, persons, organisations and subjects are included. A lot of instances are already linked to persons and concepts of the GND (Integrated Authority File). An example is shown below:

```
<foaf:Person
rdf:about="http://data.dm2e.eu/data/agent/onb/authority_gnd/118579274">
     <skos:prefLabel>Mauritius</skos:prefLabel>
     <owl:sameAs rdf:resource="http://d-nb.info/gnd/118579274"/>
</foaf:Person>
```

Besides the sameAs-link, the connection to the instance of the GND is already indicated in the URI which contains the GND identifier and the information that this identifier belongs to the GND. Existing links to such external datasets can be used to generate further links, e.g. to link to DBpedia. For instances which are not already linked to the GND, the link generation is more difficult because in most cases, only a prefered label is given. Further information like the year of birth for persons are usually not included. A linkage rule which only relies on the preferred label can lead to several problems, e.g. when two persons share the same name. Especially for places, different spellings variations, abbreviations etc. complicate the matching, as in our above mentioned example: “Frankfurt am Main”, “Frankfurt a. M.”, “Frankfurt a/M” etc. One possible solution is to exploit alternative spellings and abbreviations which are included in some of the external resources, e.g. GND or GeoNames.

7.4.2 WAB (UIB)

Instances of the Wittgenstein Archives at the University of Bergen (WAB) are publications, persons and organisations. They are not linked to any resources. Further, only a small amount of instances in contained in the dataset at all. Again, in most cases only a prefered label is given.

An example is shown below:

```
 <foaf:Person
 rdf:about="http://data.dm2e.eu/data/agent/uib/wab/Wilhelm_Busch">
      <skos:prefLabel>Wilhelm Busch</skos:prefLabel>
    </foaf:Person>
```

The name “Wilhelm Busch” is ambiguous. Thus, a linkage rule which only takes the prefered label into account might create incorrect links.

Besides the person, a concept with the label “Eduards Traum” is included in the dataset. This concept has the person Wilhelm Busch as creator.

```
<skos:Concept
rdf:about="http://data.dm2e.eu/data/concept/uib/wab/Busch_Wilhelm_Eduards_Traum">
      <skos:prefLabel>Eduards Traum</skos:prefLabel>
      <skos:altLabel>Busch, Wilhelm: Eduards Traum</skos:altLabel>
```

```
     <dc:creator
rdf:resource="http://data.dm2e.eu/data/agent/uib/wab/Wilhelm_Busch"/>
 </skos:Concept>
```

In the GND we can detect that “Eduards Traum” is the name of a publication by Wilhelm Busch. With this information, it is not only possible to link the concept “Eduards Traum” (http://d-nb.info/gnd/4341230-0) to the according publication in the GND, also the creator “Wilhelm Busch” can be linked to the correct person (http://d-nb.info/gnd/118517880). Having the links to the GND, we can again use the information to link to further resources like DBpedia. Such strategies, where matching decisions are not independent of each other, are called collective matching approaches.

7.4.3 Harriot (MPIWG)

In the Harriot dataset, provided by the Max Planck Institute for the History of Science (MPIWG), persons are already linked to persons in the GND. Other instances are organisations and places which are not linked to external resources. An example of instances that are not linked are places, e.g.:

```
<edm:place rdf:about= ”http://data.dm2e.eu/data/place/mpiwg/harriot/MPIWG-RY8Bby2zrBC_Petworth_UK”>
     <skos:prefLabel xml:lang=”en”>Petworth, UK</skos:prefLabel>
</edm:place>
```

Since only a few instances are contained in the dataset at all, not many links are expected.

7.4.4 Rare Book Collection (MPIWG)

This dataset includes persons, places and keywords. All keywords have already been linked to external resources, but only a small number of persons has been linked to the GND. Neither places nor organisations have been assigned external identifiers, so far.

Person names can be ambiguous, so it is advisable to take other data into account. For example, Robert Smith in the following description is a very common name:

```
<foaf:Person
rdf:about="http://data.dm2e.eu/data/agent/mpiwg/rareBooks/MPIWG-T7U22T80_gXW076U_Robert_Smith">
     <skos:prefLabel>Robert Smith</skos:prefLabel>
</foaf:Person>
```

However, with the help of the publication date (1767), the number of possible candidates can be reduced to one: http://d-nb.info/gnd/100339026/about/rdf.

The places in the data prove to be more difficult, as there are ancient place names to be dealt with as well as unstructured data as in “Norimbergae” or “Franckfort und Leipzig”.

While the first case is solvable as long as such a place name variant is available in the external authority, the second is rather hard to get by.

7.4.5 DTA (BBAW)

The Berlin-Brandenburg Academy of Sciences and Humanities (BBAW) provides the Deutsche Textarchiv (DTA) to the project. The source of this metadata is TEI-XML fulltext, which contains text marked up as italics or bold that was used to unsemantically markup keywords, persons and corporate bodies. These elements were mapped as *skos:Concept*. The enrichment process would attempt here to map these concepts to the right class and as a further step to other vocabularies (mapping of a concept, which is a person to foaf:Person and in a further step, connect that to its GND entry).

7.4.6 Dingler (UBER)

The "Polytechnisches Journal", often named "Dingler" based on the name of its first publisher, is provided by the Humboldt-Universität zu Berlin (UBER). The metadata is TEI-XML P5 encoded and available for the fulltext of the journal. The most potential target for enrichment is the very high number of authors and persons (around 27,000) mentioned in the journal, mapped e.g. as

```
<dm2e:mentioned>
       <foaf:Person
rdf:about="http://data.dm2e.eu/data/agent/uber/dingler/Agar_Ellis"
>
       <skos:prefLabel xml:lang="de">Agar Ellis</skos:prefLabel>
     </foaf:Person>
 </dm2e:mentioned>
```

Seeing that GND contains also information on the journal, this may be a promising target.

7.4.7 Codices (ONB)

The Codices dataset covers publications, persons, organisations and places. Some of the fields contain unstructured data. This unstructured data can nevertheless include instances which can be linked. An example would be this value for the field *dcterms:provenance*: "Wolfgang Lazius (Dr. med.; *1514; Uni. Wien; Mediziner, Historiograph, Humanist; +1565): Besitz vermutet [Menhardt, Altdeutsche Handschriften, 1960/61, 10]. (ger)". When comparing this string with instances of external sources like persons in DBpedia, it is very likely that no link will be created to http://dbpedia.org/resource/Wolfgang_Lazius since not only the name "Wolfgang Lazius" is contained. Thus, a string-based comparison will result in a low similarity value. However, the additional information can also be exploited especially if more than one link comes into question, e.g. by comparing the year of birth and death of possible matches. Therefore, we need to identify which part in the whole string refers to an instance and which parts are further descriptions.

7.4.8 Manuscript Collection (UBFFM)

The manscript collection provided by the University library JCS Frankfurt (UBFFM) already includes mappings of persons (authors and other contributors like copyists) to the GND. Other data, like places or titles serve as obvious candidates for the contextualisation task.

Further links to other external resources are not contained. Applying Silk, we assume to already get a huge amount of links. In the other cases, e.g. where more than one match is possible, we are developing methods to extend SILK such that even these links can be detected.

8 Export to Europeana

The export of the all DM2E data to Europeana is one of the declared goals of DM2E. In preparation of the export of DM2E data to Europeana, we developed an interface for OAI-PMH (Open Archives Initiative - Protocol for Metadata Harvesting).[21] The data will be ingested to Europeana at the end of the project, as it has been decided after discussions with Europeana that a continuous ingestion of DM2E data into Europeana during the project is not feasible, nor desired. For once DM2E's interoperability platform follows quicker iteration cycles than Europeana can handle at the moment. Secondly, the DM2E model, though soundly based on the EDM, is still evolving and constant adjustments would be necessary for Europeana to make use of the richness of the DM2E model.

OAI-PMH is not only the preferred mode of data ingestion for Europeana but a majority of the data providers within DM2E are familiar with it since it is widely used by libraries, archives and museums for metadata exchange. While we initially planned to implement an opaque, one-off solution to provide the data to Europeana, the need for an easy-to-use Search and Browse Interface (see next Section) finally turned the balance and justified the effort to build an OAI-PMH repository from scratch.

8.1 Indexing Linked Data

The main focus of the DM2E model had been on accommodating the richness of the provider's original metadata while making contextualisation and reuse within the scholarly environment developed in WP3 as easy as possible. To make the data usable for full-text search and browse interfaces, a few extensions of the DM2E model were necessary. For one, a new property *dm2e:displayLevel* was introduced, a boolean value indicating whether a specific CHO should be visible in search applications, including Europeana, i.e. whether the data should appear in the search results and be used for creation of browse facets. This was necessary as no simple rule can be given, which types of CHOs or which levels in a hierarchy of CHOs should generally be used for this purpose. While access to the lowest levels of containment of a CHO, such as a page, or as in the case of the data provided by the Wittgenstein Archives, even to paragraphs, is essential to the digital scholars working with the Pundit data augmentation tool, it would decrease precision and recall of a full-text search engine like Europeana Portal if a simple search for "Wittgenstein" would result in several thousands of CHOs for sentences instead of just those that are relevant to her, i.e. the manuscript level. On the other hand, datasets like the Codices provided by ONB contain manuscripts with certain pages that stand on its own as CHOs worth to be found, for example because they are magnificiently adorned. Therefore, it is the mapping creators' decision whether or not a particular CHO is to be included in the search index. Furthermore, the mechanics how images that either are already thumbnails or can be used to generate thumbnails of a CHO were improved. Many of the changes described in Section 5.2, Impact of the data model on WP1 and WP3, were also a result of this additional use case.

8.2 Implementation

After evaluating the most popular Open Source solutions on the market, we chose an own implementation for three reasons. Firstly, the OAI-PMH standard is manageable in size and a complete implementation of the standard is simplified by the wide availability of tools to validate conformance of the final product. Secondly, all generic solutions rely on static data

[21] Open Archives Initiative website: http://www.openarchives.org/pmh/ (04.02.2014).

exports but we wanted the data in the repository to be up-to-date with the current state of the data in our triplestore. Last but not least, we wanted to leverage our Linked Data API to simplify the generation of OAI-PMH records and to prove that the Linked Data API is flexible enough to not only be usable for the scholarly domain model but for a high-performance task like bulk export. The OAI-PMH repository is written in Java,[22] re-using much of the technology stack of the OmNom toolchain. Because write-access is not required and performance is of the essence, as much data as possible is kept in a cache backed by MapDB.[23]

The OAI-PMH interface offers three metadata formats: The OAI-PMH flavor of Dublin Core[24] (oai_dc), the full DM2E data serialised as RDF/XML and EDM. The former is the baseline of OAI-PMH repositories around the globe, it is a simple, literal-based attribute-less XML variant of Dublin Core Elements, but due to it's ubiquitousness and simplicity, every OAI-PMH harvester that adheres to the standard can handle all of the exported DM2E data. While this implies a loss in richness and granularity of the data, it also means that manuscript metadata that was produced as part of the DM2E interoperability platform can readily be imported into Discovery tools like vuFind, Blacklight or Ex Libris Primo or into search engines like Solr. The mapping from the RDF-based DM2E model to oai_dc is implemented as a mix of SPARQL queries for field unification and Java code for optimal performance, in close collaboration with the providers of DM2E's main Search and Discovery interface (See next Section).

For the actual ingestion to Europeana, we need to provide the DM2E data in EDM. Thanks to the DM2E model being a specialisation of EDM and the fruitful and continuous communication between the data modelling specialists at both DM2E and Europeana, DM2E retains full backward compatibility with EDM. This means that the conversion from DM2E to EDM is trivial. In fact, what it boils down to is to replace the DM2E-specific ontology resources with their EDM base counterparts and omit the small part of the data that is DM2E-specific and therefore of no use to Europeana.

Finally, the OAI-PMH repository can deliver the complete manuscript data, including the linksets generated by the contextualisation and provenance information generated by OmNom, serialised in RDF/XML. While this data could also be retrieved directly using the Linked Data API (and internally it is), the reasoning for adding another access mechanism is sustainability: Exposing our full data using a widely adopted standard like OAI-PMH allows both full and incremental replication of our data not only by clients compatible with our Linked Data API but for all OAI-PMH harvesters. Consumers of the data are guaranteed to be up-to-date with the latest revision of the datasets and linksets and can use the data the way they see fit, be it as RDF (importing it into a triple store like Jena TDB or OWLIM) or as XML (importing it into an XML database like eXist or transforming it using XSLT). When data consumers want to retrieve old versions of datasets or access and query the data on a more granular level, they can always switch to using the Linked Data API.

[22] OAI-PMH repository source code: https://github.com/DM2E/dm2e-oai (04.02.2014).
[23] MapDB: http://www.mapdb.org/ (05.02.2014).
[24] OAI-PMH DC: http://www.openarchives.org/OAI/2.0/oai_dc.xsd (04.02.2014).

9 Search and Browse

As stated in the overview, in the original Description of Work, a dedicated search and browse facility is missing. At the end of the project, Europeana will provide access to the DM2E data, but for practical reasons, we need an own access to our data already during the project, that directly works on the data and allows the scholars to search and review what's already there.

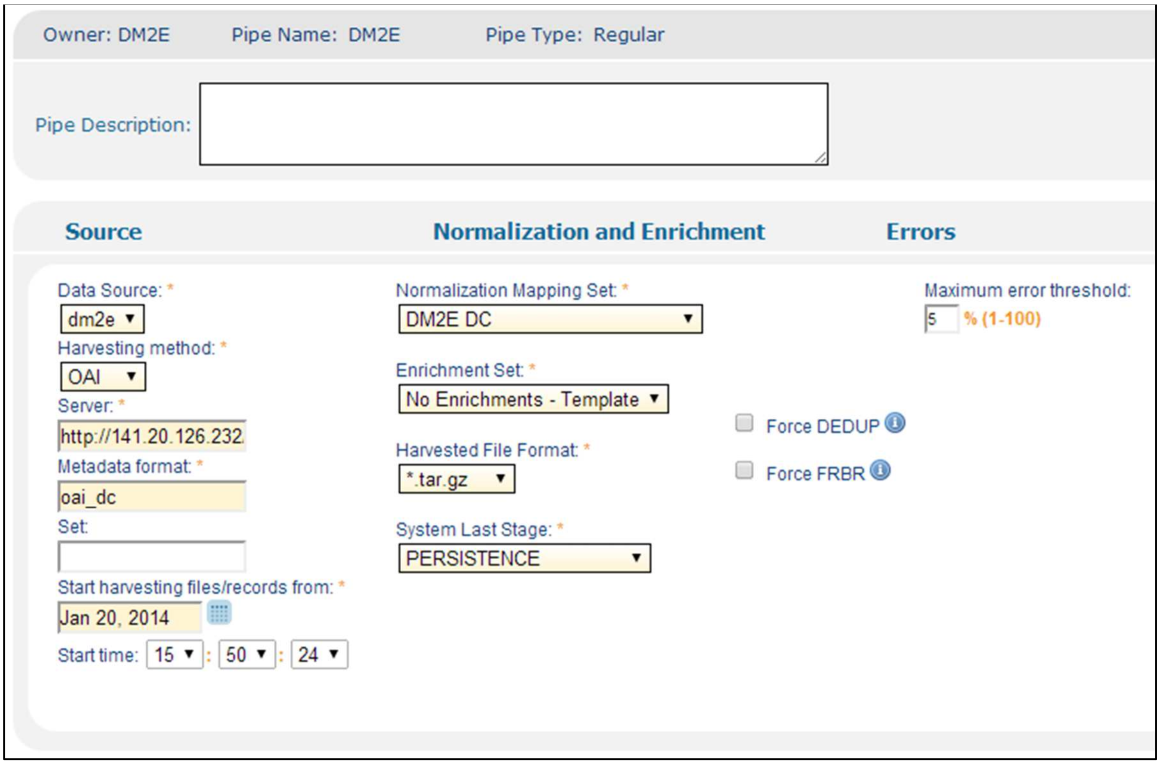
We therefore decided to fill this gap by providing an additional search and browse interface. With ExLibris as project partner, using ExLibris technology for this task seems to be straightforward. However, as DM2E is committed to openness and free software, continuously releasing the source code for all components of the interoperability platform, we did not want to develop a solution that is exclusively geared towards one particular platform and even more reluctant to employ commercial software at such a prominent place like the data access portal.

To perform this balancing act, we followed a two-track strategy. We developed the OAI-PMH repository described in Section 8 that opens the DM2E data for consumption by any indexing service, be it a commercial Discovery System, a Web search engine or Europeana. As a way to play with the data, to show that it is possible to have an open search and browse function and prove the flexibility of our architecture, Net7 rapidly prototyped a Solr-based search application, which can be used as a starting point for further search and browse solutions or other tools that want to index our data. This prototype is described in Deliverable D3.3.

At the same time, we set up an instance of the Primo Discovery system[25] that uses the same data and customised it to function as the main entry point to the DM2E data. This way, we did not invest too much time in the development of a new search portal and could provide scholars with the best out-of-the-box solutions. Since Primo is used as an addition to or even a replacement of traditional online catalogues in cultural heritage institutions world-wide, users are likely to be familiar with the interface.

Primo is a discovery system for local and remote resources, such as books, journal articles, and digital objects. It contains a publishing platform to transform various formats into the Normalised Primo XML format (PNX) which is then used for the creation of the search index, browse facets, external links and all the other features that users expect from a modern search engine interface. The standard mechanism to import data into Primo is to harvest records via OAI-PMH. Primo uses the oai_dc metadata format of the OAI-PMH interface developed in WP2. It provides all DM2E records in Dublin Core - with some custom EDM extensions where Dublin Core is not expressive enough - that can be mapped in Primo (Figure 14).

[25] Primo overview: http://www.exlibrisgroup.com/category/PrimoOverview (05.02.2014).



Owner: DM2E Pipe Name: DM2E Pipe Type: Regular

Pipe Description:

Source

Data Source: * dm2e

Harvesting method: * OAI

Server: * http://141.20.126.232

Metadata format: * oai_dc

Set:

Start harvesting files/records from: * Jan 20, 2014

Start time: 15 : 50 : 24

Normalization and Enrichment

Normalization Mapping Set: * DM2E DC

Enrichment Set: * No Enrichments - Template

Harvested File Format: * *.tar.gz

System Last Stage: * PERSISTENCE

□ Force DEDUP

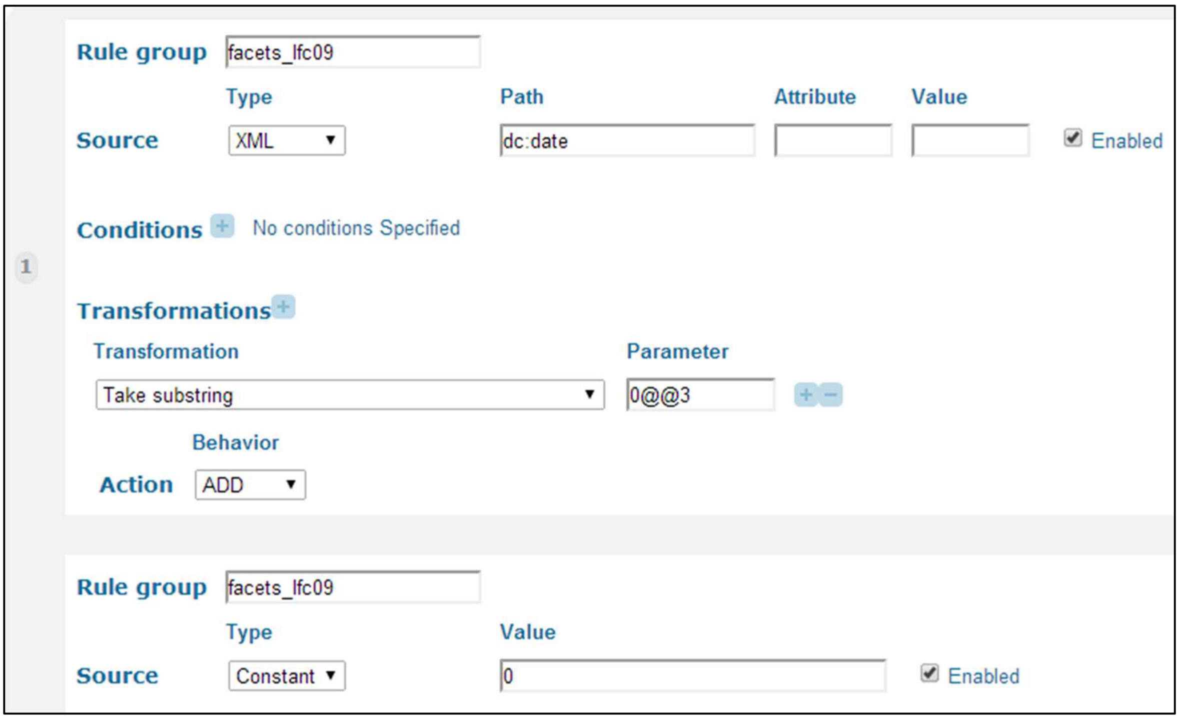
□ Force FRBR

Errors

Maximum error threshold: 5 % (1-100)

Figure 14: OAI Harvesting; Primo Pipe

This process describes the Primo pipe; harvesting of data (via OAI, FTP, SFTP or copy), normalizing records with a mapping set and indexing into a data source. For normalization we simply copied the generic mapping table for Dublin Core and made small adjustments to mend it to the EDM specifics as provided via OAI-PMH. The normalization process allows to transfer the metadata fields into several sections in Primo (control, display, links, search, sort, facets). An interesting step is the definition of the facets to support drill-down and browsing. Figure 16 illustrates the configuration in the Primo backend and figure 15 below shows how the local decade facet is generated by using regular expressions.



1

Rule group facets_lfc09

	Type	Path	Attribute	Value	
Source	XML	dc:date			☑ Enabled

Conditions No conditions Specified

Transformations

Transformation	Parameter
Take substring	0@@3

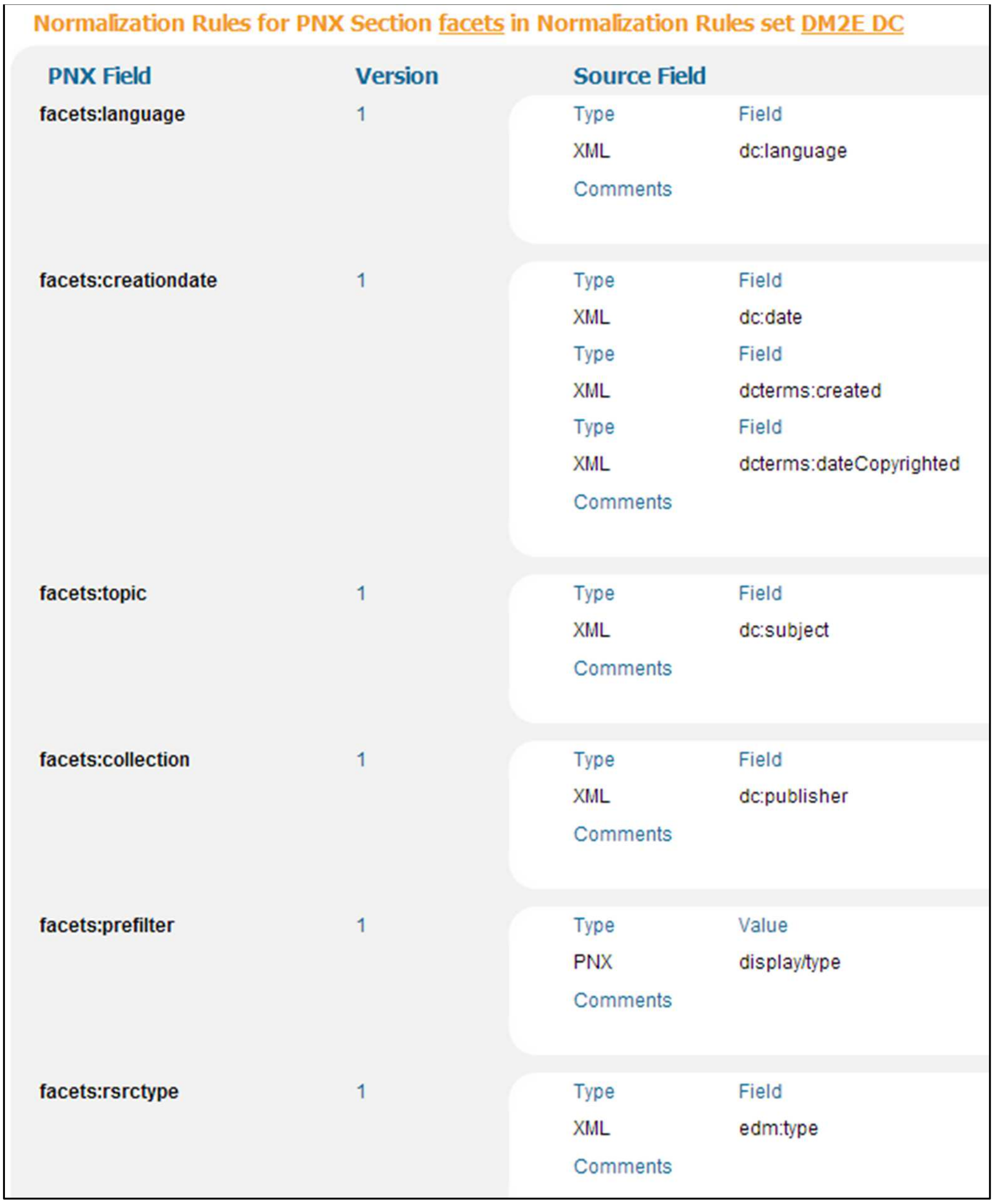
Behavior

Action ADD

Rule group facets_lfc09

	Type	Value	
Source	Constant	0	☑ Enabled

Figure 15: Mapping table for the decade facet



Normalization Rules for PNX Section facets in Normalization Rules set DM2E DC

PNX Field	Version	Source Field	
facets:language	1	Type	Field
		XML	dc:language
		Comments	
facets:creationdate	1	Type	Field
		XML	dc:date
		Type	Field
		XML	dcterms:created
		Type	Field
		XML	dcterms:dateCopyrighted
		Comments	
facets:topic	1	Type	Field
		XML	dc:subject
		Comments	
facets:collection	1	Type	Field
		XML	dc:publisher
		Comments	
facets:prefilter	1	Type	Value
		PNX	display/type
		Comments	
facets:rsrctype	1	Type	Field
		XML	edm:type
		Comments	

Figure 16: Mapping table for facets.

The benefit of using such normalization rules is to react flexible on changes in the metadata fields provided by OAI; means that additional changes and fields that probably needs to have an edm or dm2e namespace can be added as well.

Examples: edm:type, edm:dataProvider,edm:currentLocation, dm2e:shelfmarkLocation.

The following DC record that is sent via OAI is transformed to the appropriate format in Primo.

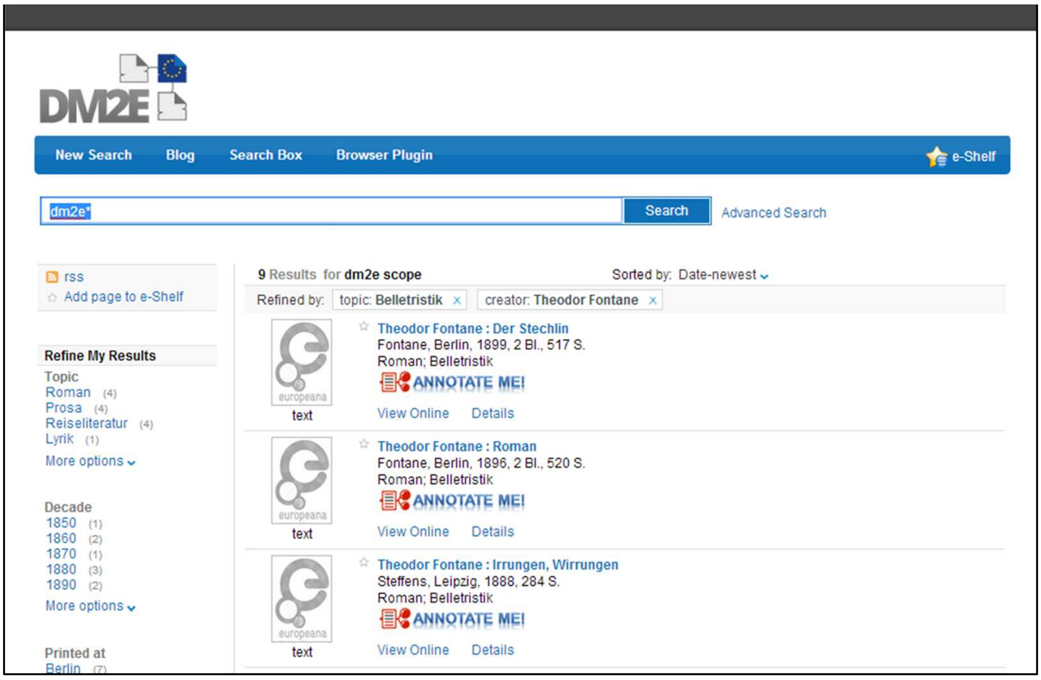
```
<?xml version="1.0" encoding="UTF-8"?>
<oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/">
    <dc:identifier linktype="thumbnail" xmlns:dc="http://purl.org/dc/elements/1.1/">http://demo.feed.thepund.it/?dm2e=null&amp;conf=timeline-demo.js</dc:identifier>
    <edm:dataProvider>bbaw</edm:dataProvider>
    <dc:title>Praktische Anweisung zum Teichbau@de -- Für Förster, Oekonomen und solche Personen, die sich weniger mit Mathematik abgeben@de</dc:title>
  <dcterms:issued>http://data.dm2e.eu/data/timespan/bbaw/dta/1798-01-01T00%3A00%3A00UG_1798-12-31T23%3A59%3A59UG</dcterms:issued>
    <dcterms:alternative>Riemann, Johann Friedrich: Praktische Anweisung zum Teichbau. Für Förster, Oekonomen und solche Personen, die sich weniger mit Mathematik abgeben. Leipzig, 1798.</dcterms:alternative>
    <dc:subject>Wissenschaft</dc:subject>
    <dm2e:displayLevel>true</dm2e:displayLevel>
    <dc:type>http://example.org/resourcemap</dc:type>
    <dc:type>http://purl.org/ontology/bibo/Book</dc:type>
    <dc:creator>Johann Friedrich Riemann</dc:creator>
    <dm2e:levelOfHierarchy>1</dm2e:levelOfHierarchy>
    <dcterms:extent>IV, 444 S., [1] Bl., IV gef. Bl.</dcterms:extent>
    <dm2e:shelfmarkLocation>SBB-PK, 50 MA 40321</dm2e:shelfmarkLocation>

      <edm:currentLocation>http://data.dm2e.eu/data/agent/bbaw/dta/Staatsbibliothek%20zu%20Berlin%20%E2%80%93%20Preu%C3%9Fischer%20Kulturbesitz</edm:currentLocation>
    <dm2e:genre>http://data.dm2e.eu/data/concept/bbaw/dta/Fachtext</dm2e:genre>
    <dc:subject>Gartenbau</dc:subject>
    <dm2e:genre>http://data.dm2e.eu/data/concept/bbaw/dta/Gebrauchsliteratur</dm2e:genre>
    <dc:identifier>17150</dc:identifier>
    <bibonumPages>467</bibonumPages>
    <dc:title>Praktische Anweisung zum Teichbau</dc:title>
    <dc:coverage>Leipzig</dc:coverage>
    <dc:language>de</dc:language>
    <dc:publisher>Fleischer</dc:publisher>
    <edm:type>TEXT</edm:type>
  </oai_dc:dc>
```

Transformed format in Primo:

```
<?xml version="1.0" encoding="UTF-8"?>
<record>
<control>
<sourcerecordid>17150</sourcerecordid>
<sourceid>dm2e</sourceid>
<recordid>dm2e17150</recordid>
<originalsourceid>dm2e</originalsourceid>
<sourceformat>DC</sourceformat>
<sourcesystem>Other</sourcesystem>
</control>
<display>
 <type>TEXT</type>
<title>Praktische Anweisung zum Teichbau@de -- Für Förster, Oekonomen und solche Personen, die sich weniger mit Mathematik abgeben@de</title>
<creator>Johann Friedrich Riemann</creator>
<publisher>Fleischer</publisher>
<format>IV, 444 S., [1] Bl., IV gef. Bl.</format>
<identifier>http://demo.feed.thepund.it/?dm2e=null&amp;conf=timeline-demo.js; 17150</identifier>
<subject>Wissenschaft; Gartenbau</subject>
<language>ger</language>
<source>dm2e</source>
<coverage>Leipzig</coverage>
</display>
<links>
<thumbnail>$$Uhttps://dl.dropboxusercontent.com/u/1415926/dm2e/europeana.png</thumbnail>
<openurlfulltext>$$Topenurlfull_journal</openurlfulltext>
<lln09>$$Uhttp://data.dm2e.eu/data/agent/bbaw/dta/Staatsbibliothek%20zu%20Berlin%20%E2%80%93%20Preu%C3%9Fischer%20Kulturbesitz$$Dedm:currentLocation</lln09>
</links>
<facets>
<language>ger</language>
<topic>Wissenschaft</topic>
<topic>Gartenbau</topic>
<collection>Fleischer</collection>
<rsrctype>TEXT</rsrctype>
<creatorcontrib>Johann Friedrich Riemann</creatorcontrib>
<library>SBB-PK, 50 MA</library>
<lfc08>Leipzig</lfc08>
<lfc09>0</lfc09>
<lfc10>bbaw</lfc10>
</facets>
</record>
```

By using Primo, users will get a single entry point to the DM2E data, from where they can narrow down their search via facets (Topic, Decade, Creator, Printed at, Publisher, Language, Library), as shown in Figure 17.



dm2e*

rss
Add page to e-Shelf

9 Results for dm2e scope   Sorted by: Date-newest

Refined by: topic: Belletristik × creator: Theodor Fontane ×

Refine My Results

Topic
Roman (4)
Prosa (4)
Reiseliteratur (4)
Lyrik (1)
More options

Decade
1850 (1)
1860 (2)
1870 (1)
1880 (3)
1890 (2)
More options

Printed at
Berlin (7)

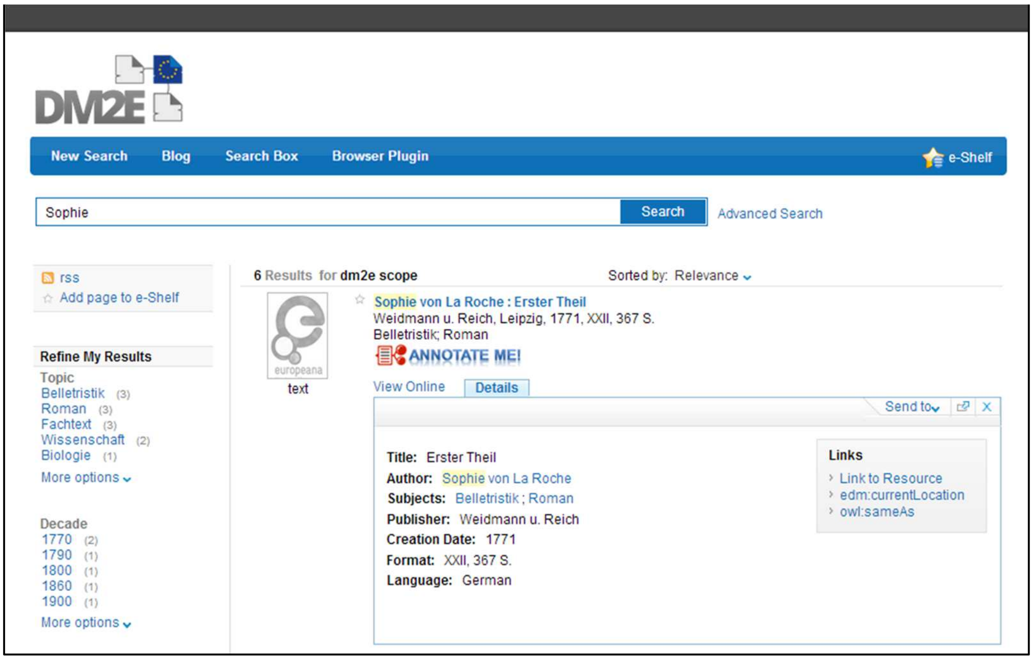
Theodor Fontane : Der Stechlin
Fontane, Berlin, 1899, 2 Bl., 517 S.
Roman; Belletristik
ANNOTATE ME!
text
View Online Details

Theodor Fontane : Roman
Fontane, Berlin, 1896, 2 Bl., 520 S.
Roman; Belletristik
ANNOTATE ME!
text
View Online Details

Theodor Fontane : Irrungen, Wirrungen
Steffens, Leipzig, 1888, 284 S.
Roman; Belletristik
ANNOTATE ME!
text
View Online Details

Figure 17: Faceted search.

The resource view shows a single manuscript (Figure 18), with direct links to the Pundit tool targeting this specific CHO.



Sophie

rss
Add page to e-Shelf

6 Results for dm2e scope   Sorted by: Relevance

Refine My Results

Topic
Belletristik (3)
Roman (3)
Fachtext (3)
Wissenschaft (2)
Biologie (1)
More options

Decade
1770 (2)
1790 (1)
1800 (1)
1860 (1)
1900 (1)
More options

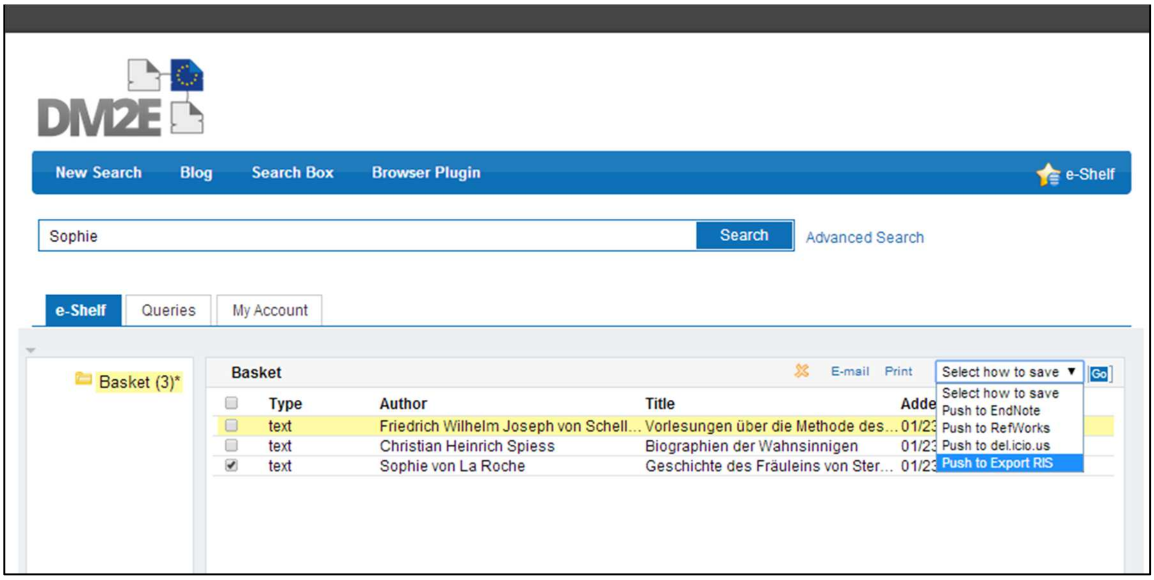
Sophie von La Roche : Erster Theil
Weidmann u. Reich, Leipzig, 1771, XXII, 367 S.
Belletristik; Roman
ANNOTATE ME!
text
View Online Details

Title: Erster Theil
Author: Sophie von La Roche
Subjects: Belletristik ; Roman
Publisher: Weidmann u. Reich
Creation Date: 1771
Format: XXII, 367 S.
Language: German

Links
- Link to Resource
- edm:currentLocation
- owl:sameAs

Figure 18: Record in the Primo Frontend.

Users can bookmark results that are of interest to them using the E-Shelf (Figure 19) functionality, which has the added benefit to allow users to export a selected set of manuscripts to a reference/citation manager or simply mail or print out the metadata. Finally, the search interface provides a variety of hyperlinks derived from the data, such as the landing page in the HTML version of the Linked Data API, the homepage of the data provider and last but not least the manuscript itself within the Web interface of the data provider.



Sophie

□	Type	Author	Title	Adde
□	text	Friedrich Wilhelm Joseph von Schell…	Vorlesungen über die Methode des…	01/23
□	text	Christian Heinrich Spiess	Biographien der Wahnsinnigen	01/23
☑	text	Sophie von La Roche	Geschichte des Fräuleins von Ster…	01/23

Figure 19: Saved records in the E-Shelf.

The search interface is currently available at http://bit.ly/1dUOnly.

10 Documentation

The WP2 outcome, tools and vocabularies, is documented in order to facilitate the tools usage for casual users as well as the reuse of the technical infrastructure for developers. As these audiences differ considerably and have thus different demands on the documentation, the documentation is divided into two main sections. There is on the one hand the user documentation, which includes among others tool guidelines, a user wiki written in an informal style and an OmNom screencast, and on the other hand a developer documentation including the documentation of the OmNom source code, the server documentation and the vocabulary specifications. The full documentation at the current state is as follows:

User documentation

- OmNom introduction and user guidelines in English
- OmNom introduction and user guidelines in German
- OmNom walkthrough
- OmNom screencast

Developer documentation

- Documentation of the OmNom source code
- System documentation
- Vocabulary specifications: DM2E model, OmNom vocabulary and OmNom types

An overview of the OmNom user documentation can be found in the DM2E wiki: http://wiki.dm2e.eu/Main_Page. The documentation parts are disseminated as part of WP4's Task 4.3 "Community documentation". The main idea or general concept of OmNom and its components is described on the dm2e.eu website: http://dm2e.eu/open-workflows/. The aim of this description is to give an access point to the documentation with some further information on what OmNom is how it is used in DM2E.

10.1 User Documentation

The OmNom user documentation in English is created in a wiki as a living document. It includes various user guidelines which should especially help users that use OmNom for the first time. The wiki does also point to other components that are part of the user documentation but are coming from other sources. An overview on OmNom and its components is also included, but the focus lays on step-by-step tutorials that help the user directly working with OmNom. The wiki is structured as follows:

- OmNom-introduction
- Overview of the OmNom components
- Workflows in OmNom
- Guidelines:
  - How to use general functions
  - How to upload and manage files
  - How to create and edit workflows
  - How to configure workflows
  - How to configure and run job

It was chosen to create a wiki as this can easily be extended by others in the project and also by users from outside the project. Having a dynamic way of documentation is also very important as OmNom may be further refined. Adaptations that are made in the system can easily be changed in the wiki. The parts "OmNom-introduction", "Overview of the OmNom components" and "Workflows in OmNom" give a more theoretical insight in how the tool works and what the main idea behind the tool is whereas the user guidelines, the how-to's, give more practical assistance in explaining how the user can proceed diverse tasks in

OmNom. The guidelines include step-by-step instructions to keep it as simple as possible. The OmNom user documentation is also translated in German and can also be accessed via the main page of the DM2E wiki.

Additional parts of the user documentation coming from other WPs are a complete OmNom Walkthrough in form of a pdf-file (WP1) and an OmNom video tutorial (WP4) which is based on the walkthrough. Both, the walkthrough and the video, aim at helping the user that is new to OmNom to use the tool for a complete ingestion process. A FAQ section is planned and will be added to the wiki as soon as the OmNom questionnaires that WP1 has developed are analysed. Questions that may come from users from outside the project can of course be added to the FAQs as well.

10.2 Developer Documentation

The developer documentation is centring around the OmNom tool description on source code level, the model specifications and the server documentation. The OmNom tool documentation for developers is directly in the source code but can also be browsed in a more user friendly way in an HTML representation of these Javadocs. The link to the source code documentation is also listed in the wiki.

Configurations that were made in order to deploy the various tools and services that are used in the ingestion infrastructure are documented as well. An overview of the deployment is also part of this deliverable and includes brief descriptions of the OmNom, server and Silk deployments. The server and tool deployments are explained more detailed and also written as a living document in the wiki as the configurations can still change.

Specifications of the DM2E model, the OmNom vocabulary and the OmNom-Types vocabulary can directly be accessed via their namespaces URIs:

- DM2E model: http://onto.dm2e.eu/schemas/dm2e/
- OmNom vocabulary: http://onto.dm2e.eu/omnom/
- OmNom-Types: http://onto.dm2e.eu/omnom-types/

The URIs opened in an html browser lead directly to the representation of the models on Neologism[26]. Here, the whole model as well as the individual properties and classes of the respective models are displayed. Every of these resources has its own definition. In addition, the DM2E model is described more detailed in a textual model description. This was especially needed as a help for data providers in the scope of the project. The latest published specification as well as the model for further (local) usage in an OWL-file can be accessed via the documentation wiki. Intermediate model versions can be found on the projects internal wiki on Redmine. As soon as a new model version is stabilised, it will appear in the open DM2E wiki and also be available from outside the project.

[26] Neologism is an open source ontology publishing platform based on Drupal. More information can be found on the Neologism website: http://neologism.deri.ie/ (05.02.2014).

11 Deployment

The development and deployment infrastructure has been moved from the former test server at the University of Mannheim to the DM2E server at the Humboldt-Universität zu Berlin. For all components of the DM2E infrastructure, separate virtual machines have been set up.

The following table lists all Virtual Machines (VMs):

Name	IP Address	Domain	OS	Services	RAM/HDD
Josso	141.20.126.231	josso.dm2e.eu	Debian 7.3.0	User Authenticatio n for OmNom	2GB/25GB
Data	141.20.126.232	data.dm2e.eu	Ubuntu 12.04.3	Pubby OWLIM	16GB/500GB
OmNom	141.20.126.233	omnom.dm2e.eu	Debian 7.3.0	OmNom	10GB/300GB
Doc	141.20.126.234	doc.dm2e.eu	Ubuntu 12.04.3	Redmine EtherPad	2GB/28GB
Context	141.20.126.235	context.dm2e.eu	Ubuntu 12.04.3	Silk Framework	16GB/32GB
Pundit	141.20.126.236	pundit.dm2e.eu		Pundit Ask Feed	8GB/25GB
Onto	141.20.126.237	onto.dm2e.eu	Ubuntu 12.04.3	Neologism	2GB/25GB

Table 3: All VMs used in DM2E with their addresses, domains, operating systems, services and RAM.

Information and documentation about deployment of all components used in DM2E can be found in the documentation, which is constantly updated. Wherever possible, existing packages of the tools are used. Where these are not available or would need considerable adjustments, own packages will be created. For a convenient quick start, we will additionally provide clean virtual machine images at the end of the project.

12 Conclusion and Next Steps

With the final version of the interoperability infrastructure, DM2E has reached an important intermediate goal. In this deliverable, we gave a broad overview of the many different components that in the end form this infrastructure. The general concepts of the infrastructure, as well as most of its components are now relatively stable. It has, however, to be stated that still, with every ingestion and every new content provider, new challenges arise and new issues that have to be addressed by means of adjustments in the DM2E model or in one or more of the other components and interfaces. We do not consider this as problem. Ultimately, the experience gained from every ingestion is preserved in these changes and from the increasing stability, we gain confidence, that DM2E investigates the domain of Digitised Manuscripts and their provision for scholars in a distributed work environment very carefully and completely. DM2E is funded for one more year and we will continue to update and improve all aspects of the infrastructure for the maintenance release that is planned for the end of the project.