A Unified Approach for Representing Metametadata

Kai Eckert
Mannheim University, Germany
kai@informatik.uni-mannheim.de

Magnus Pfeffer
Mannheim University Library, Germany
pfeffer@bib.uni-mannheim.de

Heiner Stuckenschmidt
Mannheim University, Germany
heiner@informatik.uni-mannheim.de

Abstract

Using metametadata like provenance on statement level has been proposed as a possible solution to some—if not most—problems that occur in a metadata aggregation context. Examples for these problems are the information loss that occurs during metadata format conversion or the alignment of disparate metadata quality. Until now, no feasible way to store and manage metametainformation has been proposed. In this paper, we suggest the use of RDF reification as an elegant way to fill this gap that in addition adheres to all standards and recommendations.

We demonstrate this approach by the means of two scenarios: First, an enhancement for metadata crosswalks that supports the maintenance of the converted metadata, and second the use of metametadata for the efficient integration of subject annotations from different sources.

Keywords: rdf; reification; metametadata; crosswalk.

1. Introduction

Many libraries and institutions are creating or maintaining retrieval systems that draw metadata from diverse sources. Patrons are expecting to be able to search through all resources, print or online, in a unified system. Thus the augmentation of metadata from the online catalogue containing mostly monographs and journal titles with metadata from journal subscriptions or online repositories containing individual articles is becoming more commonplace.

To create a metadata pool that can be used for retrieval and presentation, the metadata needs to be harmonized on two aspects: quality and format. Both aspects have their own inherent problems that need to be dealt with. As we will see in the following sections, several approaches to solving these issues make use of metametadata, i.e. information about the metadata like provenance. The contribution of this paper is a unified way to represent and manage all kinds of metametadata using RDF.

Metadata quality. Krause (2008) compares the various sources for metadata to layers that are formed around the core content that is catalogued and indexed with high quality and also highly relevant to the patrons of a given library. Additional content of various degrees of metadata quality and relevance are grouped around this core content and supplement it. The author points out that it is not economically feasible to raise the indexing depth and quality of the content of the outer layers to the standards of the core content and instead argues that the semantic heterogeneity of the different layers needs to be addressed by an adequate infrastructure.

Hillmann et al. (2004) considered the problem of metadata quality in the context of metadata aggregation. While mainly focused on the practical problems of aggregation, the paper addresses the aspect of subsequent augmentation with subject headings and changes the emphasis from the record to the individual statement. Preserving provenance and means of creation on this level of detail is considered necessary by the authors. They proposed an extension of OAI-PMH to implement their solution. Hillmann (2008) further expands on quality issues and notes inconsistent use of metadata fields and the lack of bibliographic control among the major problems. Preserving provenance information at the repository, record or statement level is one of the proposed methods to ensure consistent metadata quality.

https://doi.org/10.23106/dcmi.952109547

The specific need for improved metadata quality with regards to annotations has been pointed out by Barton et al. (2003):

> There will always be some aspects of the metadata that are inaccurate, inconsistent or out of date, even in systems which have extensive quality assurance procedures in place and have invested heavily in the creation of good quality metadata. For example, when a published subject classification scheme is updated, new resources may be classified using new subject terms but existing resources may not be reclassified, giving rise to inconsistent subject-based searches. Furthermore, in established systems, there may be a drift over time between policy and practice; a study into cataloging practices [...] found the issue to be widespread.

Again, the proposed way to tackle the problem is to store metametainformation like version of the scheme used, time of annotation, etc. with the subject headings.

Metadata formats. In order to harmonize disparate metadata formats, a common ground needs to be established. The Dublin Core Metadata Set (DCMS) is well suited for the task at hand, as it is a common denominator of most if not all metadata formats and is also considered sufficient for indexing, retrieval and presentation using a search engine. To facilitate a conversion between different formats, crosswalks describing a mapping of fields between two metadata formats are used. Godby et al. (2008) have described ways to deal with the problem of managing and maintaining such crosswalks. They describe a repository of crosswalks that stores the semantic information needed for mapping fields in a way that is appropriate for editing by human metadata experts.

This information is then compiled into programs that deal with the conversion process at the file level. As most bibliographic metadata formats used by libraries have a richer element set with more rigid semantics, information is inevitably lost in the conversation process. The MARC to DCMS crosswalk maintained by the Library of Congress[1] illustrates this nicely: For the dc:date field, up to seven MARC fields contain information related to different dates and could be considered, but there is no way to express the semantic distinction in the DCMS.

Application Profiles. One possible approach to mitigate the information loss incurred in converting metadata is to create an application profile that captures the most essential metadata fields and offers a way to express important semantic distinctions. This is usually done by adding extensive qualifiers to the DCMS fields. Creating consensus on an application profile can be very difficult even in a clearly defined context and a small group of participants. Examples for the challenges during the creation of application profiles are described amongst others by Devey and Cote (2006) and Friesen et al. (2002). Furthermore there are only very limited ways to create machine-readable application profiles which hinders automatic checking of metadata for compliance with a given application profile.

Overview. In this paper, we will describe an alternative approach to mitigate information loss that is based on the idea of preserving provenance and other metametainformation at the statement level. The approach utilizes inherent properties of the RDF standard and is compatible with the DC recommendations. The paper is structured as follows: We will briefly introduce RDF and RDF reification and will describe its use for encoding arbitrary metametainformation. Next, we elaborate the use-cases for two distinct scenarios: aggregating metadata from multiple sources using crosswalks and augmenting existing metadata with annotations from multiple sources.

2 RDF and RDF Reification

The setup described in this paper uses the RDF reification to store metametadata. Thus we are restricted to RDF as target format. In this section, we give a brief introduction to RDF.

[1] http://www.loc.gov/marc/marc2dc.html

https://doi.org/10.23106/dcmi.952109547

2.1 RDF

The Resource Description Framework (RDF) is a set of standards of the World Wide Web Consortium (W3C) to express descriptive statements about arbitrary resources. Such a statement is represented as a triple consisting of subject (S), predicate (P) and object (O). Subject and predicate are represented as a uniform resource locator (URI) so that these elements can be identified unambiguously. The object can be either an URI or a literal. Typical RDF statements are shown in Example 1.

	Subject	*Predicate*	*Object*
1	ex:p123	rdf:type	ex:person
2	ex:p123	ex:hasName	"Kai Eckert"
3	ex:p123	ex:worksFor	ex:unima

Example 1: A simple RDF example

There are several defined ways to express RDF statements, among which the XML formulation (RDF/XML) is the most common one. The simpler and more human readable Notation 3 (RDF/N3) is equally expressive. In this paper, we use a simplified triple notation similar to N3, with namespace[2] abbreviations to save space and improve readability.

A common visualization of RDF data is a directed graph consisting of nodes representing the subjects and objects of the statements and the edges representing the predicates. Example 2 shows a full example with our triple notation, RDF/XML and the visualization as a directed graph.

Example 2 complies with the final Dublin Core recommendation on the expression of Dublin Core metadata elements with RDF (DC-RDF) which has been published by Nilsson et al. [2008].

In the example, an URI is used to unambiguously identify the subject heading which allows further statements about it. Here this is used to assign the literal values in two languages and the notation. Thus an object of one statement can become the subject of another statement and these chained statements ultimately form the directed RDF data graph.

The various principles and practices of using Dublin Core in different application settings led to the need for some clarification and proper definition, how Dublin Core metadata elements should be assigned to documents. This is addressed by the DCMI Abstract Model (DCAM, Powell et al. (2007)) which reached recommendation status in 2005. It provides the formal basis for the instantiation of Dublin Core metadata elements and is fully compatible with the semantic model of RDF.

2.2 Reification

RDF provides a means for the formulation of statements about statements, called reification. The use of reification was mentioned explicitly in a former version of DC-RDF, but "is now considered to fall outside the scope of the specification and is therefore no longer part of the January 2008 Recommendation. As it does not interfere with the metadata itself, however, reification can still be used in accordance with RDF specifications." (Nilsson and Baker [2008])

In our proposed approach, we can use reification to provide arbitrary additional information about existing Dublin Core metadata elements. In the RDF model, this means that a complete statement consisting of subject, predicate and object becomes the subject of a new statement that adds the desired information[3].

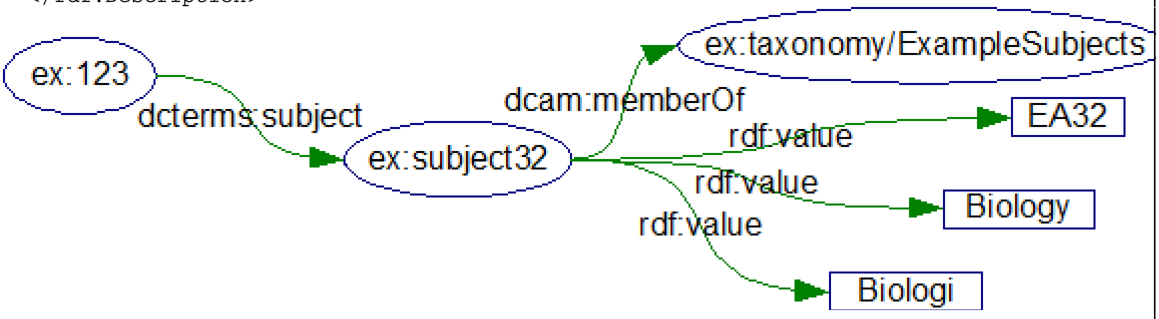
[2] Throughout this document, we use the following namespaces:
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:ex="http://example.org/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dcam="http://purl.org/dc/dcam/"

[3] As a statement cannot be identified uniquely in RDF beside the notion of S, P and O,

https://doi.org/10.23106/dcmi.952109547

	Subject	Predicate	Object
1	ex:123	dc:subject	ex:subject32
2	ex:subject32	dcam:memberOf	ex:taxonomy/ExampleSubjects
3	ex:subject32	rdf:value	"Biology"@en
4	ex:subject32	rdf:value	"Biologi"@sv
5	ex:subject32	rdf:value	"EA32"^^ ex:taxonomy/SubjectEncoding

```
<rdf:RDF>
  <rdf:Description rdf:about="http://example.org/123">
    <dc:subject>
       <rdf:Description rdf:about="http://example.org/subject32">
          <dcam:memberOf
              rdf:resource="http://example.org/taxonomy/ExampleSubjects"/>
          <rdf:value xml:lang="en">Biology</rdf:value>
          <rdf:value xml:lang="sv">Biologi</rdf:value>
          <rdf:value
              rdf:datatype="http://example.org/taxonomy/SubjectEncoding">
              EA32</rdf:value>
       </rdf:Description>
    </dc:subject>
  </rdf:Description>
```



Example 2: Subject Heading with full Description

Example 3 introduces our notation of statements about statements in this paper. We use the line number of our examples as an unique identifier for each statement and reference it as a subject via the # character.

	Subject	*Predicate*	*Object*
1	ex:123	dc:subject	ex:subject32
2	ex:subject32	dcam:memberOf	ex:taxonomy/ExampleSubjects
3	#1	ex:source	ex:p123

Example 3: Simple notation for RDF reification

2.3 Querying RDF data

There are many software tools that work on RDF data. For the examples in this paper, we used Sesame[4], an open source framework for storage, inferring and querying of RDF data.

We demonstrate by means of queries formulated in the Sesame RDF Query Language[5] that all of our use-cases can be implemented in this standard environment. The syntax is close to SQL and thus the examples can be understood without deeper knowledge of the query language.

a reification statement refers to all triples with the given S, P and O. In the Dublin Core context this ambiguity has no practical effects, as identical triples are semantically equivalent to duplicated metadata fields that can be safely discarded as redundant information.

[4] http://www.openrdf.org/

[5] http://www.openrdf.org/doc/sesame/users/ch06.html

https://doi.org/10.23106/dcmi.952109547

3 Crosswalks

In this section, we will explicate the use of metametadata in a metadata aggregation context: University libraries need to handle metadata from diverse sources that is usually encoded in incompatible metadata formats and of disparate quality. A typical setup might contain data from the local catalogue, which is encoded in MARC and adheres to high quality standards, while additional metadata from online journal providers is encoded in proprietary XML formats and contains only a minimal set of fields. Additional data from the local document repository server uses the unqualified DCMS as the metadata encoding format, but the fields are used inconsistently, as the authors of the documents are also responsible for providing the metadata.

To facilitate a unified search interface on this heterogeneous metadata accumulation, the metadata formats need to be aligned. Typically, a format that forms a common denominator of all formats involved is chosen and the metadata is converted into this target format using crosswalks. These crosswalks had usually been hand-crafted by metadata experts and then transferred into program logic or transformation stylesheets. Recently there have been efforts to establish a standard way to represent, manage and maintain crosswalks from one metadata format into another. In Godby et al. (2008) the authors describe an infrastructure that supports cooperative creating of crosswalks by providing a framework for reusable, descriptive mappings that get transferred into program logic by an automated process. The mappings have to be created by human metadata experts first and the results are consolidated into a spreadsheet of translation rules. Each line of the spreadsheet describes the rules for a single mapping of a pair of metadata elements; and it is important to note that these individual mappings are independent from each other. The automated script generation creates executables that will read metadata in the source format and writes metadata in the output format.

As the DCMS is the largest common denominator of the metadata present in our scenario, it is the obvious target format. This necessitates crosswalks from MARC to DCMS and from the proprietary formats to DCMS, which can be developed using the infrastructure described above. As our approach only works with RDF output, the DC-RDF format described above is used. We now propose to extend the program logic that is derived from the mappings to not only write the resulting metadata elements, but additionally for every element the following information:

- the version of the crosswalk used
- the number of the mapping rule used
- the source fields used

The additional information is written as extra RDF triples that describe a single metadata element. Example 4 shows a characteristic metadata record that has been extended in the proposed manner. For the following examples, we assume that all converted metadata records and the reification statements are stored in a single RDF file. We will show how the metametadata stored in this file can be used to mitigate typical problems that might arise in our scenario.

3.1 Use-case 1: Crosswalk updates

Consider the MARC to DCMS crosswalk used in the scenario. A likely source would be the crosswalk released by the Library of Congress[6] which has last been updated in 2008 and reflects MARC fields as they were at that time. There is a high likelihood that future changes and extensions to the MARC standard will cause changes to the rules in the crosswalk. In our scenario, this would imply a full re-conversion and re-indexing of all metadata that was originally in the MARC format. With the RDF file, it is possible to compose a simple query that retrieves the resource ids of the documents that have been converted using either a specific rule or a specific version of a crosswalk. See Example 5 for a sample query. The granularity of the query can be

[6] http://www.loc.gov/marc/marc2dc.html

https://doi.org/10.23106/dcmi.952109547

	Subject	*Predicate*	*Object*
1	ex:docbase/doc1	dc:title	"Example title"
2	#1	ex:rule	16
3	#1	ex:crosswalk	3
4	#1	ex:origin	MARC:245
5	ex:docbase/doc2	dc:title	"About finding a title"
6	#5	ex:rule	16
7	#5	ex:crosswalk	3
8	#5	ex:origin	MARC:245
9	ex:docbase/doc3	dc:title	"Lorem ipsum dolor"
10	#9	ex:rule	18
11	#9	ex:crosswalk	3
12	#9	ex:origin	MARC:245
13	#9	ex:origin	MARC:246
14	ex:docbase/doc4	dc:title	"Consetetur Sadipscing"
15	#14	ex:rule	19
16	#14	ex:crosswalk	6
17	#14	ex:origin	xml:/record/description

Example 4: Resulting RDF statements with additional Metametadata

chosen depending on the changes in the crosswalk. Thus the computing overhead and downtime for the re-conversion and re-indexing can be kept to a minimum.

```
select document, field, value from {document} {field} {value},
   {{document} {field} {value}} ex:rule {rule},
   {{document} {field} {value}} ex:crosswalk {crosswalk}
   where rule = 16 and crosswalk = 3

document         field     value
ex:docbase/doc1  dc:title  "Exmple title"
ex:docbase/doc2  dc:title  "About finding a title"
```

Example 5: Querying all affected records for a given crosswalk and rule

3.2 Use-case 2: Fixing mapping errors

In the ongoing operation of the retrieval system, patrons or staff will notice erroneous records. Typical examples are seemingly incomplete or corrupted records or result sets containing unexplainable search results that are caused by either quality issues in the source data or errors or insufficient or wrong rules in the crosswalk. In our scenario, an error report containing affected record ids would imply retrieval of the converted records in question, retrieval of the original records, retrieval of the crosswalk mappings and a manual search for the problematic mapping rule. With the RDF file, one can directly query for the crosswalk, its version and mapping rule that are responsible for the creation of the flawed metadata element. Even the fields from the source data can be retrieved. See Example 6 for a sample query. The result contains everything the crosswalk engineer might possibly need to analyze the cause of the error and create a fixed

```
select crosswalk, rule, origin from {document} {field} {value},
   {{document} {field} {value}} ex:rule {rule},
   {{document} {field} {value}} ex:crosswalk {crosswalk},
   {{document} {field} {value}} ex:origin {origin}
   where document = ex:docbase/doc1
      and field = dc:title and value = ''Example Title''

crosswalk  rule  origin
3          16    MARC:245
```

Example 6: Querying crosswalk, rule and original field for a given metadata element

https://doi.org/10.23106/dcmi.952109547

version in a most efficient manner. Once a new version of the crosswalk has been submitted to the repository, queries analog to the ones presented in use-case 1 can be used to determine all affected records that need to be re-indexed.

4 Annotation Provenance

In this section, we develop a second scenario where RDF reification can be used to prevent information loss while merging annotations from different sources. We show that this is the key to make transparent use of different annotation sources without compromises regarding the quality of our metadata. In line with our argumentation in this paper and the prior examples, we propose the storage of metametadata to mitigate any information loss and allow the usage of this information to achieve a better retrieval experience for patrons. With various queries, we show that we can access and use the additional information to regain a specific set of annotations that fulfills our specific needs.

This scenario focuses on the merging of manually assigned subject headings with automatically assigned ones. This corresponds to the integration of the outer layer in the Krause model, were annotation quality needs to be balanced with costs.

Example 7 shows a DC metadata record with subject annotations from different sources and additional information about the assignments via RDF reification. There is one document (ex:docbase/doc1) with assigned subject headings from two different sources. For each subject assignment, we see that a source is specified via an URI. Additionally, a rank for every assignment is provided, as automatic indexers usually provide such a rank. For example, a document retrieval system can make direct use of it for the ranking of retrieval results.

	Subject	*Predicate*	*Object*
1	ex:docbase/doc1	dc:subject	ex:thes/sub20
2	#1	ex:source	ex:sources/autoindex1
3	#1	ex:rank	0.55
4	ex:docbase/doc1	dc:subject	ex:thes/sub30
5	#4	ex:source	ex:sources/autoindex1
6	#4	ex:rank	0.8
7	ex:docbase/doc1	dc:subject	ex:thes/sub30
8	#7	ex:source	ex:sources/pfeffer
9	#7	ex:rank	1.0
10	ex:docbase/doc1	dc:subject	ex:thes/sub40
11	#10	ex:source	ex:sources/pfeffer
12	#10	ex:rank	1.0
13	ex:sources/autoindex1	ex:type	ex:types/auto
14	ex:sources/pfeffer	ex:type	ex:types/manual

Example 7: Subject assignments by different sources

For manual assignments, where usual no rank is given, this could be used to distinguish between high quality subject assignments from a library and, for example, assignments from a user community via tagging. The statements #13 and #14 are used to further qualify the source, more precisely, to indicate, if the assignments were performed manually (ex:types/manual) or automatically (ex:types/auto).

4.1 Use-case 1: Merging annotation sets

Usually, the statements from Example 7 are available from different sources (as indicated). The integration requires merging them into a single store. An interesting side-effect of the use of RDF and reification is that the merged data is still accessible from every application that is able to use RDF data, even if it is not possible to make reasonable use of our metametadata.

This is demonstrated by the first query in Example 8, which retrieves all subject headings that are assigned to a document. As in RDF all statements are considered identical that have the same subject, predicate and object, every subject heading is returned that is assigned by at least one source.

In most cases, these completely merged statements are not wanted. As promised, we show with the second query in Example 8 that we are able to regain all annotations that were assigned by a specific source (here ex:sources/pfeffer).

```
select * from {document} dc:subject {subject}

document          subject
ex:docbase/doc1   ex:thes/sub40
ex:docbase/doc1   ex:thes/sub30
ex:docbase/doc1   ex:thes/sub20

select * from {document} dc:subject {subject},
   {{document} dc:subject {subject}} ex:source {source}
   where source=<http://example.org/sources/pfeffer>

document          subject       source
ex:docbase/doc1   ex:thes/sub40 ex:sources/pfeffer
ex:docbase/doc1   ex:thes/sub30 ex:sources/pfeffer
```

Example 8: Querying the merged statements

4.2 Use-case 2: Extended queries on the merged annotations

In the following we show two extended queries that make use of the metametadata provided in our data store. Usually, one does not simply want to separate annotation sets that have been merged, but instead wants to make further use of these merged annotations. For example, we can provide data for different retrieval needs.

The first query in Example 9 restricts the subject headings to manually assigned ones, but they still can originate from different sources. This would be useful if a high retrieval precision is needed a decrease of the precision by the results of an automatic indexer is not acceptable.

The second query takes automatic assignments into account, but makes use of the rank that is provided with every subject heading.

```
select document, subject, type from {document} dc:subject {subject},
   {{document} dc:subject {subject}} ex:source {source},
   {source} ex:type {type}
   where type = <http://example.org/types/manual>

document          subject        type
ex:docbase/doc1   ex:thes/sub40  http://example.org/types/manual
ex:docbase/doc1   ex:thes/sub30  http://example.org/types/manual

select document, subject, rank from {document} dc:subject {subject},
   {{document} dc:subject {subject}} ex:rank {rank}
   where rank > 0.7
document          subject        rank
ex:docbase/doc1   ex:thes/sub40  1.0
ex:docbase/doc1   ex:thes/sub30  1.0
ex:docbase/doc1   ex:thes/sub30  0.8
```

Example 9: Ranked assignments and additional source information

https://doi.org/10.23106/dcmi.952109547

5 Conclusion

We have seen that many problems arising when trying to merge document metadata from different sources in a bibliographic context can be addressed by using so-called metametadata - data about the metadata to be integrated. This observation was already made by others but the literature criticized the lack of an adequate representation framework for representing metametadata. In this paper, we argue that the W3C Standard RDF does not only provide a suitable and convenient framework for representing the metadata itself, but also for metadata about metadata using reification. This enables us to represent metametadata very fine grained on statement level suitable for a variety of application scenarios. In our paper, we have discussed two concrete applications frequently appearing in practice, namely the use of crosswalks to convert metadata between different formats and the integration of different subject annotation sets for a given metadata set.

Our results have implications not only on a theoretical level. The use of RDF as a common language for representing metadata and metametadata at the same time has significant benefits for their practical handling. Researchers from the area of web-based information systems have developed a high number of scalable and robust tools for storing and querying RDF data. These tools can be used to implement a framework for supporting metadata aggregation in practice. This is especially attractive as more and more metadata is available in RDF format anyways. Examples include the RDF representation of the Dublin Core Standard and the SKOS format for representing vocabularies.

References

Barton J., Currier S. and Hey J. (2003) 'Building Quality Assurance into Metadata Creation: An Analysis based on the Learning Objects and E-Prints Communities of Practice', in 'Proceedings of the International Conference on Dublin Core and Metadata Applications'.

Devey M. and Cote M.C. (2006) 'The Development and Use of Metadata Application Profiles: The Government of Canada Experience.', Serials Librarian, 51(2), pp. p103–115, 13p.

Friesen N., Mason J. and Ward N. (2002) 'Building Educational Metadata Application Profiles', in 'Proc. Int. Conf. on Dublin Core and Metadata for e-Communities 2002', pp. 63–69, Firenze University Press.

Godby C.J., Smith D. and Childress E. (2008) 'Toward element-level interoperability in bibliographic metadata', code4lib (2).

Hillmann D.I. (2008) 'Metadata Quality: From Evaluation to Augmentation', Cataloging & Classification Quarterly, 46(1).

Hillmann D.I., Dushay N. and Phipps J. (2004) 'Improving Metadata Quality: Augmentation and Recombination', in 'Proceedings of the International Conference on Dublin Core and Metadata Applications', Dublin Core Metadata Initiative.

Krause J. (2008) 'Semantic heterogeneity: comparing new semantic web approaches with those of digital libraries', Library Review, 57(3), pp. 235–248.

Nilsson M. and Baker T. (2008) 'Notes on DCMI specifications for Dublin Core metadata in RDF', http://dublincore.org/documents/2008/01/14/dcrdf-notes/.

Nilsson M., Powell A., Johnston P. and Naeve A. (2008) 'Expressing Dublin Core metadata using the Resource Description Framework (RDF)', http://dublincore.org/documents/2008/01/14/dc-rdf/.

Powell A., Nilsson M., Naeve A., Johnston P. and Baker T. (2007) 'DCMI Abstract Model', http://dublincore.org/documents/2007/06/04/abstractmodel/.

https://doi.org/10.23106/dcmi.952109547