Semantic Web 0 (2012) 1–0
IOS Press

An information model for managing resources and their metadata

Editor(s): Krzysztof Janowicz, University of California, Santa Barbara, USA
Solicited review(s): Sven Schade, European Commission – Joint Research Centre, Italy; Tudor Groza, The University of Queensland, Australia; Paul Groth, VU Amsterdam, The Netherlands

Hannes Ebner [a,b,*] and Matthias Palmér [a,b]
[a] *KTH Royal Institute of Technology, 100 44 Stockholm, Sweden*
E-mail: {hebner,matthias}@kth.se
[b] *MetaSolutions AB, 133 31 Saltsjöbaden, Sweden*
E-mail: {hannes,matthias}@metasolutions.se

Abstract Today information is managed within increasingly complicated Web applications which often rely on similar information models. Finding a reusable and sufficiently generic information model for managing resources and their metadata would greatly simplify development of Web applications. This article presents such an information model, namely the Resource and Metadata Management Model (ReM^3). The information model builds upon Web architecture and standards, more specifically the Linked Data principles when managing resources together with their metadata. It allows to express relations between metadata and to keep track of provenance and access control. In addition to this information model, the architecture of the reference implementation is described along with a Web application that builds upon it. To show the taken approach in practice, several real-world examples are presented as showcases. The information model and its reference implementation have been evaluated from several perspectives, such as the suitability for resource annotation, a preliminary scalability analysis and the adoption in a number of projects. This evaluation in various complementary dimensions shows that ReM^3 has been successfully applied in practice and can be considered a serious alternative when developing Web applications where resources are managed along with their metadata.

Keywords: information model, metadata management, resource annotation, linked data, provenance

1. Introduction

Most libraries and educational institutions manage their content in repositories, where small groups of domain experts annotate and manage resources and their metadata. Such repositories are homogeneous within the respective institution that also has the authority to decide upon which standards to support and what level of interoperability to strive for. This poses a problem in a heterogeneous landscape where interoperability between repositories is sought after. An appropriate information model is needed to be able to cope with different metadata standards, separated management of resources and their metadata, and transfer and enrichment of metadata across systems.

The research described in this article was carried out within projects that focused on publication, enrichment and management of heterogeneous metadata. The common denominator in all of these projects was the annotation of resources with metadata and the collection and enrichment of already existing metadata originating from a multitude of content repositories. The heterogeneity of the metadata requires the capability of handling arbitrary formats, such as established standards and their variations, or also proprietary formats.

*Corresponding author. E-mail: hebner@kth.se.

The following requirements were relevant for several content-heavy projects (such as the Organic.Edunet project [24]), where it was necessary to

- manage resources together with their metadata,
- handle a wide variety of different metadata expressions,
- support Web technologies to enable modern Web applications, and
- to provide a unified approach for the integration of metadata from legacy (non-Web) systems.

The requirements above led to a set of specific general problems that needed to be addressed in the course of developing an appropriate information model.

1.1. Problem statement

The general problems are summarized in the following paragraphs which consist of the questions that form the corner stones of the information model.

Management of resources and their metadata. How are situations distinguished where either both resource and metadata, only metadata, or neither metadata nor resource are in the system?

Enrichment of metadata. Adding domain- or subject-specific metadata in addition to generic metadata is a primary use case in the projects in which the information model is being used. How can metadata be enriched while keeping different descriptions separate?

Organization of metadata. Metadata harvesting does not come with a built-in mechanism that connects different metadata about the same resource. What is needed to maintain the original metadata and to keep track of enrichments?

Integration of heterogeneous information sources. Metadata expressed in different models and standards should be used together, e.g. generic metadata in connection with domain-specific educational metadata. What is required from the information model that manages these metadata in a common carrier (see 2.2)?

Support for Web architecture. How should an information model for managing resources and their metadata look like if it is to be used in Web applications? A prerequisite is the support for Web architecture and standards, in particular Linked Data.

The concept of named graphs *named graphs*, and particularly the use of the Resource Description Framework (RDF) have already been suggested as a partial solution. However, even though named graphs are loosely specified in different articles and standards such as [4,31,18], they lack clear guidelines for how they should be used in the context of Web architecture. In addition to answers to the questions above, the described information model and its reference implementation sought after best practices for how to:

- express that named graphs are related, for instance if they identify metadata that describe the same resource.
- retrieve and modify named graphs using a standard protocol such as HTTP.
- keep track of provenance and access control of named graphs and resources described within them.

To solve the problems as stated above, this article introduces an information model together with a reference implementation. They can be used to manage resources and their metadata, to express relations between metadata, and to handle provenance and access control. The described approaches are intended to provide solutions that make it possible to bring already existing metadata into the world of Web standards. Such resources and metadata are then uniquely identifiable, accessible and modifiable using HTTP URIs and REST-based services following the Linked Data principles. In addition, a short summary of several showcases is presented, including some conclusions regarding the general applicability and possible future applications. The information model and its reference implementation have been evaluated from several perspectives, such as the suitability for resource annotation, a preliminary scalability analysis and the adoption in a number of projects.

1.2. Important terms

Several terms have a frequent occurrence in this article. The most important ones are explained in this section in order to disambiguate their meaning in the context of the work described here.

Uniform resource identifier and resource. This article uses the same definitions of the terms *uniform resource identifier* (URI) and *resource* as they are provided in the *Architecture of the World Wide Web* [21] in section 2 *Identification*, which is:

> *By design a URI identifies one resource. We do not limit the scope of what might be a resource. The term "resource" is used in a general sense for*

> *whatever might be identified by a URI. It is conventional on the hypertext Web to describe Web pages, images, product catalogs, etc. as "resources". The distinguishing characteristic of these resources is that all of their essential characteristics can be conveyed in a message. We identify this set as "information resources".*

Metadata. A common and widely accepted definition of *metadata* is that it is "data about data", see also the considerations in [1] where it is defined with "Metadata is machine understandable information about web resources or other things". In addition, the term *meta-metadata* is of relevance for the information model described here, so the axioms "metadata is data" and "metadata can describe metadata" are important to mention.

Resource annotation. When metadata is created or modified in order to describe a resource than we call this *resource annotation*. More specifically, the meaning of annotation in this article is that "metadata about one document can occur within a separate document which may be transferred accompanying the document" [1]. The term *document* in this definition is equivalent to *resource*.

Repository. A *repository* is a server from which resources and metadata can be retrieved using a standard protocol such as HTTP.

Harvesting. Metadata harvesting is used to retrieve metadata "records" from one or more repositories into another and can be used to build large collections of metadata. The harvesting process is usually carried out using the "Open Archives Initiative Protocol for Metadata Harvesting" [22]. It uses XML over HTTP and requires as a minimum *Dublin Core* (DC) metadata [6,7], but other representations may be used in addition to DC.

Provenance. As discussed in the introduction of the *PROV Model Primer* [19], there are different uses of *provenance*. The information model described here makes use of both *agent-centered provenance* and *object-centered provenance* which is described in 3.3.

1.3. Structure of this article

This article is organized as follows. Section 2 gives an account of the relevant state of the art for metadata management and resource annotation. The information model is introduced in section 3 which is followed by a presentation of the reference implementation in section 4. A Web application which implements a user interface to the reference implementation is described in section 5. This is followed by some showcases in section 6. The evaluations in section 7 discuss scalability, the applicability for resource annotation, and the adoption in real applications. The conclusions in section 8 summarize the work carried out and depict how the problems that have been stated in section 1 were solved. The arcticle concludes with the planned next steps and possible future work in section 9.

2. State of the art

This section briefly analyzes and summarizes the state of the art which is of relevance for the research described in this article.

2.1. Document- vs. graph-centric metadata

Traditional ways of annotating resources often take a document-centric approach and use the XML format as it is an established standard for expressing information. Unfortunately, when document-centric metadata are transferred between systems (e.g. using a harvesting protocol like OAI-PMH [22]), the metadata is copied and a fork takes place. The alternative, to reuse metadata without making a copy, requires that the original instance can be uniquely identified. This is most often not possible with the current approach of metadata repositories as everything is based on harvesting metadata from one system into another, leading to copies and forks instead of references. Information is unnecessarily duplicated and numerous variations of descriptions of the same resource are created without being able to reconstruct their history.

2.2. RDF as common carrier for metadata

To be able to create flexible annotations of resources it is necessary to use a data model which is designed to allow multiple metadata expressions following different standards to coexist. RDF is such a (meta) data model [23]. However, expressing metadata in RDF requires a thorough mapping to be crafted, which often involves an analysis of the exact semantics of the standard. Good knowledge of RDF and related standards is required as it is good practice to reuse established terms from other RDF-based vocabularies whenever possible. There are situations where the conceptual

model cannot be cleanly mapped to the RDF model and information may be lost. To avoid such situations, RDF should be considered as a basis for metadata interoperability - a common carrier - when adapting existing or creating new metadata standards. For a longer discussion on this subject see [28].

The most important metadata standards in the context of this article are *Dublin Core metadata* [6,7], its abstract model (DCAM) [30] and *IEEE Learning Object Metadata* (LOM) [8]. They are used within the showcases described in section 6. IEEE LOM is mostly used in its draft mapping into the DCAM to be able to store it in RDF.

2.3. Named graphs to manage sets of triples and provenance

The Semantic Web [3] allows statements about identifiable resources to be expressed using RDF triples which may also be made available on the Web for others to discover. When new statements are made, there is no need to duplicate information. Additional statements about the same identifiable resources can be expressed as new RDF triples and be published on the Web separately from the first set of triples. If all available triples describing the same resource are merged into a single big graph, a holistic view about a resource can be constructed. With only triples as the source of this information it is impossible to identify triples or sets of them, which creates several problems. To mention only a few, it is difficult to detect which triples have been replaced in more recent revisions, it is hard to keep track of the history of a resource's descriptions, and it is almost impossible to provide information which depends on its purpose (i.e. contextualization).

The concept of named graphs (NG) [4,18], enables us to work around this, by being able to uniquely identify sets of triples with URIs. This generic approach should be compared to how specific metadata standards have solved the same problem, for instance IEEE LOM [8] with its Metametadata identifier expressed in XML.

Another issue is related to searching and indexing. If a query matches one or more triples it is unclear where those triples originate from and in which context they express information about the described resource. This can be partially solved by using NGs as this allows for uniquely identifying the relevant triples. The use of named graphs in this article is basically identical to how they are treated by the SPARQL query language [31].

Named graphs provide an approach for denoting collections of triples which are annotated with relevant provenance information. However, in [26] it is mentioned that most approaches building on named graphs for provenance lack a clear specification of how provenance should be represented.

2.4. Representational state transfer

Representational State Transfer (REST) [17] is an architectural style for distributed hypermedia systems and is a popular design pattern used for resource based web services. REST itself is protocol-agnostic, but in this article it is used in the context of HTTP. Its architectural elements are resource identifiers, resources, resource representations and their metadata. RDF on the other hand only provides resource descriptions via resource identifiers without any knowledge of how to access those resources via resource representations. However, when named graphs are given URIs they are effectively resources that contain sets of triples and it is quite natural that they should have resource representations which makes REST a logical choice to access RDF-based systems.

"Pure" REST is difficult to achieve and most of the offered REST-ful web services are REST-oriented but also contain other concepts such as RPC-oriented methods [36]. An implementation taking advantage of HTTP makes it easier to align with the Linked Data principles as described below.

2.5. Linked Data

Linked Data (LD) is a recommended best practice for interlinking and exploiting data. This e.g. enables the exploration the *Web of Data* which is constructed by related documents on the web. The focus lies on links between uniquely identifiable *things* described using RDF. The term "thing" as used in the Linked Data rules is equivalent to the term "resource" in this article. Linked Data implements the following four rules [2]:

1. Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL) [23,31].

4. Include links to other URIs so that they can discover more things.

LD suggests the use of URIs, HTTP and RDF, which makes it more specific than REST. However, RESTful web services can operate on the Web of Data when the offered data conforms to the Linked Data principles. The growing LOD cloud [5] is easily extended by simply providing statements which link to the existing published datasets. This is also one of the big differences to traditional repositories: instead of harvesting and copying data, it is sufficient to refer to things, lookup identifiers and fetch from the original source. To improve performance the data can be cached, but this does not affect the basic principles. The mainstream of learning repositories [28] has not arrived in the LOD cloud yet and it will be necessary to provide a "bridge" between these two worlds. How this can work is also topic of this article and described further down.

2.6. Access control and RDF

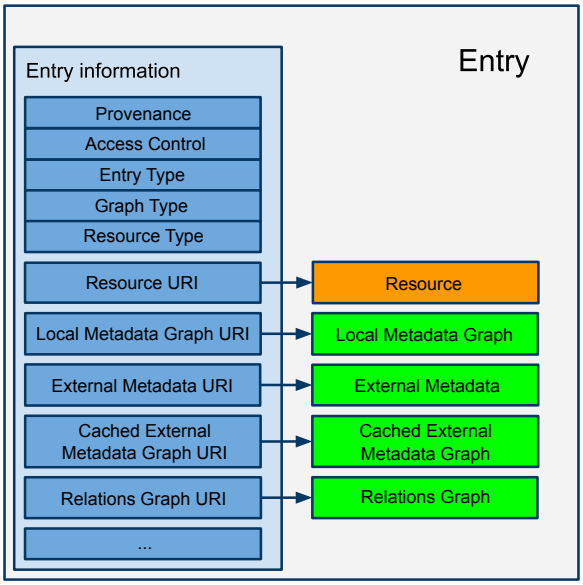
Web Access Control (WAC) as described in [39] is an existing decentralized system that makes use of an ontology for access control on the Web with several implementations. Users and groups are identified by HTTP URIs and the model allows various forms of access to resources. In addition to URIs, users are identified by WebIDs as specified in [34]. The URIs are dereferencable, which means that users and groups can be looked up across systems and given access to resources even if they do not exist in the local system. How access control is realized in ReM^3 is discussed in 3.4.

3. An information model for managing resources and their metadata

This section describes the Resource and Metadata Management Model (ReM^3) by first providing a conceptual overview and then explaining in detail how various aspects such as provenance and access control are expressed inside the model. The information model is based on and relates to the state of the art as described in the previous section.

3.1. Conceptual overview of ReM^3

ReM^3 is an information model for keeping track of resources and their metadata. It is based on the concepts of *contexts* and *entries* where each context manages a set of entries. A context is a container for a set of entries that are managed together, at a minimum it provides default ownership of the contained entries. An entry contains a resource, descriptive metadata about the resource as well as some administrative information of the entry which will be referred to as the entry information. The entry information also keeps track of access control and provenance. Access control can be managed on both context and entry level, depending on how fine-grained access control is needed. The entry also keeps track of relationships from other entries via a special *relations* graph. See figure 1 for a conceptual representation of a ReM^3 entry.



The cardinalities between the entities above are 1:1.

Figure 1. ReM^3 Entry and its linked information

Each entry has three different kinds of types that determine where the resource and its metadata reside, and how the resource is represented and how it should be treated:

- *Entry Type* indicates if neither, one, or both of the entry's resource and metadata is maintained within the local system. This is the most important type for the showcases shown in section 6 as it differentiates between local and external (remote and harvested) resources.
- *Resource Type* tells whether a resource has a digital representation or not.

– *Graph Type* indicates whether a resource gets a special treatment within the implementation of the model.

Ideally, the entry information, that is, the information about an entry, is represented in a single RDF graph which can be requested and updated as a whole or in part. If an application needs additional information about a resource, it can be represented in the same RDF graph by adding additional properties. Within the entry information, the resource, the describing metadata graphs and the relation graph are URIs that are detectable via special properties from the entry URI. The URIs for the metadata, the relation graph and sometimes the resource (in case the resource is an RDF graph) point to named graphs.

However, the availability of named graphs for an entry also depends on the entry type which indicates if metadata and the resource is to be found locally or externally. More specifically, the possible values for the entry type are as follows (see also figure 2).

– *Local* - both metadata and resource are maintained in the entry's context.
– *Link* - the metadata, but not the resource, are maintained in the entry's context.
– *Reference* - the resource and the metadata are maintained outside the entry's context.
– *Link reference* - the resource and the metadata are maintained outside the entry's context, in addition there are complementary metadata maintained in the entry's context.

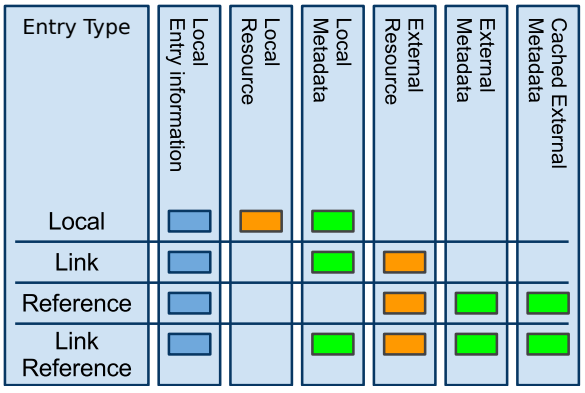


Figure 2. The ReM^3 Entry Type and its implications for the location of metadata

Whenever there are metadata maintained outside of the entry's context it may be cached locally (which makes it *cached external metadata*) to increase reliability and performance, and to avoid pushing the responsibility of doing metadata format transformations to application developers. The entry information is always kept in the corresponding context, independently from the used entry type.

The resource type indicates to which extent a resource is an *information resource*. A resource is anything that can be identified by a URI whereas an information resource is a resource whose essential characteristics can be conveyed in a message. Examples are documents, images, videos, etc., of various sorts which have representations, e.g. HTML, ODT, PNG, etc., which can transferred in a message body which is the result of an HTTP request. The idea behind the resource type is based on the Architecture of the World Wide Web [21], the W3C TAG discussions on HTTP dereferencing [38] and the W3C Interest Group Note on "Cool URIs" [32].

The two possible values for the resource type are:

– *Information resource* - resource has a representation, in the repository or elsewhere.
– *Named resource* - The resource is not an information resource, the resource can be referred to in communication but not transferred in a message.

The graph type was introduced to be able to easily recognize resources which need special treatment by the implementation. Examples are the graph types used for access control, namely *User* and *Group*; *Context* for container entries, and *List* to indicate an ordered list of entries within a context.

The ReM^3 terms as shown in figure 3 have been formalized as an RDF schema [11] and are described in the ReM^3 specification [12].

3.2. Named graphs in ReM^3

The information model is RDF-oriented and relies on the concept of named graphs which is part of the SPARQL protocol and the query language specification. The formal semantics of named graphs are described by Carroll et al. in [4] and also the SPARQL specification [31]. As every NG is identified by a URI, it is possible to keep track of the NG provenances through the entry information as described above. The entry information contains expressions that describe the relationships between graphs. This is used to express that NGs are related, as it is the case when the same resource is described in different contexts. In the case of ReM^3, NGs are used for expressing the following pieces of information:

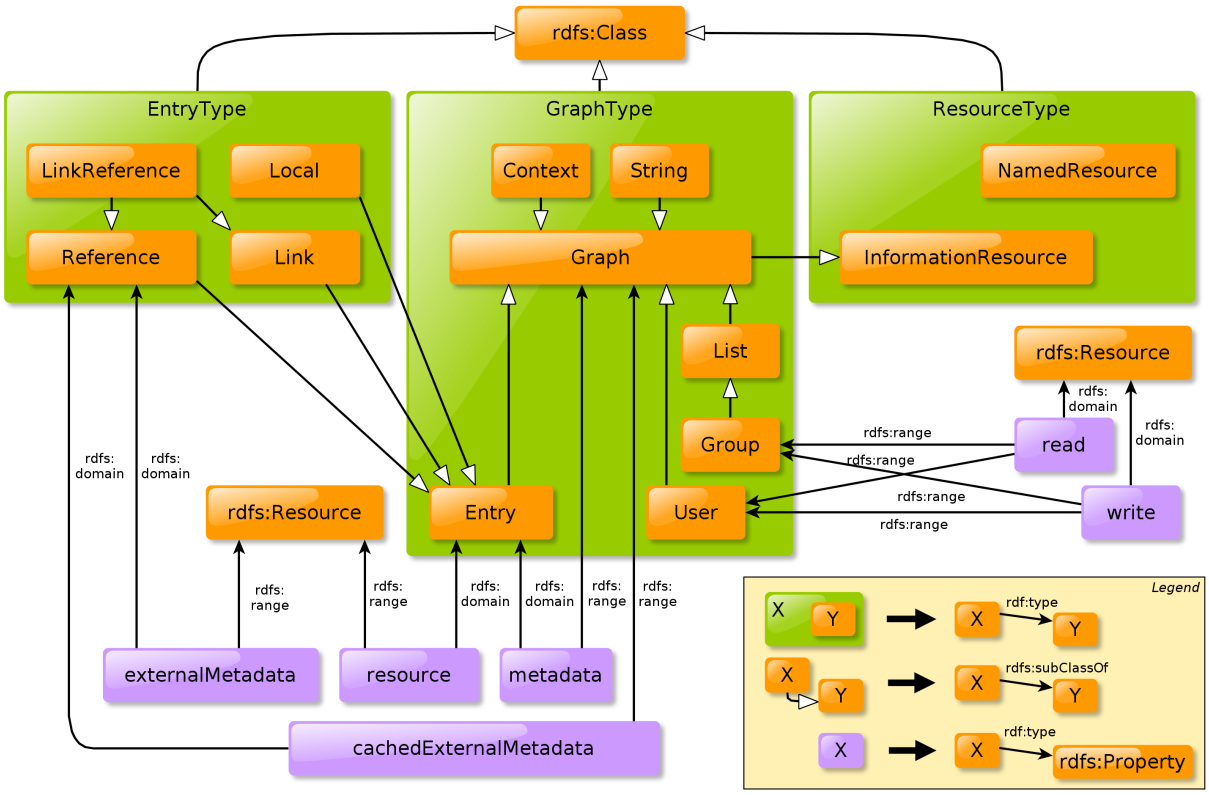


Figure 3. ReM^3 terms described in an RDF schema

– The entry itself, this is the "main" NG where all other NGs belonging to the same entry are linked together. This effectively makes it the meta-metadata.
– Metadata, if present.
– Cached external metadata, if present.
– Resource, if the resource can be expressed in RDF.

In addition to being the foundation for the ReM^3, the use of NGs makes contextualisation of metadata possible. Without naming the graphs it would be hard to differentiate between triples originating from different sources.

3.3. *Expressing provenance in ReM^3*

ReM^3 supports *agent-centered provenance*, which means that it keeps track of information about which users were involved in creating or modifying information, and *object-centered provenance*, that is, keeping track of the origins of a resource or its metadata. The same terminology and definitions are also used in the *PROV Model Primer* [19]. However, the origins of ReM^3 date back to a time (see [10]) when the PROV Model did not exist, so it could not be taken into consideration when ReM^3 was implemented.

ReM^3 keeps track of who created or contributed what, when, where and perhaps even why. All these pieces of information are kept in the entry information and are available if the resource originates from a local ReM^3-based repository. The following provenance-related properties are a minimum for being able to keep track of annotation cases where both local and external metadata are involved, i.e. entries with entry type Link Reference:

– Creator and contributor
– Creation and modification date
– Reference to the resource
– Reference to the external (possibly original) metadata
– Date when the external metadata was cached

Provenance information can be both metadata and meta-metadata, e.g. when keeping track of the origin of a resource it is metadata, while it becomes meta-

metadata when it is used to express similar information for metadata. Restrictions apply if the metadata originates from an external system, i.e. provenance for the resource and metadata is only known if this is managed and exposed by the system where the information is fetched from. If this is not the case then the "provenance trail" starts at the time the external metadata is cached in the ReM^3 system.

One of the currently existing restrictions of the model is the lack of revisions and versioning, both for the metadata and the described resource. There is previous work which can used in this context [37] which is being considered for revisions of this information model.

The information above about provenance in ReM^3 is about what the model supports in its entry information. In addition it is possible to add any information to the metadata or meta-metadata graphs, even if ReM^3 is unable to interpret it directly. Currently it is also explored to which extent the PROV Model [19] can be used.

3.4. Expressing access control in ReM^3

Just as provenance is expressed in the entry information, so is access control. The purpose of the access control in ReM^3 is to control who has rights to access entries. Access to the entry, metadata and resource is determined by specific ACL statements using the URIs of the entry, metadata and and the resource URI, respectively. The access control information for the resource is only relevant when it can be enforced by an implementation, i.e. if the resource is located in the same system (entry type is local). Similarly, access control for metadata is only relevant when it is in the same system (when entry type is local, link or link reference but not reference).

Access control information is expressed as a set of read and write permissions for users and groups on the entry, the metadata and the resource. Any explicit permission given on entry level automatically applies to the resource and metadata and does not need not be repeated. An exception is that by default anyone has read access to the entry information, but not to the resource or metadata. Anyone who has been given write access to a entry is considered to be an owner of that entry.

Contexts are also represented as entries. Access control for a context, expressed on its entry, has a special meaning with regard to all entries located in that context:

- Permissions given for the metadata of a context has no effect on the entries in the context.
- Permissions given for the resource of a context applies to all entries in the context who lack own access control. I.e., if an entry holds ACL information then those permissions override any permissions inherited from the contexts resource.
- Ownership of a context (write permissions on the entry level of a context) implies ownership of all entries in the context regardless of any access control specified on them.

Users and groups that can be given permissions are represented as entries with the special graph type User and Group respectively. There are two default users and two default groups. First, "_guest" represents any user that has not authenticated himself while "_users" is the group of all users that can authenticate themselves in the system. Second, "_admin" is a predefined superuser and "_admins" is a group to which users that should have superuser privileges can be added.

There are a two special rules with regard to lists, that is, entries with graph type list:

- Entries which are created as children of a list with custom ACL automatically inherit permissions from that list.
- An entry that belongs to a single list cannot be removed from that list (making it "unlisted") without also removing the entry itself unless the user has write permissions in the context.

Web Access Control (WAC) as summarized in 2.6 has not been implemented because the authors decided to start with a more light-weight and non-distributed access control model. WAC in its current form is limited to information resources, whereas ReM^3 differentiates between resource and metadata. However, the WAC ontology [39] and WebID [34] are being considered for integration into ReM^3 in the future.

4. Exposing ReM^3 using Web technologies

This section describes how ReM^3 can be accessed using standard Web technologies. The reference implementation and its reliance on Web architecture are summarized and implications for interopability are explained.

4.1. Reference implementation

The research questions stated in the beginning were the main driver behind developing an own framework as described in [10]. It should make it possible to manage data and its metadata in an interoperable and conceptually clean way, being compatible with traditional data sources and the possibility of using Semantic Web technologies and linking data at the same time. The information structure of ReM^3 is suitable to be exposed as a REST-ful HTTP API, see also 2.4. The described work resulted in a reference implementation called *EntryStore* [16] on which the following sections focus. The framework is built on top of the quadruple store OpenRDF Sesame[1], making it possible to identify sets of triples using named graphs as mentioned above.

4.2. REST-based interface

There are three basic kinds of REST resources in a context: resource, metadata, and entry. There are two additional kinds of resources, the relations resource that contains relations from other entries, as well as the cached-external-metadata resource that contains a cache of the external metadata if the entry type is reference or link reference.

The pattern below shows the URIs and allowed HTTP operations for the multiple kinds of REST resources:
{http-verb} {base-uri}/{context-id}/{kind}/{entry-id}

- *http-verb* is one of GET, PUT, POST or DELETE.
- *base-uri* is the base URI (namespace) that is specific for each system.
- *context-id* is a unique identifier for a context.
- *kind* is one of the kinds of REST resources.
- *entry-id* is a identifier for an entry that must be unique within each context.

Providing an easy-to-use and REST-oriented interface together with ReM^3 allows for enrichment of metadata as the protocol makes communication in both ways possible. Resources in other systems can be described by linking to them and building a connection between the metadata and the resource. Such connections are in turn exposed using Linked Data which integrates heterogeneous information sources. The HTTP API of EntryStore is only summarized here, a more detailed description can be found in an earlier paper [10].

[1] OpenRDF Sesame, `http://www.openrdf.org`

4.3. The use of Linked Data

As indicated in 2.5, ReM^3 can be used to build a bridge between traditional repositories and the Linked Data cloud. The main point for linking information in ReM^3 is the entry, but also lists are used to build indirect relations. RDF triples in the entry information graph are used to link entries, resources and their metadata together. All involved entities are identified by dereferencable URIs whenever possible and HTTP is the standard protocol.

An EntryStore repository can also be queried through a SPARQL endpoint. The ACL model of ReM^3 limits which metadata can be exposed. The SPARQL protocol does not support any access control, so this had to be solved on the level of the repository by exposing only public metadata. Other metadata, no matter whether completely private to the creator or restricted to groups, is not exposed at all through SPARQL. There are endpoints on two different levels:

1. A global endpoint for the whole EntryStore repository, including all contexts and their entries.
2. An endpoint per context, including all entries of a context. This allows to restrict queries to a limited amount of entries and speeds up queries.

Information about named graphs is also exposed using the GRAPH keyword which allows to create views of contextualized resource metadata in SPARQL query results.

4.4. Additional interfaces

EntryStore also has support for additional protocols, mainly aimed for harvesting and querying, such as OAI-PMH [22] and SQI [33]. EntryStore supports both directions, that is, querying and harvesting other systems as well as being queried and harvested itself. The architecture of EntryStore makes it possible to hook in additional protocols if required. The same applies to metadata converters as the infrastructure includes support for mapping metadata to and from RDF.

Legacy standards and protocols are supported to make an integration into already existing repository landscapes possible.

4.5. Interoperability and implementation experiences

The metadata editor in use allows editing of RDF graphs directly and send it to the backend. Dublin

Core-based application profiles (AP) are a natural choice because they map easily into RDF. As an example, to be able to do the same with *Learning Object Metadata* (LOM v1.0)-based profiles, a mapping from LOM to the Dublin Core Abstract Model (DCAM) was necessary. The DCMI developed such a mapping and published a draft in their wiki[2]. On top of that, additional mappings were created to support the LRE v3.0 AP used by the Organic.Edunet project [24,9] which is based on LOM and replaces respectively enhances some vocabularies. Dublin Core terms are (re)used wherever possible, only metadata properties specific to LOM were given an own identifier.

EntryStore supports HTTP content negotiation and performs conversions between metadata formats as needed. It is e.g. possible to send LOM/XML to the server and request RDF for the same metadata graph. The formats differ, but the information is the same due to a careful mapping that balances accuracy against discarding of information that cannot be translated in a good enough manner.

4.6. Free-text queries

There is one exception to the overall good performance: free-text queries on literals. SPARQL queries using FILTER and regular expressions are very expensive. To solve this problem an Apache Solr index[3] is used for searches in metadata literals. EntryStore implements listener interfaces and notifies Solr of events in the repository, which (re-)indexes entries and their metadata as soon as a change is made. This is important to keep the repository and the search index in sync. The combination of SPARQL and Solr queries allows for powerful and efficient searches even in large repositories.

The Solr API is not exposed directly as it would not be possible to respect the ACL. Instead, EntryStore queries Solr internally and sends the result to the client after handling the access rules. Every ReM^3 has one corresponding Solr *document* which includes all necessary fields for free-text searches in both metadata and resource. There is experimental support for full-text search in resources if they contain text and are provided in a common format. Apache Tika[4] is used for this.

[2]DCMI Education Community Wiki, `http://dublincore.org/educationwiki/`

[3]Apache Solr, `http://lucene.apache.org/solr/`

[4]Apache Tika, `http://tika.apache.org`

5. Building Web applications using ReM^3

Since the recommended way to utilize ReM^3 is via its REST-based interface we chose to focus on developing Web applications based on JavaScript, i.e. applications that maintain state on the client side and use a REST-ful approach to retrieve and update data.

There are no hard restrictions on which applications can be built on top of ReM^3, in fact the information model is very generic and it should be possible to use it in a wide variety of applications. Still, certain applications are easier to build than others due to the nature of the information model. This section focuses on an application that more or less directly exposes the capabilities of the ReM^3, namely the *EntryScape* Web application (included in the EntryStore project [16]) which previously was known as *Confolio*. EntryScape is by no means the only or necessarily the best way to expose the capabilities of ReM^3. However, it sufficiently exposes some of the complexity of building user interfaces that make use of the full flexibility of ReM^3.

EntryScape provides portfolios (which can be seen as personal spaces) for individuals and groups. Each portfolio provides a place to store resources - in the form of uploaded files, web content, physical entities or abstract concepts, together with descriptive metadata. A portfolio is represented as a ReM^3 context in EntryStore, and a resource together with its metadata corresponds to a ReM^3 entry. In figure 4 a work view is shown of a portfolio with a listing to the left and details of a selected entry to the right.

The metadata expressions may differ greatly between entries because:

- entries may represent different things, for example web pages or physical objects.
- entries may be described for different purposes and different target groups.
- entries may originate from different information sources which use different standards.

The use of RDF as common carrier allows these metadata expressions to co-exist, both between entries and sometimes within a single metadata expression. This flexibility presents a challenge when presenting and editing metadata since very little can be taken for granted. The solution taken in EntryScape is to rely on the library RForms[5] that generates user interfaces for both presentation and editing of metadata from a con-

[5]RForms is a JavaScript re-implementation of the SHAME Java library

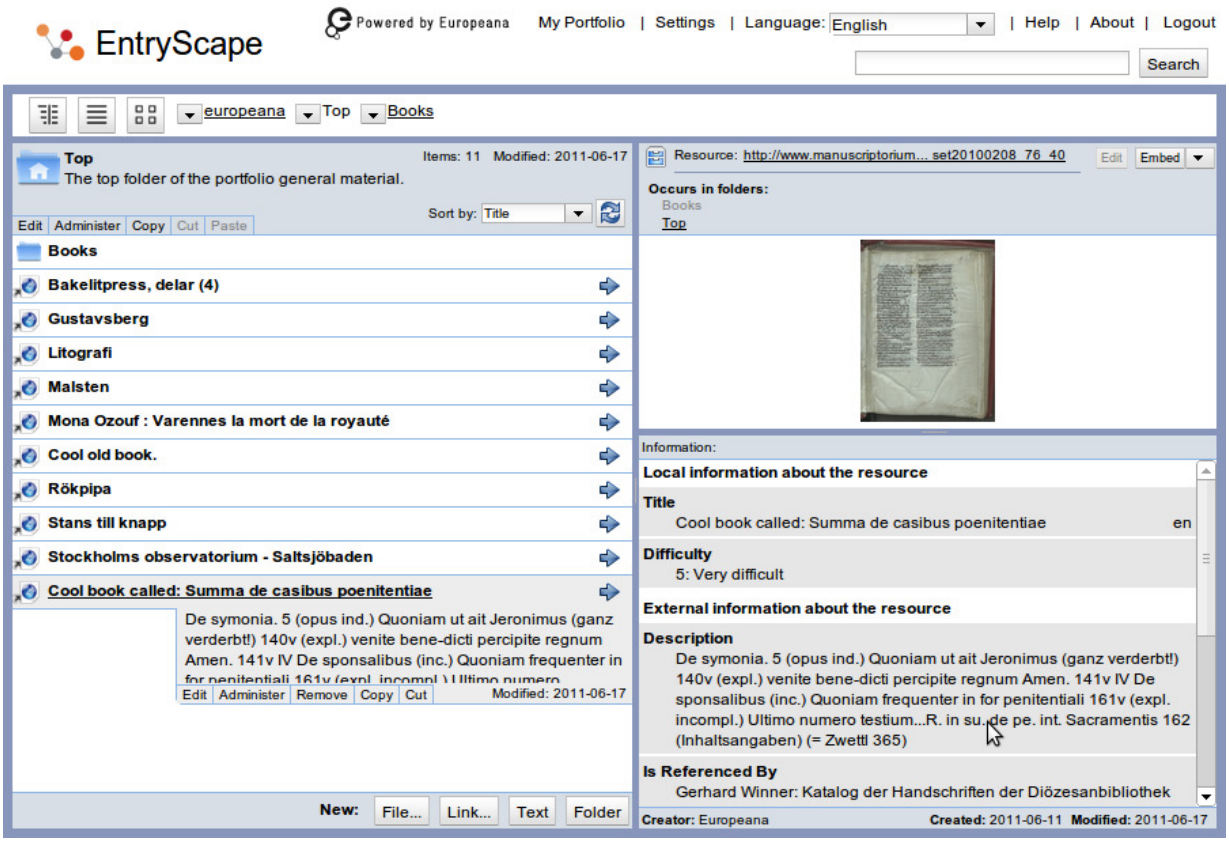


Figure 4. A screenshot of EntryScape which was extended work with the Europeana search API and metadata

figuration mechanism called *annotation profiles* (AP). The details on how RForms and APs are used to transform an RDF graph into a form is beyond the scope of this article and the interested reader is encouraged to look at [15,29,14] for details where also relations to other initiatives such as DCAP DSP [27] are discussed.

To generate an editor, RForms must be told which annotation profile or which combination of APs to use. In theory, the user could be asked which AP to use in each situation given that enough descriptive information is provided to make an informed decision. However, from a usability perspective it is often better to present users with a reasonable default and allow it to be changed into something more specific when needed. Each EntryScape installation may configure a default annotation profile for every entry type it wants to support.

In figure 5 we see basic Dublin Core metadata combined with a copyright statement from IEEE LOM.

In presentation mode the same Annotation Profile will be used, but only fields that have been filled in will be shown. If RForms detects that there are more metadata available than can be shown with the current Annotation Profile, it will look for other Annotation Profiles as a fallback. Such a situation can occur when entries originate from another system, or for that matter, the user has switched back and forth between Application Profiles or intentionally combined them.

6. Showcases

The following showcases are all centered around learning resource descriptions. They involve annotation of resources which are uploaded into or linked from the EntryScape web application as well as enhancement and contextualisation of metadata which is harvested from other repositories.

6.1. Organic.Edunet

The goal of the now successfully completed Organic.Edunet project was to facilitate access, usage and exploitation of digital educational content related to

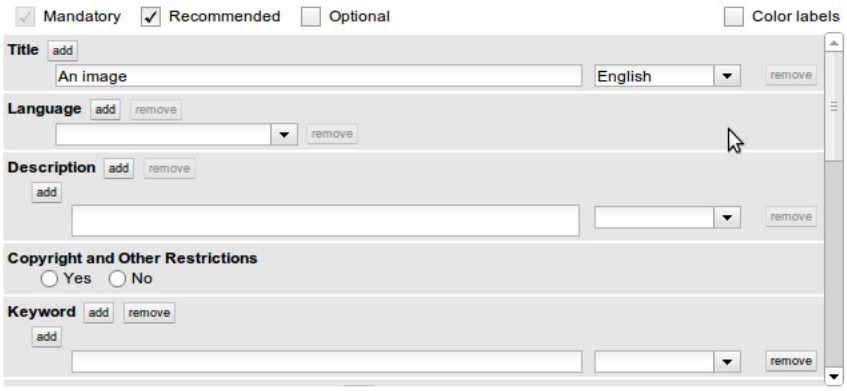


Figure 5. An editable metadata form which blends Dublin Core and LOM metadata

Organic Agriculture and Agroecology. The combination of EntryStore and EntryScape (in Organic.Edunet still called "Confolio") – in the context of this project referred to as "Organic.Edunet repository tools" – was used from the very beginning of the content population process. The Organic.Edunet federation consists of numerous repository tool installations which are harvested using OAI-PMH by the Organic.Edunet portal [24] on a regular basis. More than 11000 educational resources have been described with educational metadata by several hundred contributors so far. Roughly half of the learning resources were already described with some basic metadata without educational information. These already existing metadata instances were harvested using OAI-PMH and converted and mapped into RDF and LOM/DCAM.

Additional educational metadata was added in the Organic.Edunet repositories. This approach is greatly supported by the ReM^3 model, which allows a differentiation between local and external resources and metadata. Such a differentiation in combination with the use of separate metadata graphs is used to enhance harvested resource descriptions from e.g. the Intute repository[6]. In this case, two metadata graphs are used per resource: one with cached external metadata (in simple DC format harvested using OAI-PMH) and one with local educational metadata using LOM/DCAM. If Intute modifies the metadata in its repository it will be reflected in EntryStore after the next re-harvest. The locally annotated educational metadata remain untouched, which is only possible by keeping metadata from different origins in separate graphs.

6.2. *ARIADNE*

Following up on the results from Organic.Edunet and as proof-of-concept for the general applicability of ReM^3 and the reference implementation, the OAI-PMH target of the ARIADNE foundation[7] (see [35] for a description of ARIADNE's architecture) was harvested and triplified, resulting in around 50 million triples within 1.2 million metadata graphs in one EntryStore repository. The provided LOM metadata was mapped into the DCAM and converted into RDF during the harvesting process. As in the case of Organic.Edunet, a scaffolding approach to describing learning resources can be taken. The surrounding context of a learning resource can be bootstrapped using Link References, e.g. by providing different descriptions for different learning scenarios.

Another benefit of having all ARIADNE metadata in RDF is the possibility of running SPARQL queries against a large amount of learning resource descriptions. SPARQL can be used to formulate complex queries based on the LOM/DCAM elements to query and build graphs in the repository. An example is requesting a list of all LOM Learning Resource Types that a specific person has used when annotating learning materials. More complex queries can be formulated by using additional metadata elements and advanced query logic. A use case is the contextualization of learning resources, to get information on how different persons described the same resource with different metadata to reflect their specific use within various educational (or other) activities. The amount of triples will increase in the future as the implementation of the LOM/DCAM mapping is refined and completed.

[6] http://www.intute.ac.uk

[7] ARIADNE foundation, http://www.ariadne-eu.org

6.3. Europeana

The "Hack4Europe!" competition in Stockholm[8], organized by the Europeana project[9], had the goal to show the potentials of the Europeana content by building applications to showcase the social and business value of open cultural heritage data.

Within the scope of the hack day we developed another showcase to demonstrate how heterogeneous metadata can be managed using ReM^3. Like in Organic.Edunet, a combination of EntryStore and EntryScape was used. Both applications were extended in a way so that they can search in Europeana and extract Europeana metadata from the search results. This allows for adding resources directly from a Europeana search result to a user's personal portfolio for further annotation with contextual metadata. The demonstrated use case *Europeana portfolio*[10] was to search for resources which are suitable to be used in an educational context and to turn them into learning resources by annotating them with educational metadata in EntryScape. Technically this means searching and caching metadata described using the *Europeana Data Model* [13] and adding educational metadata (e.g. in LOM/DCAM) using a ReM^3 Link Reference in EntryStore. Everything is integrated into the EntryScape interface and the end user does not have to know anything about where the metadata originates from or which formats that are used.

7. Evaluation

So far, ReM^3 and its reference implementation EntryStore have been evaluated with respect to scalability, their suitability for resource annotation processes, and the adoption in production environments. These aspects are described below and provide a picture of the applicability of the platform in question.

7.1. Scalability analysis

A series of load tests has been carried out to get an impression of the scalability of the ReM^3 reference implementation EntryStore. In the sections below the test environment and the results from different scenarios will be discussed. The overall goal was to find out how many concurrent requests could be run while still staying below 100 ms response time to ensure the user has a feeling of instantenous response [25]. Taking round-trip times, request creation and parsing, and also user interface updates into consideration, the tests below aimed for response times at a maximum of 50 ms. The focus was on requests of entries for reading and metadata graphs for modification. There were no tests carried out with binary resources as their size depends too much on the specific use case and their handling does not involve any complex operations in the triple store.

7.1.1. Test environment

An EntryStore instance was deployed in a KVM-based virtual machine (VM) on a Linux host. The VM had access to four Intel Xeon X3440 CPU cores running at 2.53 GHz and 4 GB of memory. The client for creating the traffic to the server came with an Intel Core i5 with two CPU cores at 2.4 GHz. To create, log and graph the requests the application JMeter was used in version 1.6. The network connection between client and server was 100 Mbit (duplex) with an average round-trip time of 5 ms. EntryStore was configured to use Sesame's native store (using one "cspo" index) and all communication with the EntryStore instance was done via its HTTP API. The graphs as seen in this evaluation were created with *Loadosophia.org*. The tested scenario was a medium-sized repository as it was used in the Organic.Edunet project. The EntryStore instance was seeded with a copy of the Organic.Edunet data from the KTH installation. It consisted of 1 024 898 triples in 57 353 named graphs, holding around 11 000 ReM^3 entries.

7.1.2. Test results

To get a first feeling for the relation between response times and the amount of concurrent request threads, a first test run was made. For this a ramp-up scheme was used, where the amount of active client threads was increased up to a maximum of 300 concurrent threads during 360 seconds. In JMeter a virtual user (VU in the figures) is equivalent to one client thread. Every thread continuously requested a random entry from EntryStore, to retrieve a JSON object which assembles information from up to four named graphs (entry, metadata, cached external metadata, and resource). The results of this first test run, as can be seen in figure 6 and figure 7, shows that the response times increase with the amount of concurrent client threads, while the transactions per second stay at the same level. Interpreting these two graphs led to the

[8] http://www.hack4europe.se
[9] http://europeana.eu
[10] http://hack4europe.se/information/meta-solutions-europeana-portfolio/

conclusion that around 20 concurrent threads is most likely the number which provides low response times (below 50 ms) and high request throughput in this specific setting.

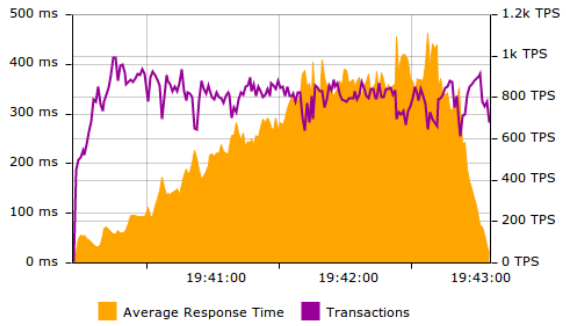


Figure 6. Response time and read transactions per second with 300 active client threads

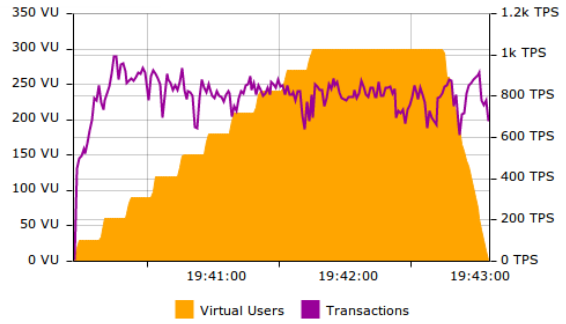


Figure 7. Read transactions per second with 300 active client threads

The following tests were run for 360 seconds with 20 concurrent threads, which resulted in an average response time of 38 ms while sustaining an average throughput of 478 requests per second (see figure 8). The spikes in the graph are probably caused by garbage collection in the JVM, but this was not further investigated.

Another test was carried out with modifying requests where metadata graphs of random entries were updated with graphs consisting of 40 triples (this was the average amount of triples per metadata graph in the Organic.Edunet repository). Modifying transactions are not treated concurrently by EntryStore and according to an analysis of the Apache log files of the Organic.Edunet installation only slightly more than 1% of all requests were modifying. As a consequence the amount of concurrent threads was decreased to 5

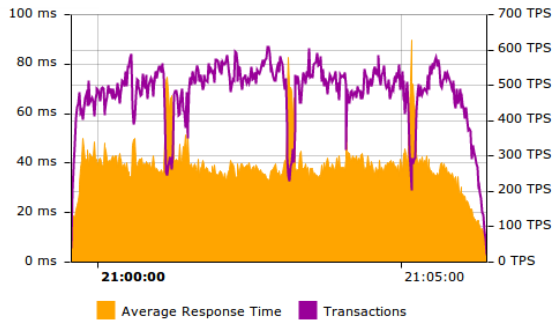


Figure 8. Response time and read transactions per second with 20 active client threads

in this test run. The average response time in this case was 34 ms while maintaining an average of 144 transactions per second, see figure 9.

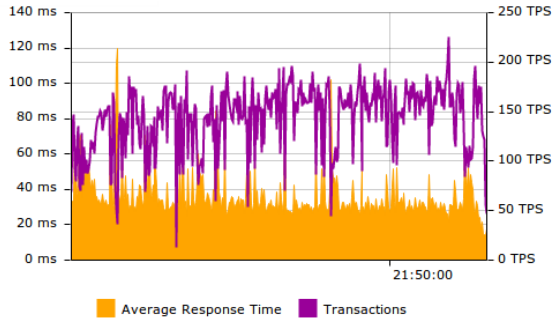


Figure 9. Response time and write transactions per second with 20 active client threads

The significantly lower number of transactions per second can be explained by several critera which are specific for modifications in the repository:

- To ensure consistency, there is no support for concurrent write transactions in Sesame's Native Store, so modifying requests block each other (this does not affect reading requests).
- Each modification triggers not only updates in the triple store, but also in the Solr index as all literals are indexed for free-text search.
- The request body in RDF/JSON has to be parsed by the server, which is not the case for reading requests.

7.1.3. Discussion and possible improvements

The described test is only the beginning of a more structured series of tests in which various different scenarios will be designed. This is out of scope for

this article and will be published separately. However, the numbers above give a preliminary picture of the scalability of EntryStore in the context of the Organic.Edunet scenario. The next steps are to also test with different backends, so far only Sesames Native Store was used. There are other backends which implement the *Sesame Sail API*[11] and claim to scale well (and also support concurrent write operations in contrast to Native Store), such as *Bigdata*[12], *OWLIM*[13] and *Virtuoso*[14]. Big data sets with realistic data on both a single server and in a cluster are to be tested and provisional tests with an EntryStore instance holding slightly above 2 million entries on a single machine have shown good results. However, there is no thorough analysis of EntryStore instances of this size yet, so this was not included in this article.

7.2. Suitability for resource annotation processes

A survey and structured interview were carried out in order to analyze the current practices and technologies for resource annotation processes in heterogeneous metadata repositories. The results are primarily valid for learning repositories because of the background of the participating experts, but can be also be applied to metadata management in generic repositories because of the common characteristics of these systems. User experience (UX) was not investigated even though some of the discussed issues may have implications for the UX as well.

7.2.1. Survey results

The survey consisted of 5 multiple choice questions and 7 structured interview questions (see the appendix for a list questions). The results are based on the answers from 16 experts. All of the respondents have professional experience with resource annotation, the majority of them (69%) within research. Most of them (63%) either develop systems for resource annotation and some also annotate themselves (38%). Some are or were involved in the development of metadata specifications or other similar activities (25%). All of the participants were affiliated with a university at the time the survey was carried out. In total 12 organizations from 7 European countries were represented.

Most of the annotation activities in which the participants were involved were about annotation from scratch (69%) and enhancement of already existing metadata either in the same (50%) or a different context (44%). In the cases where existing metadata was enhanced (64%) the most common solution was to copy the original metadata into the local system and modify there, i.e. causing a fork. In only 21% of the cases the original metadata was preserved and linked to instead of forked. All respondents were involved in the annotation of information resources. Some have experience with annotation of parties such as persons and organizations (31%), but also with physical objects such as non-human objects or substances (13%). The primary classification techniques were ontologies (88%) and tags (56%). In almost half of the cases also formal relationships (i.e. predicates) between resources and tags or concepts (44%) were applied.

7.2.2. Structured interview results

In the structured interview the respondents were asked to discuss several issues in metadata management in the context of resource annotation. The biggest and most relevant for this evaluation were: important features for managing information; possible problems when copying (as it is the case in harvesting) metadata between systems; and problems that could occur when enhanced metadata are linked to their original metadata (instead of copied).

Most important features. The respondents were asked to give an account of the features they would prioritize if they were to build a tool which is capable of handling resources and their metadata. The primary feature requirement was interoperability of protocols and formats, almost all respondents mentioned it in one way or another. Abstract models such as DCAM or MLR were recommended over XML-based models. The handling and transformation of metadata in various schemas and formats was a requirement. The reuse of existing metadata was mentioned together with the possibility of contextualizing resources. A majority of the respondents also considered provenance and the tracking of contributions as important. Another inquired feature was the capability to maintain a clear separation between metadata from different sources. This included a demand for links between all metadata and resources. Interlinking resources (i.e. creating relations between resources) and automatic or semi-automatic term extraction were other features that were on the wish list.

[11] http://www.openrdf.org/doc/sesame2/system/ch05.html
[12] http://www.systap.com/bigdata.htm
[13] http://www.ontotext.com/owlim
[14] http://virtuoso.openlinksw.com

Issues when copying. When asked which issues come to their mind when metadata is copied (i.e. harvested) between repositories, most respondents were concerned about redundancy and conflicting changes in the metadata. Redundancy because the same or slightly modified versions of the same descriptions coexist in several different systems. Conflicts because changes in one repository are not reflected in other places or in the worst case overriden. Issues of data consistency and multiple copies of the same data are caused by cascaded harvesting, i.e. aggregators with intersecting harvesting targets. This demands a sophisticated identification management and provenance of metadata records. Inconsistencies between the semantics of same metadata elements in different systems have also been experienced. The attribution of metadata authors in the connection with IPR and metadata quality (e.g. certification or validation) is another issue which was considered non-trivial, as this requires meta-metadata being transferred between systems in addition to the usual metadata record.

Issues when linking. The answers were very different from the previous case (copying) when the respondents described possible issues when metadata was linked to instead of copied. All of them focused on performance issues and broken links or unavailable information. They stressed the fact that the performance is usually poor when external systems are involved and that metadata may disappear from the originating system and therefore become unavailable. Nobody mentioned the possibility of caching to avoid performance deterioration or other similar techniques. Instead of conflicting changes like in the previous case, it seemed to be a problem that original metadata only can be extended but not changed. Another potential problem mentioned was the maintenance of semantics of the established links, both between metadata and resources, in all combinations.

It should be mentioned that very specific requirements, issues or ideas, possibly touched on by only one person, were not included in the summaries above.

7.2.3. ReM^3 in the light of the interview results

This section takes a look at the capabilities of ReM^3 in the context of the results of the structured interview as summarized above.

ReM^3 and its reference implementation EntryStore have good prerequisites for solving interoperability issues. The model itself is based on RDF which can be considered a model that works well to support metadata harmonization (see [28], chapter 6 "Horizontal harmonization"). ReM^3's reference implementation EntryStore rests upon Web architecure and its standard protocols. Linking and a clear separation between different metadata graphs describing the same resource are built into the model, this is the core of a ReM^3 entry. The meta-metadata is part of an entry and keeps track of provenance which satisfies requirements such as metadata attribution, licensing, and quality certification. The performance issues that almost all respondents worried about are circumvented with a special metadata graph that only holds cached external metadata as described in 3.1. This makes it robust in cases where the external system is slow, offline, or if the resource or their metadata have been removed completely from the original system. The semantics of relationships were mentioned to be problematic because of potentially different interpretations. The relationships in ReM^3 are defined in RDFS [12] and if other relationships (i.e. predicates) are used they originate from a well-defined vocabulary or ontology.

In the light of the interview results the combination of ReM^3 and EntryStore can be considered being very close to the capabilities of repositories as imagined by the experts. To clarify, they were asked to think of how they would design a system that conformes to their requirements, but they were not requested to comment on ReM^3 or EntryStore.

7.3. Adoption

The showcases described in section 6 describe three quite different scenarios, one of which is the use of EntryStore and EntryScape as repository tools in the Organic.Edunet project (see 6.1). This is a production setting, with a federation consisting of four autonomous repositories. The other two briefly described showcases have a more experimental character. There is the proof-of-concept of using EntryStore for a large-scale triplification of ARIADNE (see 6.2) which basically could be used in production, but has not undergone thorough testing of specifics relevant for large datasets. There is also the Hack4Europe! contribution described in 6.3 where EntryStore and EntryScape integrate with the Europeana APIs and provide an interface for the contextualization of the cultural heritage data provided by the Europeana foundation.

In addition to Organic.Edunet, there are several other projects which make use of EntryStore and its EntryScape frontend in a production setting:

Virtual Open Access Agriculture & Aquaculture Repository (VOA3R). The VOA3R project[15] uses EntryScape to offer providers of Open Educational Resources (OER) a means to upload, link, and describe learning material which is compliant with the FRBR-based VOA3R Application Profile. The project is ongoing and it is yet unclear how many resources are going to be described using EntryScape.

TEL-Map. EntryScape is used in the TEL-Map project[16] to provide advanced descriptions of collaborative EC-funded projects as well as projects by individual researchers (such as PhD students). These metadata describe the overall project aspects which include the projects' contexts, assumptions, goals, impacts, and approaches. Relationships between projects or between projects and organisations can be formulated using semantic connections, e.g. an organisation can be partner, stakeholder, collaborator, etc, whereas projects can be successors, predecessors or simply related to other projects.

Hematology Net. In Hematology Net[17] a combination of EntryScape and the EHA CV passport[18] was used. A user can upload a resource and link it to a learning goal in the CV passport. Users have the possibility to provide their own competence profile through a digital version of the EHA CV passport in EntryScape. This allows them to match their competence profile with their learning goals to get recommendations for for educational material relevant for reaching their goals.

8. Conclusions

In this section the problems from 1.1 are revisited along with some considerations regarding the applicability of the information model and its reference implementation.

8.1. Problems in retrospect

Management of resources and their metadata. ReM^3 allows to distinguish between situations where either both resource and metadata, only metadata, or neither metadata nor resource are handled in the system. The solution is the introduction of the entry type (see 3.1), which is also the most important type for the showcases shown in section 6 as it provides a clear distinction between local and external (remote and harvested) resources.

Enrichment of metadata. The same resource can be described with different metadata, identified by named graphs. Basic metadata with e.g. title and description can be enhanced with educational metadata in an additional metadata graph and used in an educational context. A resource can be contextualized just by annotating it with additional metadata, building upon the already existing descriptions. Write-access is not necessary, the metadata belong to the respective users, and since the LD principles are followed, anyone can point to and describe anything.

Organization of metadata. An ReM^3 entry keeps track of all involved pieces of information and links back to the original source of information. Metadata are cached locally for performance reasons. This linking approach is also part of the solution to the problem on how to expose the combination of resource and their metadata in a Linked Data way. The ReM^3 entry contains statements to keep resources and their metadata together with the possibility of including additional relations.

Integration of heterogeneous information sources. It was unclear how to design an information model that can integrate heterogeneous information sources and supports harmonization of metadata expressions from different standards using a common carrier. The RDF data model is used as a common carrier by the reference implementation. Together with the ReM^3 entry type this provides a solution to the problem on how to integrate heterogeneous metadata and how to enrich metadata originating from other sources, as mentioned above. The latter also benefits from a clear distinction of separate descriptions, implemented using named graphs and kept together by ReM^3 entries.

Support for Web architecture and best practices for using named graphs in Web applications. All entities in the ReM^3 model (see 3.1) such as the entry itself, metadata, relations, principals, are identified by named graphs which are given dereferencable URIs. The resources are linked with their metadata through the entry information. Relationships between named graphs are expressed in in the entry information where

[15] Funded by the EC through ICT PSP, http://voa3r.eu

[16] Funded by the EC through FP7, http://telmap.org

[17] Funded by the EC through Leonardo da Vinci, http://www.hematologynet.eu

[18] http://www.ehaweb.org/education/online-learning-tools/curriculum-cv-passport/

also provenance and access control is handled. Named graphs can be retrieved and queried using HTTP and ReM^3's URL-schema (see 4.2). The use of HTTP and RDF wherever possible fulfills the requirement of supporting Web architecture and faciliates the creation of Web applications.

8.2. Applicability and evaluation

The information model has a reference implementation which is described and evaluated in the sections 4 and 7. It shows that ReM^3 is more than just an idea and that it can be applied in a meaningful way in real-world applications. Based on the answers from experts, an evaluation showed that ReM^3 implements a feasible approach for resource annotation processes (see 7.2). The extent of the adoption of EntryStore in several successfully completed and ongoing projects shows that ReM^3 works in the described scenarios (see 7.3) and showcases (see 6).

The focus on Semantic Web technologies and the support for SPARQL and Linked Data in general makes it possible to contribute to the LOD cloud. Even though the system currently is mostly deployed in the context of learning repositories with a focus on educational metadata, there are no restrictions regarding the metadata standards or application profiles in use. The supported protocols for metadata harvesting and querying can easily be extended.

9. Future work and next steps

In the context of metadata and resource management it is relevant to provide means for creating additional links between resources. Currently, interlinking is mostly based on lists (assuming that resources contained in the same list have something in common) and entries keeping together different metadata graphs describing the same resource in different contexts. In addition to that it would be useful to provide semantics for lists, such as e.g. programmes, courses, course modules, etc. and explicit semantic relations between resources by exposing this in the user interface.

The system works well with large repositories (several million entries in one installation), but there is little knowledge about where the limits are for concurrent access to very large systems in typical usage scenarios. An elaborated set of benchmarks has to be developed to collect significant evidence regarding performance and scalability. However, this is more related to the reference implementation than to the information model.

The REST-based interface has its limitations in certain collaborative use cases. If an update of a resource or metadata is performed by a client, all clients which have data cached e.g. in the browser cache have to make a conditional reload in order to see the changes. Due to the nature of HTTP, the clients do not get notified of any changes on the server side, they have to poll instead. This is an issue which can be solved by adding support for push technologies such as WebSockets [20] to EntryStore.

In addition to pushing information updates to the clients, it can be of interest to keep a version history of a resource and its metadata. Such snapshots in combination with a short summary of changes can provide input for collaborators on e.g. what has been changed by whom, when and why; to put it simply, information about how a resource has evolved over time.

Acknowledgements

The work presented in this article has been partially carried out with financial support from the EIT ICT Labs activity *Data Bridges* in the thematic action line *Digital Cities of the Future*, and the EC-funded projects *Organic.Edunet* (grant agreement ECP-2006-EDU-410012), *TEL-Map* (grant agreement 257822) and *Responsive Open Learning Environments* (ROLE) (grant agreement 231396), which the authors gratefully acknowledge.

Appendix

Questions of structured interview as summarized in section 7.2

- What is your relationship with metadata annotation?
- What was the purpose of the annotation process you were involved in?
- If existing metadata were enhanced, how was this managed?
- What kinds of resources have been annotated?
- Were any classification techniques used?
- From an information management perspective, which features do you consider as most important for an annotation tool?

- If you copy (harvest) metadata between systems and modify it, which potential problems come to your mind?
- If you - instead of copying - link between metadata instances (original and enhancement), which potential problems come to your mind?
- Which system(s) did or do you use for annotation?
- Are you familiar with or aware of any tool that allows for annotating information resources by linking metadata instead of copying?
- Do you have any other comments or questions that you consider relevant and that you think should have been asked?

References

[1] T. Berners-Lee. Design issues – axioms of web architecture: Metadata. http://www.w3.org/DesignIssues/Metadata.html, 1997.

[2] T. Berners-Lee. Linked Data - Design Issues. http://www.w3.org/DesignIssues/LinkedData.html, 2009.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284:34–43, 2001.

[4] J. Carroll, C. Bizer, P. Hayes, and P. Stickler. Named graphs. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(4):247–267, 2005.

[5] R. Cyganiak. The Linking Open Data cloud diagram. http://richard.cyganiak.de/2007/10/lod/, 2012.

[6] Dublin Core Metadata Initiative recommendation – Dublin Core metadata element set, version 1.1. http://dublincore.org/documents/dces/, 2010.

[7] Dublin Core Metadata Initiative recommendation – DCMI metadata terms. http://dublincore.org/documents/dcmi-terms/, 2010.

[8] E. Duval. Final draft standard for Learning Object Metadata (LOM) IEEE 1484.12.1-2002. http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf, 2002.

[9] H. Ebner, N. Manouselis, M. Palmér, F. Enoksson, N. Palavitsinis, K. Kastrantas, and A. Naeve. Learning object annotation for agricultural learning repositories. In *Proceedings of the Ninth IEEE International Conference on Advanced Learning Technologies, 2009. ICALT 2009*, pages 438–442. IEEE, 2009. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5194270.

[10] H. Ebner and M. Palmér. A mashup-friendly resource and metadata management framework. In F. Wild, M. Kalz, and M. Palmer, editors, *Proceedings of the First International Workshop on Mashup Personal Learning Environments (MUPPLE08). Maastricht, The Netherlands*, volume 388, pages 14–17, September 2008. http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-388/ebner.pdf.

[11] H. Ebner and M. Palmér. RDFS of the Resource and Metadata Management Model (ReM^3). http://entrystore.googlecode.com/svn/store/trunk/doc/terms.rdf, 2012.

[12] H. Ebner and M. Palmér. Specification of the Resource and Metadata Management Model (ReM^3). http://code.google.com/p/entrystore/wiki/ReM3, 2012.

[13] Europeana Data Model documentation. http://pro.europeana.eu/edm-documentation, 2012.

[14] F. Enoksson. *Flexible Authoring of Metadata for Learning: Assembling forms from a declarative data and view model*. Licentiate thesis, KTH Royal Institute of Technology, School of Computer Science and Communication, 2011.

[15] F. Enoksson, M. Palmér, and A. Naeve. An RDF modification protocol, based on the needs of editing tools. In *Metadata and Semantics, Post-proceedings of the 2nd International Conference on Metadata and Semantics Research, MTSR 2007, Corfu Island, Greece*. Springer, October 2007.

[16] EntryStore project at Google Code. http://code.google.com/p/entrystore/, 2012.

[17] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000. http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.

[18] F. Gandon and O. Corby. Name That Graph or the need to provide a model and syntax extension to specify the provenance of rdf graphs. *Proceedings of the W3C Workshop - RDF Next Steps*, 2010.

[19] Y. Gil, S. Miles, K. Belhajjame, H. Deus, D. Garijo, G. Klyne, P. Missier, S. Soiland-Reyes, and S. Zednik. PROV model primer. *W3C Working Draft*, 2012. http://www.w3.org/TR/prov-primer/.

[20] I. Hickson. The WebSocket API. *W3C Working Draft*, 2012. http://www.w3.org/TR/websockets/.

[21] I. Jacobs and N. Walsh. Architecture of the World Wide Web. *W3C Recommendation*, 1:1–37, 2004. http://www.w3.org/TR/webarch/.

[22] C. Lagoze, H. Van de Sompel, M. Nelson, and S. Warner. The Open Archives Initiative Protocol for Metadata Harvesting. http://www.openarchives.org/OAI/openarchivesprotocol.html, 2002.

[23] F. Manola and E. Miller. RDF primer. *W3C Recommendation*, 10(February):1–107, 2004. http://www.w3.org/TR/rdf-primer/.

[24] N. Manouselis, K. Kastrantas, S. Sanchez-Alonso, J. Cáceres, H. Ebner, M. Palmer, and A. Naeve. Architecture of the Organic.Edunet web portal. *International Journal of Web Portals (IJWB)*, 1(1):71–91, 2009.

[25] R. B. Miller. Response time in man-computer conversational transactions. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, AFIPS '68 (Fall, part I), pages 267–277, New York, NY, USA, 1968. ACM.

[26] L. Moreau. The foundations for provenance on the web. *Foundations and Trends in Web Science*, 2(2-3):99–241, 2010.

[27] M. Nilsson. Description Set Profiles: A constraint language for dublin core application profiles. http://dublincore.org/documents/dc-dsp/, 2008.

[28] M. Nilsson. *From Interoperability to Harmonization in Metadata Standardization: Designing an Evolvable Framework for Metadata Harmonization*. PhD thesis, KTH Royal Institute of Technology, School of Computer Science and Communication, 2010. http://kmr.nada.kth.se/papers/SemanticWeb/

FromInteropToHarm-MikaelsThesis.pdf.

[29] M. Palmér, F. Enoksson, M. Nilsson, and A. Naeve. Annotation profiles: configuring forms to edit RDF. In *Proceedings of the 2007 international conference on Dublin Core and Metadata Applications*, DCMI '07, pages 10–21. Dublin Core Metadata Initiative, 2007.

[30] A. Powell, M. Nilsson, A. Naeve, P. Johnston, and T. Baker. DCMI abstract model. http://dublincore.org/documents/abstract-model, 2007.

[31] E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF. *W3C Recommendation*, 2009(January):1–106, 2008. http://www.w3.org/TR/rdf-sparql-query/.

[32] L. Sauermann, R. Cyganiak, and M. Völkel. Cool URIs for the Semantic Web. *W3C Interest Group Note*, 49(December 2008):1–15, 2008. http://www.w3.org/TR/cooluris/.

[33] B. Simon, D. Massart, F. Van Assche, S. Ternier, E. Duval, S. Brantner, D. Olmedilla, and Z. Miklos. A simple query interface for interoperable learning repositories. In *Proceedings of the 1st Workshop on Interoperability of Web-Based Educational Systems in conjunction with 14th International World Wide Web Conference (WWW'05)*, pages 11–18. CEUR, 2005. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-143/paper02.pdf.

[34] H. Story, S. Corlosquet, M. Sporny, T. Inkster, B. Harbulot, and R. Backmann-Gmur. WebID 1.0 – web identification and discovery. *W3C Editor's Draft*, 2011. http://webid.info/spec/.

[35] S. Ternier, K. Verbert, G. Parra, B. Vandeputte, J. Klerkx, E. Duval, V. Ordoez, and X. Ochoa. The ARIADNE infrastructure for managing and storing metadata. *IEEE Internet Computing*, 13(4):18–25, 2009.

[36] R. Thurlow. RPC: Remote procedure call protocol specification version 2. *Internet Network Working Group Request for Comments*, page 25, 2009. http://www.ietf.org/rfc/rfc5531.txt.

[37] H. Van De Sompel, R. Sanderson, M. L. Nelson, L. L. Balakireva, H. Shankar, and S. Ainsworth. An HTTP-based versioning mechanism for Linked Data. In *Proceedings of the workshop on Linked Data on the Web (LDOW)*, April 2010.

[38] World Wide Web Consortium Technical Architecture Group issues list: httpRange-14: What is the range of the HTTP dereference function? http://www.w3.org/2001/tag/issues.html#httpRange-14.

[39] Web Access Control Ontology. http://www.w3.org/wiki/WebAccessControl, 2012.