Exposing Scholarly Information as Linked Open Data: RDFizing DSpace contents

Abstract

This paper introduces a transformation engine which can be used in order to convert an existing institutional repository installation into a Linked Open Data repository. We describe how the data that exists in a DSpace repository can be semantically annotated in order to serve as a Semantic Web (meta)data repository. We present a non-intrusive, standards-compliant approach that can run alongside with current practices, while incorporating state-of-the art methodologies. Also, we propose a set of mappings between domain vocabularies that can be (re)used towards this goal.

Keywords: Linked Open Data, RDF, SPARQL, DSpace, R2RML, Semantic Web, Ontology, Relational Database, Mapping

1. Introduction and motivation

During the last years, there is an increasing interest among the digital libraries community in the Linked Data paradigm and the use of Semantic Web technologies in the context of traditional bibliographic tasks. This raise in the awareness of the library community with respect to advances in the Linked Data front is best exemplified by several initiatives that aim at the introduction of new, flexible data models in the place of older and more rigid ones (Library of Congress, 2011), the activity of W3C's Library Linked Data Incubator Group (Baker et al., 2011) as well as the initiatives of several national libraries that serve their bibliographic metadata as Linked Data (Malmsten, 2008; Deutsche National Bibliothek, 2012; Biblioteca Nacional De Espana, 2012; British Library, 2012).

The advantages of the flexible RDF model compared with monolithic bibliographic standards employed so far in the bibliographic community are significant, allowing for

- easier integration with bibliographic content residing in external systems,
- sound grounding of bibliographic concepts in terms of well-established ontologies, vocabularies and taxonomies and
- a wide range of Semantic Web tools for the management, processing, visualisation and analysis of bibliographic information.

The interpretation of digital library content as Linked Data also creates potential for the development of applications reusing and mashing up open data from several interconnected domains and hence, creating knowledge along the way and facilitates serendipitous discovery of a library's content, when the latter is part of the Linking Open Data Cloud (lod-cloud.net).

Unfortunately, as in most cases where a new model is about to replace an older one, the advantages of RDF model adoption come at the cost of substituting existing software

infrastructure with new software solutions and migrating existing content to the new format. This is a severe hindrance for the integration of digital libraries to the Semantic Web and as a result, there is space for flexible and unobtrusive solutions that work on top of existing software.

We propose a modular approach to extract RDF from metadata stored in relational database-backed digital library systems, by layering on top of them a relational-to-RDF mapping engine. This engine accepts an input connection to a database and, by means of manually-defined R2RML mappings (RDB to RDF Mapping Language), generates an appropriate RDF graph. R2RML is the latest W3C recommendation (Das et al., 2012), specifying a mapping language for the definition of mappings among a relational database and one or more RDF graphs. Adoption of R2RML as a mapping language ensures interoperability and reuse of relational-to-RDF mappings across different mapping engine implementations and various database system products.

We justify the feasibility of our proposed solution by applying and testing it in the context of a use case scenario, considering the most popular and widely deployed institutional repository platform, DSpace (dspace.org). The data is mapped to RDF and, in accordance to the benefits presented above, the system contents escape the relational database schema storage limitations and are unleashed openly as RDF.

Furthermore, another contribution lies in the mappings themselves that are introduced in this paper. We expect these mappings to be, directly or with slight modifications, reusable by digital library managers that seek ways to release their content as RDF using popular, mature Semantic Web ontologies. This is especially true for institutional repositories that employ the default DSpace Dublin Core metadata schema, current research information systems based on the CERIF data model, and to be more general, systems where the application of custom metadata schemas is kept to a minimum. However, what needs to be stressed here is that our solution can be applied to all kinds of systems that use a relational database backend for the storage of their metadata.

The paper is structured as follows: Section 2 reviews the bibliography works that offer similar or alternative approaches. Section 3 offers an in-detail system presentation, while Section 4 concludes the paper with our most important observations and future directions that could expand the hereby presented work.

2. Related work

In order to expose relational databases as triplestores, much work has been conducted during the years (Spanos et al., 2012; Sahoo et al., 2009; Konstantinou et al., 2008). Broadly, using the solutions that are available in the literature, we notice that the resulting data in triples is either extracted from a relational database of aligned to it. In other words, the data is either replicated or links are maintained, uni- or bi-directionally between the triples and the data that resides in the database.

In the former case, data can be migrated from the database. This is the case with tools such as 4store (Harris et al., 2009), an RDF server whose key features are performance, scalability

and stability, as it is targeted toward commercial environments. 4store is a triplestore as such, and it does not offer connectivity options to relational databases. Similarly to 4store, YARS2 (Harth et al., 2007) and Mulgara (mulgara.org) are semantic stores, engineered towards scalability.

We have to note that the aforementioned approaches serve merely for storing triples and do not provide any means to transform, or maintain mappings of any kind between the relational database contents and the resulting triples.

In the latter case, the data that lies in the database can be returned, answering queries on the ontology. This is the case with OpenLink's Virtuoso RDF Views (Erling and Mikhailov, 2007). This approach offers triplestore views over relational database contents. The mapping file, generated by Virtuoso, comprises a set of quad map patterns, which are in fact declarations that specify how the column values of tables are mapped to RDF triples. Triplify (Auer et al., 2009) also offers a solution to expose relational database contents as RDF and Linked Data. D2RQ (Bizer and Seaborne, 2004) uses the table-to-class and column-to-predicate approach to generate the mapping files automatically. The declarative approach that is followed is implemented as a Jena (Carroll 2004) graph. The approach allows relational databases to offer their contents as RDF triples without the need to replicate their contents.

The tool presented here belongs to the first category: the triplestore exists in parallel with the existing solution. The main advantage of the suggested approach compared to the rest of the tools in the literature is that it allows the migration to take place by following a standard such as R2RML. In the data migration case, in the surveyed tools, special emphasis is given in scalability, rendering the data migration process locked between the specific tools used. The hereby suggested approach solves both problems by allowing standards-based data export and migration to a repository that does not affect production systems. Additionally, reasoning is enabled at the resulting triplestore, allowing intelligent application development and further exploitation of the dataset, for instance discovering inconsistencies or deducing implicit information.

3. System Description and Use-cases

3.1 System description

The R2RML Parser is implemented as a modular Java project that comprises: (a) the parser, and (b) the faceted ontology browser. The parser consumes a relational database (MySQL and PostgreSQL supported at the time) using an R2RML description document, encoded in RDF, N3 notation. As the standard goes, the definition document describes how triples will be generated according to SQL queries posed on the relational database. A higher level overview of the system architecture is illustrated in Figure 1. As it is analyzed, on one side, to the left, we have the existing system that can be a (potentially) layered application, based on a relational database. The application may also have content binaries that reside in the file system, as is the case in DSpace institutional repositories. Using the proposed approach, the

user is able to generate triples, according to the mapping definition document, which is essentially a set of instructions to the parser, according to the R2RML specification.

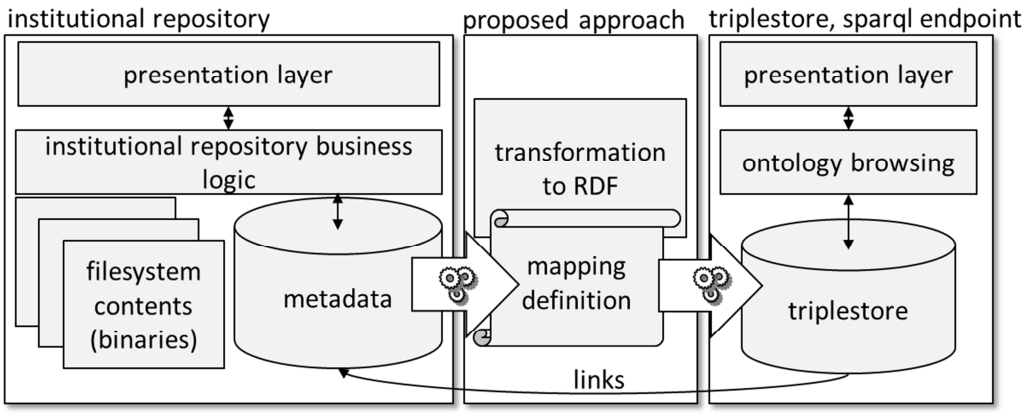
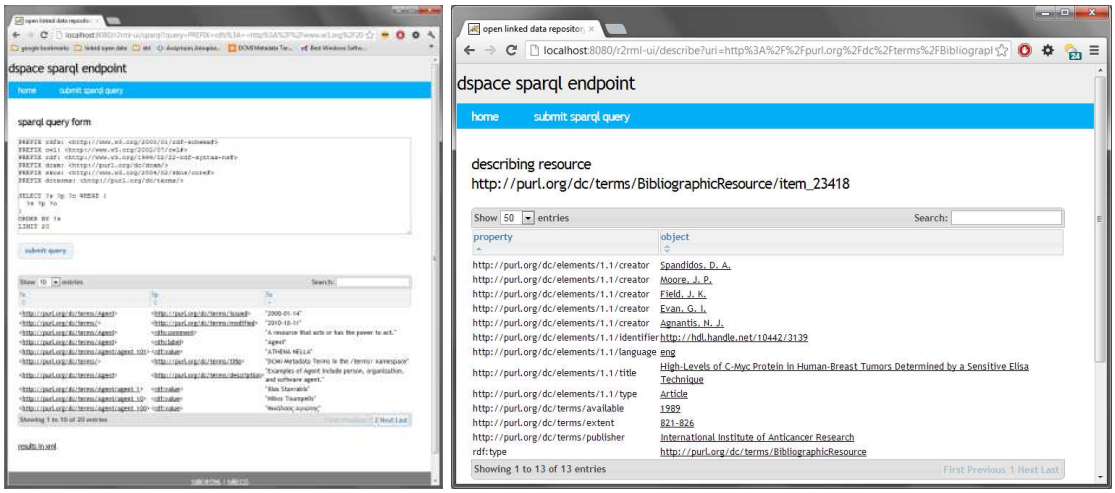


Figure 1. Component overview

The result of executing the mapping command is a triplestore and, as it is illustrated on the right of Figure 1, it comprises the metadata triplestore, the ontology browsing API that allows a presentation layer creation and the provision of a SPARQL endpoint.

Screenshots of the ontology browser over the resulting triplestore are depicted in Figure 2. On Figure 2 (a), a screenshot is offered with an example SPARQL query that is submitted via the web interface, followed its results. In Figure 2 (b), we can see a subject followed by its predicate-object, in a page collecting all known facts about a certain resource URI.

However, the presentation is the tip of the iceberg: the simplistic approach through which the information is served to the client conceals the complexity in defining the transformations through which it has to undergo. In its current implementation, the web application allows browsing the class hierarchy, imposing arbitrary SPARQL queries and retrieving the results either in the UI or in XML as defined by W3C (Becket and Broekstra, 2008).



(a) (b)

Figure 2. Faceted ontology browsing and SPARQL endpoint

Technically, the parser system is tested against a DSpace installation on a Linux server, using the PostgreSQL database for its backend. The resulting triplestore is implemented using Spring MVC for the presentation, Jena TDB for ontology manipulation, and it is tested using MySQL or PostgreSQL for the triplestore persistence.

3.2 Mapping DSpace contents to RDF

The DSpace data model provides a basic infrastructure, on which metadata can be stored, following arbitrary schemas and vocabularies. Figure 3 illustrates how this is materialized: table `metadatavalue` holds all metadata values for all items. Each metadata value belongs to a metadata field, registered in the `metadatafieldregistry` table that stores the metadata fields for each schema that, in its turn, exists in the `metadataschemaregistry` table. We note that PK stands for Primary Key, while FK1 and FK2 refer to Foreign Keys.

The simplicity of the approach in its internal schema database, however, does not reflect the versatility that DSpace presents, as a whole. To be specific, DSpace allows inclusion and support of any metadata schema, Dublin Core being included by default. In that sense, the existing implementation does not aim to cover every DSpace installation but rather to provide a concrete starting point upon which the mappings can be completed.

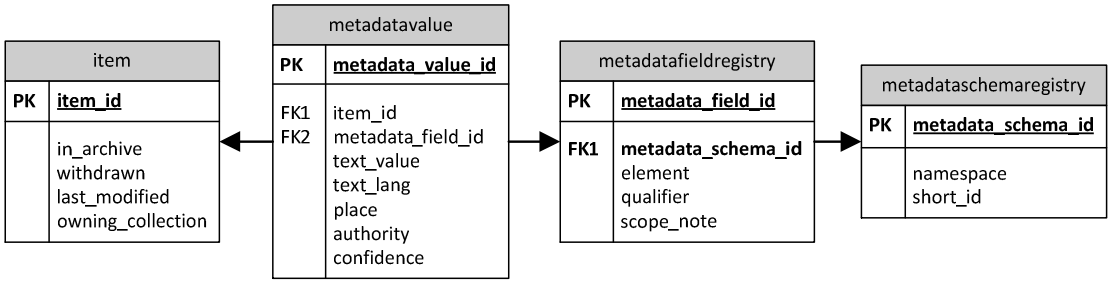


Figure 3. Metadata values storage in DSpace

Based on these, we can construct the SQL queries and respective RDF classes to map to. In the following, we present some correspondences between the default metadata elements shipped with DSpace (from now on, we will refer to them collectively as the DSpace Application Profile) and properties from widely used Semantic Web ontologies. The DSpace Application Profile is in fact a customized version of the Dublin Core Library Application Profile[1], a collection of Dublin Core elements that are specialized for use within the context of library-related applications. These correspondences can serve as a guide for the definition of R2RML mappings that will specify the form of the RDF graph produced from the contents of the repository database.

One could argue that some trivial mapping from the DSpace Application Profile to the Dublin Core RDF vocabulary would suffice for the publishing procedure, given that the latter is one of the most popular Semantic Web ontologies[2]. However, this is not entirely true for two

[1] DC-library Application Profile: http://dublincore.org/documents/library-application-profile/

[2] According to Linking Open Data Cloud statistics to date (stats.lod2.eu), the Dublin Core ontology is the third most used among all other ontologies, based on the overall number of statements that contain a term from it.

important reasons. First and foremost, the DSpace Application Profile extends the original Dublin Core Metadata Element Set with several custom qualifiers that do not have an exact counterpart in the Dublin Core vocabulary. Therefore, one would be forced to map these elements to the appropriate parent Dublin Core element and, by consequence, lose the semantics associated with the more specific qualifier.

The second reason is the fact that Dublin Core is a general purpose vocabulary for describing web resources and, as a result, the exact semantics of its elements is deliberately left underspecified. Notable examples of this are the *dc:relation* and *dc:date* elements, the meaning of which is vague and can vary depending on the exact nature of digital objects and resources considered. As long as the repository administrator – or, in general, the user who is responsible for the RDF translation process – knows the nature of the stored digital objects and the exact meaning of the DC metadata elements applied, she can choose more specialized properties from other vocabularies that represent closer conceptual matches than DC elements do. In other words, the mappings of Table 1 are not exact and their application must be combined with knowledge of the meaning of metadata elements or equivalently, knowledge of the metadata guidelines followed.

In Table 2, we present mappings from DSpace Application Profile to popular Semantic Web ontologies for the case of scholarly works, i.e. theses and scientific publications, which represent the largest amount of content deposited in institutional repository platforms. Likewise, mappings for other sorts of material (e.g. audiovisual content, museum exhibits, learning material) could be defined. We expect the presented mappings to serve as a guide to the selection of the most appropriate ontology term for each metadata field, a task that is often daunting for a repository administrator who is not aware of the wealth of relevant ontologies in the library domain. Often, the mapping that determines the form of the resulting RDF graph will result after communication among several persons, such as library domain experts, knowledge engineers and of course, the repository administrator. Table 1 gathers all ontologies referenced in Table 2. Note that this is a non-comprehensive list, in addition to not being the sole correct approach; ontology mapping and alignment is an ever-changing domain.

Title	**URL**	**Name space**	**Namespace URL**
The Bibliographic Ontology	bibliontology.com	bibo	http://purl.org/ontology/bibo/
Creative Commons Rights Ontology	creativecommons.org	cc	http://creativecommons.org/ns#
CiTo, the Citation Typing Ontology	purl.org/spar/cito	cito	http://purl.org/spar/cito/
Legacy Dublin Core element set	dublincore.org/documents/dces/	dc	http://purl.org/dc/elements/1.1/
DCMI Metadata Terms	dublincore.org/documents/dcmi-terms/	dcterms	http://purl.org/dc/terms/
FaBiO: FRBR-aligned bibliographic ontology	purl.org/spar/fabio	fabio	http://purl.org/spar/fabio/
FRBRcore	purl.org/vocab/frbr/core	frbr	http://purl.org/vocab/frbr/core#
FRBRextended	purl.org/vocab/frbr/ex	frbre	http://purl.org/vocab/frbr/extended#

	tended#		
IFLA's FRBRer Model	iflastandards.info/ns/fr/frbr/frbrer/	frbrer	http://iflastandards.info/ns/fr/frbr/frbrer/
International Standard Bibliographic Description (ISBD)	iflastandards.info/ns/isbd/elements/	isbd	http://iflastandards.info/ns/isbd/elements/
Lexvo.org Ontology	lexvo.org/ontology	lvont	http://lexvo.org/ontology#
MARC Code List for Relators	id.loc.gov/vocabulary/relators	mrel	http://id.loc.gov/vocabulary/relators/
Open Provenance Model Vocabulary	purl.org/net/opmv/ns	opmv	http://purl.org/net/opmv/ns#
PRISM: Publishing Requirements for Industry Standard Metadata	prismstandard.org	prism	http://prismstandard.org/namespaces/basic/2.0/
Provenance Vocabulary Core Ontology	purl.org/net/provenance/ns	prv	http://purl.org/net/provenance/ns#
RDA Relationships for Works, Expressions, Manifestations, Items	rdvocab.info/RDARelationshipsWEMI	rdarel	http://rdvocab.info/RDARelationshipsWEMI
Schema.org	schema.org	schema	http://schema.org/

Table 1. Ontologies related to scholarly information

Mappings to the legacy Dublin Core element set are implied and have been omitted from Table 2. Therefore, in all cases of unqualified DC elements, the use of the corresponding homonymous property from the Dublin Core Ontology is proposed.

DSpace DC elements		**Ontology properties**
contributor		mrel:*, schema:contributor[3], dcterms:contributor, dcterms:creator
	advisor	mrel:ths
	author	mrel:aut
	editor	mrel:edt, schema:editor
	illustrator	mrel:ill, schema:illustrator
	other	mrel:oth
coverage	spatial	schema:contentLocation, dcterms:coverage, dcterms:spatial
	temporal	dcterms:coverage, dcterms:temporal
creator		mrel:cre, mrel:aut, frbrer:P2009, frbr:creator, schema:author, schema:creator, dcterms:creator
date		frbrer:P3003, frbrer:P3010, dcterms:date
	available[4]	fabio:hasEmbargoDate, prism:embargoDate, dcterms:available
	copyright	schema:copyrightYear, dcterms:dateCopyrighted
	created	opmv:wasGeneratedAt,prism:creationDate, schema:dateCreated, dcterms:created
	issued	fabio:hasPublicationYear, frbrer:P3055, prism:publicationDate, isbd:P1018, schema:datePublished, dcterms:issued
	submitted	fabio:hasDepositDate, dcterms:dateSubmitted

[3] Depending on the exact semantics of contribution, any of the MARC Code List for Relators properties may be used. They are all defined as specializations (subproperties) of dc:contributor.

[4] DSpace date.available element denotes the date when an item becomes freely available to public, in other words when all access restrictions and embargoes are lifted.

	updated	fabio:dateLastUpdated, schema:dateModified
description		bibo:shortDescription, isbd:P1037-P1046, isbd:P1064-P1068, isbd:P1073, isbd:P1078-P1079, isbd:P1086-1087, isbd:P1090-P1101, isbd:P1123-1124, isbd:P1136[5], rdarel:descriptionOfWork, schema:description, schema:comment, dcterms:description
	abstract	bibo:abstract, rdarel:abstractWork, rdarel:abstract, rdarel:abstractExpression, dcterms:abstract
	provenance	opmv:wasDerivedFrom, opmv:wasEncodedBy, opmv:wasGeneratedBy, prv:createdBy, prv:serializedBy, dcterms:provenance
	statementofresponsibility	frbrer:P3021, isbd:P1007, isbd:P1010, isbd:P1029, isbd:P1059, isbd:P1141, isbd:P1142, isbd:P1153
	tableofcontents	dcterms:tableOfContents
	version	prism:versionIdentifier
format		frbrer:P3023, dcterms:format
	extent[6]	bibo:numPages, fabio:hasPageCount, frbrer:P3024, isbd:P1022, isbd:P1053, prism:byteCount, schema:numberOfPages, dcterms:extent
	medium	frbrer:P3025, isbd:P1003, dcterms:medium
	mimetype	schema:encodingFormat
identifier		bibo:asin, bibo:coden, bibo:doi, bibo:eanucc13, bibo:gtin14, bibo:handle, bibo:identifier, bibo:lccn, bibo:oclcnum, bibo:pmid, bibo:upc, fabio:hasDigitalArticleIdentifier, fabio:hasArXivId, fabio:hasCODEN, fabio:hasHandle, fabio:hasNationalLibraryOfMedicineJournalId, fabio:hasPII, fabio:hasPubMedCentralId, fabio:hasPubMedId, fabio:hasURL, frbrer:P3028[7], frbrer:P3031, isbd:P1032,isbd:P1154, prism:doi, prism:url, schema:url, dcterms:bibliographicCitation, dcterms:identifier
	citation[8]	bibo:cites, cito:citesAsAuthority, cito:citesAsDataSource, cito:citesAsEvidence, cito:citesAsRecommendedReading, cito:citesAsRelated, cito:citesAsSourceDocument, cito:citesForInformation, cito:citesAsMetadataDocument, frbre:isReferentiallyRelatedToExpression, frbre:isReferentiallyRelatedToWork
	isbn	bibo:isbn, bibo:isbn10, bibo:isbn13, prism:isbn, schema:isbn
	issn	bibo:eissn, bibo:issn, fabio:hasIssnL, isbd:P1030, prism:eIssn, prism:issn
	sici	bibo:sici, fabio:hasSICI
	uri	bibo:uri
language		frbrer:P3011, lvont:language, schema:inLanguage, dcterms:language
	iso	lvont:iso639P1Code, lvont:iso639P2BCode, lvont:iso639P2TCode, lvont:iso639P3Code, lvont:iso639P5Code, iso15924Alphacode
	rfc3066	dcterms:RFC3066

[5] ISBD contains several specialized properties that relate a bibliographic resource to a note describing some of its aspects.

[6] The extent of the resource described is the number of physical units that make up the physical carrier: these units may be either pages or even bytes.

[7] frbrer:P3028 stands for "manifestation identifier", while frbrer:P3031 stands for "item identifier", following the known FRBR distinction. Usually, repository items correspond to a manifestation, but depending on the conceptual organization of items in the repository, frbrer:P3028 may also be suitable.

[8] Roughly represents the inverse of the relation.isreferencedby element.

publisher		mrel:pbl, frbrer:P3056, isbd:P1017, schema:publisher, dcterms:publisher
relation		frbrer:P2043-P2110[9], rdarel:relatedWork, dcterms:conformsTo
	isformatof	dcterms:isFormatOf
	ispartof	frbrer:P2058, frbrer:P2080, frbrer:P2086, frbrer:P2092, frbr:part of frbre:isPartOfExpression, frbre:isPartOfItem, frbre:isPartOfManifestation, frbre:isPartOfWork, rdarel:containedInWork, rdarel:containedInExpression, rdarel:containedInManifestation, rdarel:containedInItem, rdarel:containedIn, dcterms:isPartOf
	ispartofseries	rdarel:inSeriesWork, rdarel:inSeries
	haspart	frbrer:P2057, frbrer:P2079, frbrer:P2085, frbrer:P2091, frbr:part, frbre:hasPartExpression, frbre:hasPartItem, frbre:hasPartManifestation, frbre:hasPartWork, rdarel:containsWork, rdarel:containsExpression, rdarel:containsManifestation, rdarel:containsItem, rdarel:contains, rdarel:wholePartRelationship, rdarel:wholePartRelationshipWork, rdarel:wholePartRelationshipExpression, rdarel:wholePartRelationshipManifestation, rdarel:wholePartRelationshipItem, dcterms:hasPart
	isversionof	frbrer:P2062, frbr:revisionOf, frbre:isARevisionOfExpression, rdarel:expandedVersionOfExpression, rdarel:expandedVersionOf, rdarel:revisionOf, rdarel:revisionOfExpression, dcterms:isVersionOf
	hasversion	frbrer:P2061, frbr:revision, prism:hasPreviousVersion, dcterms:hasVersion
	isbasedon	rdarel:basedOnWork, rdarel:basedOnExpression, rdarel:basedOn
	isreferencedby	bibo:citedBy, cito:isCitedBy[10], cito:isCitedAsAuthorityBy, cito:isCitedAsDataSourceBy, cito:isCitedAsEvidenceBy, cito:isCitedAsMetadataDocumentBy, cito:isCitedAsRecommendedReading, cito:isCitedAsRelatedBy, cito:isCitedAsSourceDocumentBy, cito:isCitedForInformationBy, frbre:isReferentiallyRelatedToExpression, frbre:isReferentiallyRelatedToWork, dcterms:isReferencedBy
	references	dcterms:references
	requires	dcterms:requires
	isrequiredby	dcterms:isRequiredBy
	replaces	dcterms:replaces
	isreplacedby	dcterms:isReplacedBy
rights		cc:license, prism:copyright, dcterms:accessRights, dcterms:license
	uri	cc:legalCode
	holder	schema:copyrightHolder
source		dcterms:source
	uri	frbrer:P3033

[9] The FRRBer Model contains several properties that relate Works, Expressions, Manifestations and Items with each other. Any of these properties may be used, if the relation nature is known. The same remark applies to the FRBRcore ontology, which contains several relevant properties.

[10] CiTO contains several properties denoting relationships between the described resource and another one referencing it. Any of those could be used to model this relationship, if its exact nature is known.

subject		fbrber:P2023, frbrer: P2025, frbrer:P2027, frbrer:P2029, frbrer:P2031, frbrer:P2033, frbrer:P2035, frbrer:P2037, frbrer:P2039, frbrer:P2041, frbr:subject, prism:keyword[11], schema:about, schema:keywords
	classification	fabio:hasSubjectTerm
title		frbrer:P3001, frbrer:P3008, isbd:P1004, isbd:P1012, isbd:P1026, schema:headline, dcterms:title
	alternative	bibo:shortTitle, isbd:P1005, isbd:P1027, prism:alternateTitle, schema:alternativeHeadline, dcterms:alternative
type		prism:genre, schema:genre, dcterms:type

Table 2. Mappings from DSpace Application Profile to Semantic Web ontologies

Next, we describe the core idea about how to transform a DSpace repository into Linked Open Data using an example. Table 3 illustrates a (subset of a) record with its metadata entries, as it is stored in DSpace. The language is assumed to be English ([en]) in all of the language-enabled fields.

DC Field	Value
dc.creator	Cairns, Francis
dc.title	Some reflections on the ranking of the major Games in fifth century B.C. epinician poetry
dc.date.available	1989-05-21
dc.identifier.uri	http://hdl.handle.net/10442/359
dc.type	Conference Item
dc.format.extent	5 pages
dc.coverage.spatial	Greece
dc.language	eng

Table 3. Example metadata record

The goal is to expose the above as an RDF description, taking into account the DCMI recommendation (Nilsson, 2008). Next, we provide a snippet of the target description, in N3 notation, using the namespaces provided in Table 1.

```
<http://purl.org/dc/terms/BibliographicResource/item_9386>
      a dcterms:BibliographicResource;
      dc:creator "Cairns, Francis";
      dc:title "Some reflections on the ranking of the major Games
                in fifth century B.C. epinician poetry";
      dcterms:available "1989-05-21";
      dc:identifier <http://hdl.handle.net/10442/359>;
      dc:type "Conference Item";
      dcterms:extent "5 pages";
      dc:coverage <http://purl.org/dc/terms/Location/location_9386>;
      dc:language "eng".
<http://purl.org/dc/terms/Location/location_9386> a dcterms:Location;
      rdf:value "Greece".
```

[11] The frbrer:P20XX properties differ on the nature of the subject (e.g. concept, object, place etc.). Furthermore, in some repositories, the subject element is used to denote a keyword, thus the prism:keyword property might be relevant.

In order to achieve the transformation, a mapping needs to be declared. Following the R2RML W3C specification (Das et al., 2012), the mapping definition takes the form of a set of instructions like the following:

```
<#dc-creator>
    rr:logicalTable <#dc-creator-view>;
    rr:subjectMap [
        rr:template
"http://purl.org/dc/terms/BibliographicResource/item_{item_id}";
    ];
    rr:predicateObjectMap [
        rr:predicate dc:creator;
        rr:objectMap [
            rr:template
      "http://purl.org/dc/terms/Agent/agent_{text_value}" ];
    ].

<#dc-creator-view>
      rr:sqlQuery """
      SELECT i.item_id AS item_id, mv.text_value AS text_value
      FROM item AS i
      INNER JOIN metadatavalue AS mv
      ON i.item_id=mv.item_id
      INNER JOIN metadatafieldregistry AS mfr
      ON mfr.metadata_field_id=mv.metadata_field_id
      INNER JOIN metadataschemaregistry AS msr
      ON msr.metadata_schema_id=mfr.metadata_schema_id
      WHERE i.in_archive=TRUE AND
      mv.text_value IS NOT NULL AND
      msr.namespace='http://dublincore.org/documents/dcmi-terms/' AND
      mfr.element='creator'
      """.
```

Alternatively, by altering the subject, predicate and object maps in the instructions above, we could map the contents of the example record from Table 3 into Linked Open Data, using a set of different vocabularies. Similarly to the above, the following snippet holds the same information, exposed using schema.org:

```
<http://purl.org/dc/terms/BibliographicResource/item_9386>
      a dcterms:BibliographicResource;
      schema:creator "Cairns, Francis";
      schema:headline "Some reflections on the ranking of the major
                       Games in fifth century B.C. epinician poetry";
      dcterms:available "1989-05-21";
      dc:identifier <http://hdl.handle.net/10442/359>;
      dc:type "Conference Item";
      schema:numberOfPages "5 pages";
      schema:contentLocation "Greece";
      schema:inLanguage "eng".
```

Using mapping declarations building on the concept above, we were able as a result to populate a database with DSpace entries, and expose the data via RDF, allowing the triplestore to coexist with the installed version.

It must be noted, that the resulting ontology should make use of more than one vocabulary and not be restricted to DC, but rather include properties from other domains where applicable. However, since the core DSpace installation provides only the DC vocabulary, usage of mapping definitions of other vocabularies is up to the user. This approach is both indicated and encouraged since, in the Linked Data world, vocabularies form islands of information which must be interconnected in the form of a graph, where applicable (5stardata.info). This is materialized by links from one vocabulary concepts to another and altogether provides the context for each piece of information, leveraging the overall description value.

3.3 System evaluation

The system was evaluated and its performance was compared with state-of-the art software. Specifically, it was assessed and validated against the D2RQ database-to-relational mapping platform, regarding both the mapping results and the corresponding times needed to produce them.

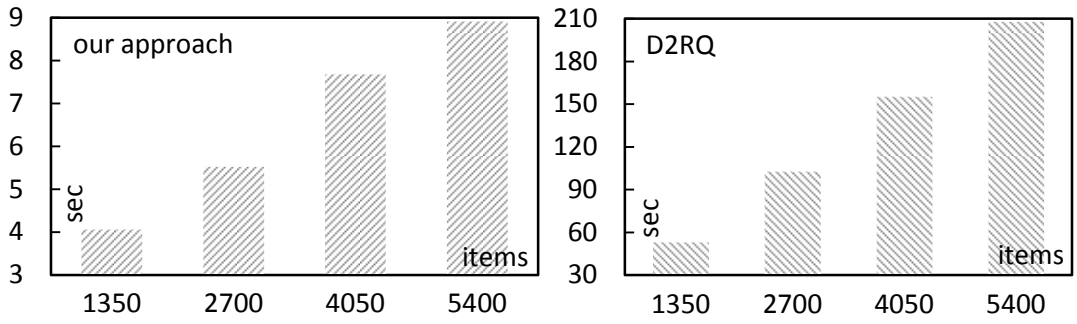


Figure 4. The time in seconds needed to export a DSpace repository into an RDF graph, depending on the number of items that are present in the repository

As an evaluation metric, we use the export time of the metadata contents of a DSpace repository to an RDF graph. The dataset we used for the experiments was the Helios database (helios-eie.ekt.gr), the repository for the National Hellenic Research Foundation publications, containing at the evaluation time 5498 records by 6303 authors. The lab environment, on which the tests were realized, consisted of an Intel Core i5 processor, 4GB of RAM, running a Windows 7 64bit OS, with Java 6 and Postgresql 9.0.

- We considered item subsets of increasing size. The resulting RDF graphs contained 19086, 36048, 53830, and 72418 statements in the case of 1350, 2700, 4050, and 5400 bibliographic items in the repository, respectively. The data that was used is real-world data. We did not extend the dataset to millions of items because, keeping in mind the nature of the data, several thousand publications is a number big enough for any repository.
- The triples that were exported followed the form of the graph presented in the first example in Section 3.2. The resulting RDF graphs were identical, either having been produced by our approach or by D2RQ. For every measurement, three distinct tests were performed and the noted time is their average.
- The D2RQ version against which our software was tested is the (still experimental) R2RML version. At the time of the tests, D2RQ covered all R2RML features except for

the `rr:RefObjectMap` class, while our approach supported a functional minimum R2RML class subset that comprised: `rr:logicalTable`, `rr:subjectMap`, `rr:template`, `rr:class`, `rr:predicateObjectMap`, `rr:predicate`, and `rr:objectMap`.

- Our approach does not currently cover the full SQL expressivity as it does not allow combining SQL queries in a single one (query nesting, union, intersection or difference).
- It has to be noted that controlled vocabularies were not taken into account for the mappings. Both our tool and D2RQ make the assumption that the R2RML mapping provided defines an RDF graph that encodes the correct semantics of a relational database contents; thus, no additional semantic validation is performed.

As it can be observed in Figure 4, the time required to export the contents of a repository to RDF is proportional to the items that are stored in the repository, both for the hereby presented approach, and D2RQ. We can also note that our approach takes almost 5% of the time needed by D2RQ to export the RDF graph. This is because of the wealth of features that are supported by D2RQ and its maturity as software compared to our evolving prototype. Moreover, our approach does not support real-time transformations between SPARQL and SQL, as is the case with D2RQ, which supports RDF dumps as an additional feature to its core functionality. This explains the order of magnitude difference between the respective RDF export times. Therefore, our approach demonstrates outstanding results in cases when data needs to be dumped in an RDF graph that will coexist with the database.

4. Conclusions and Future Work

4.1 Discussion and Conclusions

We need to mention that, in addition to the variety of tools that can be used to enable SPARQL endpoint creation, a number of services start being offered indicating that the technologies involved are nowadays mature to the extent that they can support production systems. A list of active endpoints is maintained by W3C (w3.org/wiki/SparqlEndpoints). Therefore, the hereby presented approach contributes to the direction of facilitating the effort.

It could be argued that embedding microdata elements, to the web pages that are created could serve as semantic annotation. This approach allows web authors to add extra information to web pages in a manner materialized merely as instructions to search engine robots, and invisible to the web visitor. The approach is based on the fact that microdata instructions, when not recognized, they are ignored by the browser. However, this approach does not allow intelligent queries and needs additional implementation on the information system.

Notably, it turns out that the semantic technologies are an excellent fit for the digital library domain since in order to ensure its preservation it must be correctly (syntactically and semantically) annotated. Additionally, our approach in exposing digital library metadata can have a series of benefits, because of the following reasons:

- Since data does not change frequently enough to require real-time updates, asynchronous exports at time intervals do offer a plausible solution, as justified by the evaluation in Section 3.3.
- It allows avoidance of vendor lock-ins. In case there is a need for the mapping result to be migrated or moved to another tool, one can easily switch, i.e. parse the R2RML document using another tool such as Virtuoso.
- It is a non-obtrusive approach. Databases, being a mature technology are widespread to the extent that people involved in digital libraries are reluctant to embrace a new – however promising yet still not mainstream – technology and replace existing ones.
- It allows complex queries to be evaluated on the results, utilizing the full capacities of SPARQL. This is particularly important in DSpace installations since its aged implementation only allows browsing, full-text and simple-filter search queries.
- It allows for definition of de facto approaches when mapping existing well-defined and/or standardized database schemas to ontology schemas.
- Digital library content can be harvested and integrated by third-party remote software clients in order to create valuable meta-search repositories such as Europeana (europeana.eu) or OpenAIRE (openaire.eu), through which researchers can browse, search and retrieve scientific publications related to their work.
- Bringing existing content into the semantic web opens new capabilities about its migration, especially in the case where metadata is of larger volume than the actual data.

The application of an R2RML mapping that takes into account the mappings in Table 2 fosters the reuse of popular Semantic Web ontologies, a crucial factor for the uptake of the Semantic Web vision. Moreover, an RDF dataset that uses terms from widely deployed vocabularies has a higher probability of attracting third-party references, compared to a dataset based on custom-made ontologies, the exact purpose of which may be unclear. However, mere reuse of popular ontology terms does not lead by itself to true 5-star Linked Data (Berners-Lee, 2006). In order to achieve the latter, the *entities and concepts* referenced in the repository metadata must be recognized and suitable identifiers for them must be found among already published RDF datasets, in order to establish links between them. Such entities mainly include authors, subjects and keywords of items stored in a digital library. We argue that this recognition can be done either prior of after the RDF translation process.

One solution would be to perform the recognition of entities referenced in an item's metadata during the item submission process. This could be achieved by appropriate user interfaces features (e.g. auto-complete fields) and underlying web services that query popular datasets, such as DBPedia (dbpedia.org), in order to select the most relevant concept or entity. This procedure can be viewed as a semantic annotation of the item and therefore, should be performed by a person that is familiar with the content of the item being annotated. Luckily, in most item submission workflows in digital repositories, the person submitting the item is also among its creators and therefore, is familiar with the entities referenced in its descriptive metadata. The challenge though is to build an intuitive interface that facilitates the discovery of equivalent concept IRIs, even for end users that are not familiar with the concept of Linked Data and Semantic Web technologies.

The second solution would be to establish links after the generation of RDF data, with the help of automated matching tools and link discovery services, such as Silk (Volz et al., 2009) or Google Refine[12], combined with a validation step from a human expert. This approach, which we implicitly follow in this paper, is the most popular one in the Linked Data realm, favoring a "pay-as-you-go" style of integration (Paton et al., 2012), where the burden of entity disambiguation and link establishment is split between the data publisher and the data consumer or even third parties who can post dataset mappings freely on the Web. In other words, we just take the first step of translating bibliographic metadata into RDF form and worry later for mappings with already published IRIs, that will render the generated RDF graph into 5-star quality Linked Data.

Overall, the actual contribution lies in the definition of a standardized mapping document between the contents of a relational database supporting an institutional repository on one hand, and ontology triples on the other side, in addition to a collection of vocabulary elements that can be used and combined in order to describe common concepts in scholarly literature.

4.2 Future Work

The hereby presented work focuses on how to deal with the remaining legacy information that repositories have collected over time, given the amount of DSpace installations worldwide. DSpace is used in more than 700 institutions, and therefore this gives practical value to the proposed solution. However, besides DSpace, mapping files could also be created for similar institutional repository software, such as EPrints, which uses VoID[13] to describe the RDF datasets that it exports, or Fedora, that uses the rels-ext ontology.

The mapping document can be extended to include possible additions and modifications that repositories may have on their vocabulary. Nevertheless, the issue of discovering correspondences between RDF datasets is a challenging and interesting one, which we plan to deal with in the future. Furthermore, the approach could be extended to other types of repository software such as EPrints (eprints.org), Greenstone (greenstone.org), BePress (bepress.com) and ContentDM (contentdm.org). Also, an important step ahead would be to investigate the possibility of incremental exports. This would be expected to decrease export times and, when run as a system service/daemon, it could fully automate the procedure required to have a triplestore running side-by-side with the repository.

References

Auer, S., Dietzold, S., Lehmann, J., Hellmann, S., Aumueller, D. (2009) "Triplify: lightweight linked data publication from relational databases", *Proceedings of the 18th International Conference on World Wide Web*, Madrid, Spain, pp. 621–630.

Baker, T. et al. (2011) "Library Linked Data Incubator Group Final Report", available at http://www.w3.org/2005/Incubator/lld/XGR-lld-20111025/ (accessed February 2013).

[12] Google Refine homepage: http://code.google.com/p/google-refine/
[13] The VoID vocabulary: http://semanticweb.org/wiki/VoiD

Beckett, D., Broekstra, J. (2008) "SPARQL Query Results XML Format, W3C Recommendation", available at http://www.w3.org/TR/rdf-sparql-XMLres/ (accessed 29 August 2012).

Berners-Lee, T. "Linked Data – Design Issues", available at http://www.w3.org/DesignIssues/LinkedData.html (accessed 13 September 2012).

Biblioteca Nacional De Espana (2012) "Linked Data at Spanish National Library", available at http://www.bne.es/en/Catalogos/DatosEnlazados/index.html (accessed January 2013).

Bizer, C., Seaborne, A. (2004) "D2RQ—treating non-RDF databases as virtual RDF graphs", *Proceedings of the 3rd International Semantic Web Conference (ISWC2004)*, Hiroshima, Japan, November.

British Library (2012) "Linked Data at the British Library", available at http://www.bl.uk/bibliographic/datafree.html (accessed December 2012).

Carroll, J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K. (2004) "Jena: implementing the semantic web recommendations", *Proceedings of the 13th World Wide Web Conference*, New York City, May.

Das, S., Sundara, S., Cyganiak, R. (2012) "R2RML: RDB to RDF Mapping Language, W3C Recommendation", available at http://www.w3.org/TR/r2rml/ (accessed 2 February 2013).

Deutsche National Bibliothek (2012) "The Linked Data Service of the German National Library", available at http://www.dnb.de/EN/Service/DigitaleDienste/LinkedData/linkeddata_node.html (accessed January 2013).

Erling, O., Mikhailov, I. (2007) "RDF support in the Virtuoso DBMS", *Proceedings of the 1st Conference on Social Semantic Web*, Leipzig, Germany, September, pp. 59–68.

Harth, A., Umbrich, J., Hogan, A., and Decker, S. (2007) "YARS2: A Federated Repository for Querying Graph Structured Data from the Web", Proceedings of the ISWC2007, pp. 211–224

Harris, S., Lamb, N., and Shadbolt, N. (2009) "4store: The Design and Implementation of a Clustered RDF Store", *Proceedings of the 5th International Workshop on Scalable Semantic Web Knowledge Base Systems*.

Jörg, B., et al. (2012) "CERIF 1.3 Full Data Model (FDM) Introduction and Specification", available at http://www.eurocris.org/Uploads/Web%20pages/CERIF-1.3/Specifications/CERIF1.3_FDM.pdf (accessed 15 February 2012)

Konstantinou, N., Spanos, D.E., Mitrou, N. (2008) "Ontology and Database Mapping: A Survey of Current Implementations and Future Directions", *Journal of Web Engineering*, Vol. 7 No. 1, pp 1–24.

Library of Congress (2011) "A Bibliographic Framework for the Digital Age", available at http://www.loc.gov/marc/transition/news/framework-103111.html (accessed 1 February 2013).

M.Malmsten (2008) "Making a Library Catalogue Part of the Semantic Web", International Conference on Dublin Core and Metadata Applications.

Nilsson, M., Powell, A., Johnston, P., Naeve, A (2008) "Expressing Dublin Core metadata using the Resource Description Framework (RDF), DCMI Recommendation", available at: http://dublincore.org/documents/dc-rdf/ (accessed August 2012)

Paton, N., Christodoulou, K., Fernandes, A., Parsia, B., Hedeler, C. (2012) “Pay-as-you-go Data Integration for Linked Data: Opportunities, Challenges and Architectures”, *Proceedings of the 4th International Workshop on Semantic Web Information Management (SWIM’12)*.

Sahoo, S., Halb, W., Hellmann, S., Idehen, K., Thibodeau, T., Auer, S., Sequeda, J., Ezzat, A. (2009) “A Survey of Current Approaches for Mapping of Relational Databases to RDF”, available at http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport.pdf (accessed 1 February 2013).

Spanos, D.E., Stavrou, P., Mitrou, N. (2012) “Bringing relational databases into the semantic web: A survey”. *Semantic Web Journal*, Vol. 3 No. 2, pp. 169-209.

Volz, J., Bizer, C., Gaedke, M., Kobilarov, G. (2009) “Discovering and Maintaining Links on the Web of Data”, *Proceedings of the 8th International Semantic Web Conference (ISWC 2009)*, pp. 650–665.