Building Blocks of Linked Open Data in Libraries

Abstract

In chapter 2 of Library Technology Reports *(vol. 49, no. 5) "Library Linked Data: Research and Adoption" we explore the world of linked open data (LOD) and linked open vocabularies (LOV) through the lens of our five building blocks of metadata (data model, content rules, metadata schema, data serialization, and data exchange). This chapter provides a common foundation for understanding the technologies used in the case studies in chapter 3 and frames the issues and opportunities associated with LOD/LOV in LAM (libraries, archives, and museums) communities.*

Introduction

Across all of the issues, trends, and research areas discussed in chapter 1, the need for lossless metadata interoperability and cross-domain harmonization is cited as a key enabling factor in the development of high-quality metadata systems.[1] This focus is seen in the specifications for RDA, which seeks to work for a wide range of material types and institution types, and in research that explores how standards like RDA, DACS, and CCO fit together and what might be done to improve the interoperability of cataloging and content standards in general.[2]

Within the broader web community, similar questions are being asked about how information can best be created, shared, harvested, and used in web-based environments. The LODLAM community and the W3C Library Linked Data incubator group help aggregate and explore issues of semantic and linked data in the context of cultural heritage and memory institutions. These communities are representative of LAM (libraries, archives, and museums) interest in the technology, data creation, and data publishing practices of web-centric communities.

LODLAM
http://lodlam.net

W3C Library Linked Data incubator group
www.w3.org/2005/Incubator/lld

As examples of this focus, the principles of linked open data (LOD) and linked open vocabularies (LOV) have been implemented in a number of LAM projects, including the Library of Congress Linked Data Service Authorities and Vocabularies, Europeana linked data, the Swedish Union catalog, the German National Library, the British National Bibliography, the Open Library, and the newly launched Digital Public Library of America (DPLA). While the scope and output of these initiatives differ, each offers a valuable example of how LAM metadata may be transformed, repurposed, and reused. In conjunction with this work, there is a wealth of research on the design, use, and implications of LOD for cultural heritage data.[3]

Library of Congress Linked Data Service Authorities and Vocabularies
http://id.loc.gov

Europeana Linked Open Data
http://data.europeana.eu

Swedish Union catalog
www.libris.kb.se

German National Library
www.dnb.de

British National Bibliography
www.bl.uk/bibliographic/natbib.html

Open Library
http://openlibrary.org

Digital Public Library of America (DPLA)
http://dp.la

To enable us to better understand how these systems work and what role LOD and LOV play in them, this chapter explores LOD in detail, using the metadata building blocks model introduced in chapter 1. For a refresher on the building blocks, refer to table 1.2. In addition, before we explore the LOD component, we will briefly review a definition for each building block.

What Is Linked Open Data?

Linked open data is a broad trend grounded in the work of the Semantic Web and largely described by referencing Tim Berners-Lee's work.[4] LOD is comprised of two distinct concepts, the first being that data published on the Web should connect readily with related information ("linked") and that in doing so should be as accessible to computers as it is to humans ("data"). The second concept central to LOD is that in order for data to be linked and reused, it must be open and free from legal and copyright restrictions ("open"). Like LOD, LOV follows these same principles, but instead of publishing data (resources and information about resources), LOV publishes vocabularies (term lists, metadata schemas, taxonomies, and ontologies) that help build the Semantic Web.

The perceived payoff of LOD is that it will revolutionize the Web by making it possible to build an information network that is understandable by computers as well as by humans and as a result is more scalable than human-readable data would be alone. The impact of LOV is that it enables communities to share data using common and copyright-free data structures and concepts. These vocabularies include rudimentary vocabularies (e.g., resource type) and complex ontologies (e.g., "friend of a friend"— FOAF) that help situate a resource in a large knowledge framework. Therefore, it is reasonable to view LOD as the mechanics through which libraries share data and LOV as the thread that binds this data together. There are a number of efforts in the LAM community to publish LOV, including the Open Metadata Registry and the Library of Congress Linked Data Service Authorities and Vocabularies.

Open Metadata Registry
http://metadataregistry.org

Because there are many publications that do an excellent job of listing these vocabularies and talking about tools and web services developed using LOD/LOV techniques,[5] we will not attempt a comprehensive listing here. Instead, we will explore how LOD/LOV serves as a new model for LAM metadata. We will do this by considering how it supports the five building blocks of metadata (data model, content rules, metadata schema, data serialization, and data exchange). For more information and definitions, refer to tables 1.1 and 1.2. To help us better understand how vocabularies fit into linked data structures, LOV will be considered to be a building block of LOD in our exploration. Therefore, we will focus on LOD as the primary item we are evaluating and consider vocabularies to be a component of the single building block metadata schema. Despite this simplification, it is important to realize that LOV is itself constructed using LOD specifications. Likewise, while the O in LOD is very important, it is a licensing and rights issue rather than a metadata design issue, and as such it is not the center of our exploration.

Succinctly stated, linked data is founded on four basic principles (adapted from Berners-Lee): (1) use URIs; (2) use HTTP URIs; (3) return useful data when HTTP URIs are dereferenced; (4) include links to other URIs. In addition, Berners-Lee employs a five-star rating system ranging from one star (data is on the Web with an open license) to five stars (data is fully linked RDF that employs open standards and can be read by machines).[6]

In his TED Talk, Berners-Lee discusses LOD as an "on-ramp" to the Semantic Web.[7] He describes an environment in which computers and people work together to identify things not using just free text, but also defined vocabularies and links to other web-based resources. He gives a simple example in which he labels the theater where he was presenting on a web-based mapping platform. That single contribution, he says, can now be used and reused by people and computers. This relatively simple idea is the root of LOD and is the inspiration for LAM metadata systems that seek to help our patrons, not by just giving them text-based records about our resources, but instead by helping them connect resources from multiple databases in an unambiguous and detailed way.

In order to understand LOD a bit better, let's take a look at its component parts.

Building Block 1: The LOD Data Model

In metadata terms, a data model is the underlying data structure that defines how a metadata statement or record is structured. The RDF (Resource Description Framework) model that we explore in this section uses a node/edge data model (graph). In contrast, MARC records use a flat-file record-based model in which all of the metadata statements belong to a record. Other data models that we are not considering in this chapter include entity-relationship models and object-based models.

The data model commonly used for LOD is RDF. RDF provides a model through which metadata about a resource can be captured, but it is unlike other standards like MARC, EAD, and other record-based metadata models in that it focuses on single assertions about a resource (e.g., the Title of this book is The Adventures of Huckleberry Finn). RDF by itself is simply a data model. It provides a method for making these simple statements and connecting statements together so that a series of statements can be viewed as a complete descriptive record of a resource. RDF is a W3C recommendation that was published in 1999 with a revised specification in 2004[8] and is a building block for Semantic Web concepts and technologies like OWL, SKOS, and linked data.

There are a number of excellent guides to RDF, and a full understanding of RDF requires more time than we have in this issue. Some excellent sources of information include the W3C RDF documentation (start with the primer) and the W3C list of books on the Semantic Web. In addition, chapter 2 in Liyang Yu's book *A Developer's Guide to the Semantic Web* provides a particularly accessible introduction that steps the reader through incrementally complex examples of RDF.[9] Given the accessibility of this approach, our exploration of RDF from a LAM perspective uses Yu's instructional model as a guide.

W3C RDF Primer (with links to other RDF documentation)
www.w3.org/TR/2004/REC-rdf-primer-20040210

W3C: Books on Semantic Web
www.w3.org/2001/sw/wiki/Books

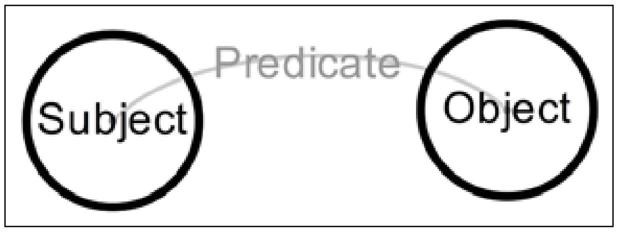

Figure 2.1
An RDF statement consists of a subject, a predicate, and an object

One of the key strengths of the RDF model is its ability to describe physical objects, digital objects, and the relationships between a primary object and its surrogate. This ability has been instrumental in supporting LAM adoption of RDF as a data-modeling standard, and it is no accident that the systems at the center of metadata developments in 2012 use RDF as a foundation.

The building blocks of RDF are statements that are also known as 3-tuples or triples. These statements consist of a subject, a predicate, and an object and conform to the node/edge structure of a graph. Figure 2.1 shows the basic structure of a statement. It is worth noting that the statement is directional in that the predicate is the name of some relationship that connects the subject and object entities. These statements at their core are very similar to the field/subfield and value model employed in MARC and the element/attribute and value model employed in record-based metadata models. For this reason, predicates, also known as properties, are very similar to metadata elements in other models.

An RDF triple consists of two nodes (subject and object) and an edge (property) that indicates a relationship between the resource and subject. RDF data models are considered to be more flexible than record-based metadata models like MARC because each statement is complete and does not depend on other statements. In contrast, every field of a MARC record depends on its context in the larger record and would have little meaning outside this context. In order to build this flexibility with such a simple model (subject, predicate, object), RDF allows objects and predicates to themselves become subjects, making an infinitely scalable tree structure that does not suffer from issues of ambiguity and scale that might accompany a flat-file standard like MARC. This is an advantage of RDF databases, also known as graph databases or triple stores, over relational databases that require a predefined schema and have limits to the relationships that can be asserted.

The introduction of graph-based and tree-based (e.g., FRBR) models into library cataloging has made MARC an unsuitable data model and data serialization

Table 2.1
Sample statements extracted from a MARC record for the book *The Adventures of Huckleberry Finn*

The Adventures of Huckleberry Finn	is_a	"Text"
The Adventures of Huckleberry Finn	is_written_by	"Mark Twain"
The Adventures of Huckleberry Finn	has_publication_date	"1884"
The Adventures of Huckleberry Finn	has_length	"438 pages"
The Adventures of Huckleberry Finn	is_published_by	"Chatto & Windus"
The Adventures of Huckleberry Finn	is_about	"Finn, Huckleberry (Fictitious Character) Fiction"
The Adventures of Huckleberry Finn	is_about	"Runaway children, Fiction"

format because the record-based structure of MARC requires considerable duplication of metadata and relies on free-text (literal) data rather than unique identifier (URI) data. This limitation of MARC (the inability to show open-ended relationships easily) parallels a key strength of RDF that is seen in both the data modeling and system scalability arenas. For this reason, RDF is said to allow n-ary or n-way relationships. This simply means that predicates and objects can themselves be resources with their own predicates and objects and that, in this way, a single resource can accurately track complex relationships between itself and external resources.

For the rest of our exploration of the building blocks in this chapter, we will use the book *The Adventures of Huckleberry Finn* by Mark Twain, first published in 1884 by Chatto & Windus Press, as an example. The MARC record for this book can be located in the Library of Congress online catalog. By extracting metadata from this record and recoding it following RDF principles, we will demonstrate the value of both the node/edge data model and the use of unique identifiers (URIs) over free-text (literal) values. A few such statements are shown in table 2.1, including authorship statements, publishing statements, and topical statements. This first extraction of metadata and representation as graph statements using a table structure does not vary much from the MARC data model. In addition, table 2.1 and figure 2.2 use literal values rather than URIs, so figure 2.2 implements only part of the RDF model. Subsequent examples will show considerable enhancement to the current bibliographic data structure and a more complete implementation of metadata in RDF.

MARC record for Adventures of Huckleberry Finn
http://lccn.loc.gov/35020965

This table is shown as a visualized graph in figure 2.2. While some liberties have been taken with the predicate composition, these predicates roughly relate to MARC fields such as the author, publisher, publication date, and subject. This model is largely accurate, but because we did not translate our literal values to URIs, we are stuck using the title of the book as the primary identifier. In doing so, we are relying on the concept of "Main entry" from traditional cataloging practice and, as you might expect, this would create difficulties for a title like ours in which there are many identical printings and editions.

In order to address this issue, we will select a unique identifier that points to a location that has more information about this title. For now, let's make that identifier the Uniform Resource Locator (URL) to the metadata record in the Library of Congress catalog. The results of this are seen in figure 2.3. The URL has been used as the Uniform Resource Identifier (URI) for the resource. The title has been become an object in a new statement with the predicate `has_title`. This simple shift is important because it helps us aggregate all of the statements about this work under a single URI.

Having replaced the resource identifier with a URI, it is also productive to find identifiers for the other descriptive metadata. As we will see in later chapters, there is a growing number of sources for these URIs. In some cases, we want to link to an external vocabulary (e.g., resource type). In other cases, we may want to link to other resources (e.g., a related work, an author, or a subject heading). Our rather simple set of statements in table 2.1 includes subject and name authority references, temporal references, and some miscellaneous values that may not have a defined vocabulary. It is okay to use literal values when necessary. In fact, at some point a literal value is required so a human can understand the content of the record! For simplicity's sake, we will use the Library of Congress's Linked Data Service Authorities and Vocabularies for our URIs. After replacing the literals for name and subject authorities, our data is starting to look a bit more "linkable." The results are seen in table 2.2.

Table 2.2 shows a simple view of this data with both the subject and object values replaced with URIs. For now, we will not convert other values like "Printed book" to a URI, although there are LOVs that would facilitate the translation of this value to an RDA

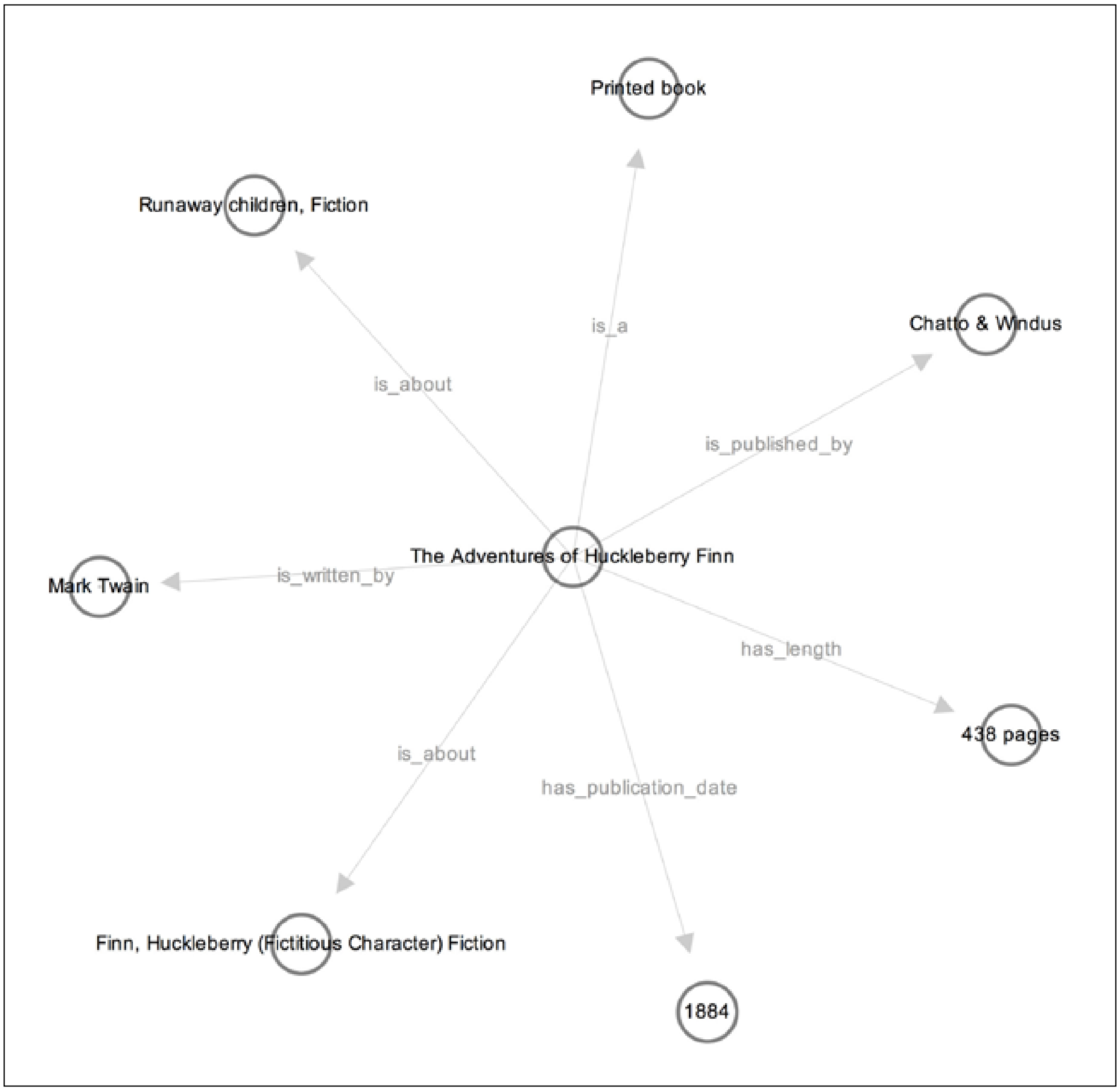

Figure 2.2
RDF statements about the book *The Adventures of Huckleberry Finn*

LOV-compliant series of triples. Even though we have not yet modified our predicates, we can already see that URIs would help us aggregate statements for a given work without any ambiguity, and URIs to topics, authors, and other subjects would help a computer perform detailed cross-searching and query expansion operations.

Having substituted URIs for our subject and predicate values, we can also use URIs to identify predicates as well. This has a number of benefits, including unambiguously aligning predicates with vocabularies and metadata schemas. Removing ambiguity by absolutely identifying predicates increases interoperability and the reliability of automated processing of data. For example, this level of reliability can enable low-level inferencing of unstated relationships between resources. In best-case scenarios, this also leads to reuse of shared vocabularies for predicate URIs.

While most LOD metadata employs multiple vocabularies and metadata schemas for description, for the sake of simplicity this exploration of RDF will use only a single metadata schema, Dublin Core. In table 2.3, the constructed predicate names have been mapped to elements from the Dublin Core vocabulary.

Each of these external vocabulary names (e.g., `dc:title`) employs both a namespace and an

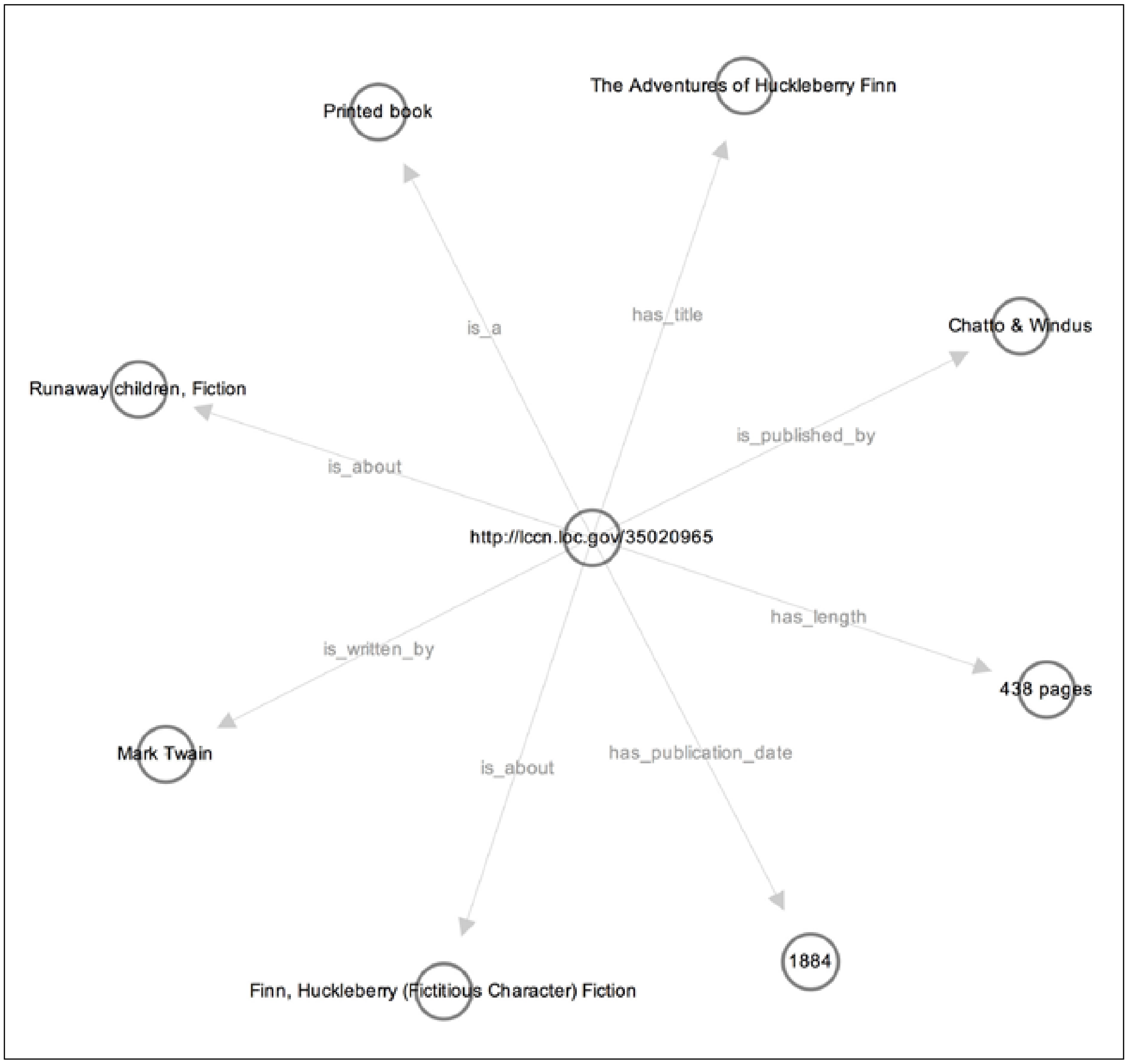

Figure 2.3
RDF statements about the book *The Adventures of Huckleberry Finn* with the LoC permalink used as the URI

element name. While we will be exploring namespaces in more detail in the discussion of serialization of RDF, it is appropriate to point out that namespaces are a built-in function of XML that allows the shortening of names from long URIs like `http://purl.org/dc/elements/1.1/title` to `dc:title`. This is accomplished through the use of the xmlns attribute. An example of how this is implemented can be seen below in list 2.1.

The RDF Abstract Model

In our relatively brief exploration of RDF, we introduced the triple (figure 2.1) as the building block of RDF statements and explored the notion of URI-based statements that facilitate the building of complex relationship descriptions using externally referable standards and vocabularies. Not surprisingly, the best practice guidelines for RDF conform to common practice in library and information science, including the need to reuse standards when available, the benefit of authorities, and the need to granularly and unambiguously describe content.

As we found when comparing MARC and RDF data models, RDF allows a more granular approach to description when working with complex one-to-many relationships. As stated at the beginning of our RDF exploration, this data model is a core building block

Table 2.2
Statements updated to reflect URIs representing the literal values extracted from the MARC record

http://lccn.loc.gov/35020965	is_a	"Printed book"
http://lccn.loc.gov/35020965	is_written_by	http://id.loc.gov/authorities/names/n79021164
http://lccn.loc.gov/35020965	has_publication_date	"1884"
http://lccn.loc.gov/35020965	has_length	"438 pages"
http://lccn.loc.gov/35020965	is_published_by	http://id.loc.gov/authorities/names/n85242407
http://lccn.loc.gov/35020965	is_about	http://id.loc.gov/authorities/subjects/sh2008103799
http://lccn.loc.gov/35020965	is_about	http://id.loc.gov/authorities/subjects/sh2008110345

Table 2.3
Predicates mapped to the Dublin Core external vocabulary

Predicate name	External vocabulary name	Associated object literals or URIs
is_a	dc:type	"text"
is_written_by	dc:creator	http://id.loc.gov/authorities/names/n79021164
has_publication_date	dc:date	"1884"
has_length	dc:extent	"438 pages"
is_published_by	dc:publisher	http://id.loc.gov/authorities/names/n85242407
is_about	dc:subject	http://id.loc.gov/authorities/subjects/sh2008103799
is_about	dc:subject	http://id.loc.gov/authorities/subjects/sh2008110345
title	dc:title	"The Adventures of Huckleberry Finn"

not only of the Semantic Web but also of LOD. In order to understand the role that other components such as vocabularies and linked data endpoints play, we will first turn to ways of writing down RDF, also known as serialization. As we turn our attention to the serialization options for RDF, let's remind ourselves of the core rules of RDF as stated by Yu.[10] RDF optimizes information for machine processing. RDF can make use of both explicit and implicit statements, and because RDF creates metadata using non-ambiguous URIs, it is possible to aggregate statements about a resource from multiple sources. Although the data model gets us part of the way to these goals, the actual structures and the systems that can interpret them make the goals happen.

Building Block 2: Content Rules

Content rules are an important element of LIS-based metadata schemas. Real-world examples of content rules include RDA and AACR2 and, generally speaking, dictate the process by which data is extracted from a resource for cataloging. Content rules work hand-in-hand with metadata schemas (e.g., once you extract a title of a resource, you would store it in a title field associated with a metadata schema), but they are separate entities.

The content creation principles of LOD using RDF are to some extent content rule–agnostic. While RDF has strict guidelines on how statements are defined and how relationships are established, it is not prescriptive on the content that is cataloged into the RDF statements. In fact, this is a strength of the RDF data model. By employing a graph structure that does not rely on predefined properties, the linked data model can be easily extended to include other descriptive data.

Because RDF does not have built-in rules for how to represent literals except for simple syntax and localization commands, it relies on the use of externally defined vocabularies and ontologies for content. This is seen below in list 2.1, in which the Bibliographic Ontology is used to define the resource type being cataloged. The Bibliographic Ontology, also known by its recommended namespace BIBO, is used in a number of LODLAM projects, including the Library of Congress's Chronicling America.

The Bibliographic Ontology
http://bibliontology.com

Chronicling America
http://chroniclingamerica.loc.gov

Building Block 3: Metadata Schema in LOD

Metadata schemas are perhaps the best understood building block in metadata circles. Schemas relate to the rules governing the structure and content of a metadata record or triple. Metadata schemas and content rules are closely related building blocks, but it is important to distinguish between them, given their different goals. For example, while the Dublin Core metadata schema defines the property subject with guidelines for vocabularies to use, the process of applying that vocabulary to a resource requires content rules that are not part of the metadata schema definition.

RDF Schema

Our RDF examples helped demonstrate the suitability of the data model for representing resources but also demonstrated the importance of external vocabularies to use in the process of description and representation. In addition, however, RDF has structures that allow us to show more complex relationships between resources. These structures are part of the specification called RDF Schema (RDFS), which provides an internal vocabulary to help establish the rules and structure of the assertions made about resources.[11] This means that RDFS is a vocabulary-building language rather than a language of description. By using these structures, we can enhance a computer's ability to infer relationships between resources, helping us bridge from resource description to knowledge representation.

As with the external vocabularies used in previous examples, the RDFS vocabulary can be referenced with the URL `http://www.w3.org/2000/01/rdf-schema#`. The RDF Schema is comprised of three main structure types: classes, properties, and utility properties. A key feature of RDFS is the ability to show parent/child relationships. In RDFS, these are known as super and sub relationships and extend to classes and properties (`rdfs:subClassOf`, `rdfs:subPropertyOf`). RDFS also enables definition of data types (`rdfs:Datatype`), the scope (`rdfs:domain`), and the set of acceptable values (`rdfs:range`) for a given property or class. Both domain and range have more sophisticated counterparts in the Simple Knowledge Organization System (SKOS) and Web Ontology Language (OWL) vocabularies that address higher order relationships, including transitive/nontransitive and joint/disjoint relationships. In addition to parent/child relationships, data typing, and domain/range definitions, RDFS includes a "see also" property (`rdfs:seeAlso`), a definition property (`rdfs:isDefinedBy`), and a labeling property (`rdfs:label`).

Vocabulary Building Blocks: SKOS and OWL

Metadata implemented using RDF fulfills the basic elements of LOD, but if we want to track the role of a resource in a complex knowledge system, we need to use vocabularies developed using an advanced version of RDFS. SKOS and OWL are such systems that support the creation of sophisticated vocabularies (e.g., LOV), also known as taxonomies and ontologies. SKOS and OWL help support interoperability by defining the relationships between resources in a way that goes far beyond a simple predicate statement (e.g., `has_subject`). This is particularly important when bringing together data sources from different LOD datasets, as OWL properties such as `owl:sameAs`, `owl:differentFrom`, and `owl:disjointWith` enable automated inferencing from harvested data.

In addition to supporting resource relationship description in LOD, the use of SKOS and OWL structures in LOV provides the detailed relationship features required to migrate complex taxonomies, classification schemas, and ontologies. The Metadata Authority Description Schema in RDF (MADS/RDF) defined by the Library of Congress is presented as an OWL ontology. A full explanation of SKOS and OWL are outside of the scope of this issue, but there are a number of good tutorials associated with the Protégé platform that provide an introduction to the standard via hands-on activities.

Library of Congress MADS/RDF Primer
www.loc.gov/standards/mads/rdf

Protégé platform
http://protege.stanford.edu

An Example RDF-Based Schema for Metadata in LAM: Open Archives Initiative Object Reuse and Exchange (OAI-ORE)

The OAI-ORE specification grew from a vision of enabling open archiving of metadata while handling the issues and complications that arise when gathering metadata from multiple sources.[12] Previous work with the Protocol for Metadata Harvesting (OAI-PMH) revealed challenges associated with gathering data from multiple sources, including issues of provenance, versioning, object structure, and metadata quality. The OAI community continues to be active in the development of these specifications, including the Protocol for Metadata Harvesting (PMH) specification, the ORE specification, and a new project designed to facilitate resource synchronization across systems called ResourceSync.[13] ORE has seen increasing adoption in digital library systems[14] and is a foundational component in new LOD metadata specifications such as the Europeana Data Model (EDM).[15]

Michael Witt's 2010 work on OAI-ORE in *Library Technology Reports* provides a detailed introduction to the standard, and readers completely unfamiliar with it are encouraged to start there.[16] A brief overview of the ORE specification is warranted in this chapter,

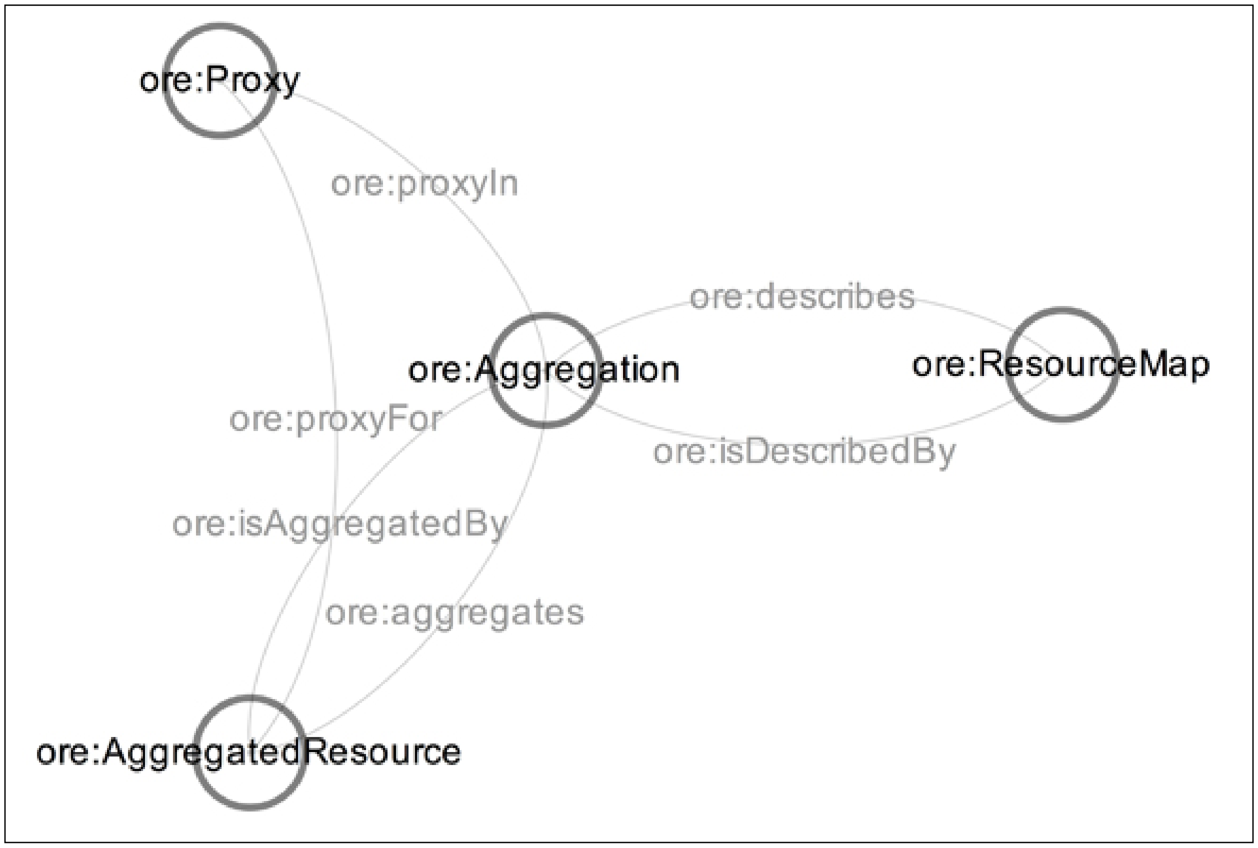

Figure 2.4
A simple graph of the relationships between the four main ORE entities

given its use in the EDM investigated in chapter 3.

The ORE specification consists of four main entities, an Aggregation, an Aggregated Resource, a Resource Map, and a Proxy. The primary objective of the ORE ontology is to provide a model for describing resources with appropriate differentiation between a primary object and its surrogates as well as differentiation between versions of resources.

The `ore:Aggregation` entity collects resources in the form of a direct relationship to one or more Resource Maps, one or more Aggregated Resources, or one or more Proxies. Aggregations are a top-level entity in an ORE collection. Resource Maps collect resources and their associated structural and descriptive statements. These entities have bidirectional properties that show relationships. For example, an Aggregation is described by (`ore:isDescribedBy`) a Resource Map, and a Resource Map describes (`ore:describes`) an Aggregation.

Likewise, an Aggregation aggregates (`ore:aggregates`) an Aggregated Resource, and an Aggregated Resource may be described by a Proxy (`ore:Proxy` and `ore:ProxyFor`). In the ORE specification, Proxies can represent an Aggregated Resource and are primarily used when the context of the assertions made is not needed. Figure 2.4 provides a graphical representation of the basic relationships between these entities as well as the properties (predicates) that connect them.

The relationships demonstrated in figure 2.4 are the basic relationships available in the ORE model and do not, for example, show how an Aggregation may aggregate multiple resources. For the purposes of our exploration of LOD systems, it is enough to understand the four entities in the ORE model and the properties that connect them.

Building Block 4: Serializations of LOD

Serializations are also commonly called encoding models in LAM metadata communities. Serialization refers to methods for writing out a resource to a physical or digital storage medium. Some common serializations that we will explore in this chapter include XML, JSON, and N3. Serializations are often tied to a data model and in the case of MARC were also closely tied to the metadata schema.

The application of the RDF abstract model requires the use of a series of metadata elements that, in RDF, are referred to as the RDF vocabulary. This vocabulary lives at the URI `http://www.`

Table 2.4
List of RDF element type, element names, and general use

RDF element type	RDF element	General purpose
Syntax	rdf:RDF	The root element of an RDF/XML document
Syntax	rdf:Description	Facilitates the definition of the “subject” of an RDF triple. General use is rdf:Description about=”URI to subject”>
Syntax	rdf:ID	This attribute provides a method for shortening RDF syntax by allowing the use of relative URIs. This attribute can be used in conjunction with the xml:base attribute.
Syntax	rdf:about	This attribute essentially stores the “subject” of a triple
Syntax	rdf:parseType	This attribute enables a simplified syntax for the creation of blank nodes
Syntax	rdf:resource	
Syntax	rdf:nodeID	This attribute supports the naming of blank nodes
Syntax	rdf:datatype	This attribute allows the definition of the data type of the object value when literals are used in an RDF statement. Datatype values conform to the XMLSchema datatypes.
Class	rdf:Property	
Class	rdf:XMLLiteral	
Property	rdf:type	This element allows an explicit definition of the type of resource being described and enables cross-resource inferencing
Property	rdf:value	This element enables the definition of a literal in context of a unit of measure
Property	_n (n is any value > 1)	

`w3.org/1999/02/22-rdf-syntax-ns#` and typically is defined with the namespace prefix `rdf:` in RDF documents. The RDF vocabulary consists of four groups of elements: syntax, class, properties, and resources. Table 2.4 provides a quick reference for RDF elements that may be useful in examples.

RDF/XML looks and works like all other XML documents. The sample RDF/XML document in list 2.1 builds on the metadata presented in table 2.2.

List 2.1
Basic RDF/XML file for *The Adventures of Huckleberry Finn*

```
 1 <?xml version="1.0"?>
 2 <rdf:RDF xmlns:rdf="http://www.
   w3.org/1999/02/22-rdf-syntax-
   ns#"
 3 xmlns:dc="http://purl.
   org/dc/elements/1.1/"
   xmlns:bibo="http://purl.org/
   ontology/bibo/">
 4 <rdf:Description
   rdf:about="http://lccn.loc.
   gov/35020965">
 5 <dc:title>The Adventures of
   Huckleberry Finn</dc:title>
 6 <dc:publisher
   rdf:resource="http://id.loc.
   gov/authorities/names/
   n85242407"/>
 7 <dc:subject
   rdf:resource="http://id.loc.
   gov/authorities/subjects/
   sh2008110345"/>
 8 <rdf:type
   rdf:resource="bibo:Book"/>
 9 </rdf:Description>
10 </rdf:RDF>
```

List 2.1 shows a simple RDF record that implements four triples from table 2.3. Let’s review this RDF/XML document line by line. Line 1 establishes the document as an XML document. Line 2 opens the `rdf:RDF` element and is the root element of the document. The `rdf:RDF` element also defines three namespaces, its own (`rdf`), the namespace for Dublin Core (`dc`), and the namespace for the Bibliographic Ontology (`bibo`). These namespaces are used as the predicates for triples on lines 5–8. The `rdf:Description` element in line 4 establishes the subject of the triple statements that are located in lines 5–8. Line 5 uses a literal value as the subject, while lines 6, 7, and 8 use resource references.

This simple RDF/XML document can be seen in graph form by entering the document into the W3C validator online. A modified view of this graph is shown in figure 2.5. For readability, the URLs in the RDF/XML record were shortened to the salient parts of the subject, predicate, and object.

RDF Validator
www.w3.org/RDF/Validator

This simple exploration of the RDF/XML serialization format is not intended to be a comprehensive introduction. In addition to the verbose and overly simplified document shown here, RDF/XML also supports the range of XML-based standards, including entity definition and use, expanded use of QNames for URI simplification, and the incorporation of XML-based vocabularies like XML Datatype. In addition, there are a number of shortcuts and abbreviated ways of creating RDF. For readability, the examples presented in this chapter follow a more verbose model. For more details on RDF syntax, please consult the RDF core documentation on the W3C website.

Finally, the RDF data model contains a number of elements and attributes that support the definition of groups of things in either an ordered (`rdf:Seq`) or unordered (`rdf:Bag`) structure. Each of these elements is a type of container, and the individual elements of a bag or sequence are a series of `rdf:li` elements. RDF also includes the concept of a collection, a defined list of fields that can be configured so that additions cannot be made to the list. This is an important part of creating ontological structures in which the RDF document defines the "necessary and sufficient" elements of a resource. This concept will be covered in more detail in our exploration of the Web Ontology Language (OWL).

RDF also supports the creation of administrative metadata using the `rdf:statement` and associated elements. This process is known as reification and can include the addition of external vocabularies to add contextualizing information. More information about reification is available in the RDF Primer.

RDF Notation-3/N3, Turtle, and N-Triples

N3 is a text-based, non-XML serialization of RDF that is more readable for humans but still follows a syntax that can be easily processed by computers. The N3 Primer provides an extensive overview of the RDF implementation of N3, including the definition of RDF-based rules. In contrast, the Terse RDF Triple Language (Turtle) is a scoped-down version of N3 that simply focuses on RDF statements. N-Triples are a dissemination format that focuses on serializing triples one line at a time. For this reason, N-Triple files are commonly used to disseminate RDF statements, given their compact syntax.

N3 Primer
www.w3.org/2000/10/swap/Primer.html

Being able to read Turtle is a good on-ramp to understanding the RDF query standard SPARQL because it uses much of the same syntax.

Applications of RDFa and JSON-LD: Microformats and Microdata

While much of the work in the LAM metadata community centers on creating large-scale solutions to complex metadata problems, there is also a movement to incorporate document-embedded metadata systems such as Schema.org, RDFa, JSON-LD, Open Graph Protocol, and other standards into LAM-related web data. There are many benefits to these formats, including simplicity, native interoperability with popular web services like Facebook, and easy scalability.[a] For example, OCLC's incorporation of Schema.org metadata into WorldCat creates new opportunities for developers to write data-rich web services and enables search engine providers to use bibliographic data in indexing and retrieval.[b]

Microformats are easy to implement as they build on existing metadata data schemas (e.g., hCard, vCard) and serialization methods (e.g., HTML). Likewise the standards are easy to implement, and because the standards integrate into web pages naturally, they are an important first step in equalizing computational and human understanding of web data.[c]

At the same time, however, microdata solutions are typically an end-user-facing product and will not have the same impact as the fundamental restructuring of LAM metadata for back-end systems. In order to make microdata solutions scalable, this fundamental shift needs to occur. In fact, the current suite of microdata solutions tends to cater to specific solutions. Facebook's Open Graph Protocol, for example, is widely used on the Web and has been implemented in LAM-related sites including LibraryThing and IMDB.

a. Erik Mitchell, "Linked Data Publishing for Libraries, Archives, and Museums: What Is the Next Step?" *Journal of Web Librarianship*, forthcoming.

b. Ted Fons, Jeff Penka, and Richard Wallis, "OCLC's Linked Data Initiative: Using Schema.org to Make Library Data Relevant," *Information Standards Quarterly* 24, no. 2/3 (Spring/Summer 2012): 29–33, doi:10.3789/isqv24n2-3.2012.05.

c. Liyang Yu, *A Developer's Guide to the Semantic Web* (Heidelberg; New York: Springer, 2011), 95.

Schema.org
http://schema.org

LibraryThing
http://librarything.org

IMDB
www.imdb.com

Figure 2.5
Graph display of simple metadata record as represented in List 2.1

RDF element type	RDF element	General purpose
Class	rdfs:Resource	The Root class of an RDFS vocabulary
Class	rdfs:Class	This term is used to define a class in a vocabulary
Class	rdfs:Literal	
Class	rdfs:Datatype	
Property	rdfs:range	
Property	rdfs:domain	
Property	rdfs:subClassOf	
Property	rdfs:subPropertyOf	
Property	rdfs:label	
Property	rdfs:comment	
Utility Property	rdfs:seeAlso	
Utility Property	rdfs:isDefinedBy	

The general structure of a Turtle statement follows the subject, predicate, object syntax but using a single-line sentence structure. List 2.2 shows the RDF record from list 2.1 represented in Turtle. Each statement has its own line and is organized into subject, predicate order. Notice that each statement ends with a period and that the `@prefix` element is used to define the Dublin Core (`dc`) namespace on line 1.

List 2.2
RDF resource expressed in Turtle

```
1 @prefix dc: <http://purl.org/dc/
  elements/1.1/>.
2 @prefix madsrdf: <http://www.
  loc.gov/mads/rdf/v1#>.
3 <http://lccn.loc.gov/35020965>
  dc:title "The Adventures of
  Huckleberry Finn".
4 <http://lccn.loc.gov/35020965>
  dc:publisher <http://id.loc.
  gov/authorities/names/
  n85242407>.
5 <http://lccn.loc.gov/35020965>
  dc:subject <http://id.loc.
  gov/authorities/subjects/
  sh2008110345>.
6 <http://id.loc.gov/authori-
  ties/names/n85242407> a
  madsrdf:Authority.
```

Turtle also includes specialized tokens such as the letter a, which references the `rdf:type` element (`http://www.w3.org/1999/02/22-rdf-syntax-ns#type`). This shorthand allows the creation of `rdf:type` statements as seen in line 6 of list 2.2. There are further shorthand notations for Turtle, including the use of a semicolon to concatenate statements where only the objects change (`<subject>`; `<predicate>` `<object>`, `<object>`.) and the definition of blank nodes using square brackets `[ ]`. For more information on proper structuring of Turtle statements, see the W3C documentation or one of our recommended resources.

W3C: Turtle
www.w3.org/TR/turtle

RDFa

RDFa is mentioned as a serialization of RDF because it implements a lightweight version of the RDF model using HTML attributes. The W3C site has a robust primer for RDFa, which is commonly used to embed metadata in line with HTML data. The RDFa standard includes many of the same constructs in the RDF data model, including the type concept, support for namespaces, and structures to represent predicates (e.g., `property`, `rel`, and `rev`) and subjects and objects (e.g., the `about` attribute, depending on context).

RDFa Primer
www.w3.org/TR/xhtml-rdfa-primer

List 2.3 provides a basic RDFa record embedded in HTML. In this example, the `span` element is used to contain RDF statements in context of the free-text versions. In the first statement, identifying the work title on line 3, the `about` attribute serves as the subject pointer, the `rel` attribute serves as the predicate, and literal value in the `span` element serves as the object. On line 6, a similar structure exists, but rather than using the literal value following the `span` element, a URI is used as the object with the content attribute.

List 2.3
RDFa version of an HTML-based bibliographic description of *The Adventures of Huckleberry Finn*

```
1 <div xmlns:dc="http://
  purl.org/dc/elements/1.1/"
  xmlns:bibo="http://purl.org/
  ontology/bibo/">
2 <p>
3 <span about="http://
  lccn.loc.gov/35020965"
  rel="dc:title">The  Adventures
  of Huckleberry Finn</span>is a
4 <span about="http://lccn.loc.
  gov/35020965" rel="bibo:Book"
  />book printed in 1884 by
5 <span about="http://
  lccn.loc.gov/35020965"
  property="dc:publisher"
  content="http://id.loc.
  gov/authorities/names/
  n8524240"/>Chatto & Windus
  about
6 <span about="http://
  lccn.loc.gov/35020965"
  property="dc:subject"
  content="http://id.loc.
  gov/authorities/subjects/
  sh2008110345"/>Huckleberry Finn
7 </p>
8 </div>
```

Using the principles of web design, it would be a relatively simple process to turn these RDFa-rich HTML documents into both human- and computer-actionable text. While there is much more detail to RDFa, it is sufficient for the context of this *LTR* issue to be aware of its existence and use. There are a number of good tutorials and instructional resources available for more information on RDFa.[17] An alternative approach to using RDFa that focuses on embedding metadata in the web page is to use the Gleaning Resource Descriptions from Dialects of Languages (GRDDL) standard. The GRDDL standard is based on the perspective that contextual metadata, like author and publisher information, and rich links are best separated into highly structured XML files that can be modified for specific uses via XSLT transformations.[18] As with RDFa, the GRDDL standard is on the perimeter of the focus of this *LTR* issue, so we will not go into any more detail on it.

JSON-LD

JavaScript Object Notation for Linking Data (JSON-LD) is documented online. This serialization format implements some features of the RDF abstract model using JSON and is one of many JSON-serialized RDF models. JSON, an object-based notation employed on the Web, is popular because of its lightweight syntax and ability to be used as objects without any ingest or modification process. A limitation of the JSON format, however, is the lack of vocabulary alignment in the key/value data pairs. For example, list 2.4 shows our Mark Twain book data represented in regular JSON. While the record is very readable, there are no indicators to the vocabularies used to create the data and no URIs employed that would facilitate linking with external data.

JSON-LD documentation
http://json-ld.org

List of RDF serializations in JSON
www.w3.org/wiki/JSON%2BRDF

List 2.4
A basic JSON record with bibliographic data for a book

```
{
"The Adventures of Huckleberry
 Finn": {
"is_a": "Printed Book",
"is_written_by": "Twain, Mark",
"has_publication_date": "1884",
"has_length": "438 pages",
"is_published_by": "Chatto and
 Windus",
"is_about": "Finn, Huckle-
 berry (Fictitious Character)
 Fiction",
"is_about": "Runaway children,
 Fiction"
}
}
```

In contrast, JSON-LD provides support for a resource reference using a URI and provides a typing structure similar to `rdf:type` in the form of the `@context` property. The ability to work with JSON-LD as a native object within a programming language dramatically lowers the barriers to adoption, as no additional parsing libraries are required to access the data; it is viewed as a key enabler to encouraging LOD adoption.[19] JSON-LD is a W3C working draft and has been implemented in the DPLA application programming interface (API). More information about JSON-LD can be found in the documentation mentioned above, and the standard will be discussed in more detail in chapter 3.

List 2.5
A simple JSON-LD record with three descriptive fields for a bibliographic record

```
{
"@context": "http://json-ld.
  org/contexts/book.jsonld",
"@id": "http://lccn.loc.
  gov/35020965",
"Title": {
"@id": "http://lccn.loc.
  gov/35020965",
"name": "The Adventures of
  Huckleberry Finn"
},
"is_a": {
"@id": "http://purl.org/
  ontology/bibo#book",
"name":"Printed Book"
},
"is_written_by": {
"url":"http://id.loc.gov/
  authorities/names/n79021164",
"name":"Twain, Mark"
}
}
```

Building Block 5: Exchanging LOD

A key part of exchanging LOD is ensuring that the institution or community publishing the data has the right to do so. Publishing LOD in a community platform or via an aggregated disseminator poses unique challenges because it requires new provider agreements and a certain level of technical and community support so that only openly licensed data is made available as linked data.[20] At the same time, however, national- and international-scale services like Europeana and the Digital Public Library of America are essential in helping to build critical mass in the deployment and adoption of LOD, so these issues are both important and in scope. The Creative Commons Rights Expression Language (ccREL) is a common open vocabulary for expressing rights[21] and has been used in many LODLAM projects.

In addition to rights issues, however, the technical process of exchanging LOD involves query protocols like SPARQL Protocol and RDF Query Language (SPARQL). SPARQL, a W3C recommendation, consists of a set of specifications that govern the query structure and a protocol for querying and receiving data. SPARQL 1.1 supports results serialized in JSON, CSV, TSV, and XML. Building on the semantic relationships defined in LOD, SPARQL focuses on providing "answers" as opposed to "documents." As a result, SPARQL enables deep graph searching across LOD sources and itself returns RDF data, meaning that a SPARQL query is itself a new LOD data source.

SPARQL 1.1 Overview
www.w3.org/TR/sparql11-overview

SPARQL is an important element of the Semantic Web and has been implemented by the Europeana Foundation on the LOD service. For more details on SPARQL queries, I recommend Heath and Bizer's book *Linked Data* as well as Yu's work on the Semantic Web.[22]

Conclusion

In this chapter, we have explored the building blocks of LOD/LOV and have considered the roles that our five building blocks of metadata play in supporting these specifications. We found that RDF, while more cumbersome for humans, possesses many advantages for computational work and as such opens new doors for scaling the creation, dissemination, management, and use of metadata. We also found that vocabulary specifications like RDFS, SKOS, and OWL enable computational reasoning on LOD/LOV data.

Our brief exploration of OAI-ORE, SPARQL, and JSON-LD may feel out of place, but these are important concepts to understand as we embark on our case studies in chapter 3. For example, OAI is a core construct in the EDM, and JSON-LD is the preferred serialization format of the DPLA API. The building blocks we discussed here have parallels in the systems that are required to implement them. RDF is commonly stored in a database called a triple store (e.g., Apache Fuseki), which is constructed of resource descriptive statements contextualized using external vocabularies, metadata schemas, and ontologies; is queried using SPARQL; and is serialized using one of many output streams including RDF/XML, JSON-LD, or even lightweight schemes like RDFa. While this may seem overwhelming, a key advantage to an LOD implementation is that the triple store can store data on any number of resources types or vocabularies, in essence eliminating the "silo" issues commonly associated with library metadata. In chapter 3 we will rely on our familiarity with these concepts as well as our understanding of metadata building blocks to examine three LOD systems.

Notes

1. Craig Willis, Jane Greenberg, and Hollie White, "Analysis and Synthesis of Metadata Goals for Scientific Data," *Journal of the American Society for Information Science and Technology* 63, no. 8 (August 2012): 1505–1520, doi:10.1002/asi.22683.
2. Joan C. Biella and Heidi G. Lerner, "The RDA Test and Hebraica Cataloging: Applying RDA in One Cataloging Community," *Cataloging and Classification Quarterly* 49, no. 7–8 (2011): 676–695, doi:10.1080/01639374.2011.616450; William E. Landis, "Plays Well with Others: DACS and CCO as Interoperable Metadata Content Standards," *VRA Bulletin* 34, no. 1 (2007): 97–103; Ilana Tolkoff, "The Path toward Global Interoperability in Cataloging," *Information Technology and Libraries* 29, no. 1 (March 2010): 30–39; Yuji Tosaka and Jung-ran Park, "RDA: Resource Description and Access—A Survey of the Current State of the Art," *Journal of the American Society for Information Science and Technology* 64, no. 4 (April 2013): 651–662.
3. Getaneh Alemu, Brett Stevens, Penny Ross, and Jane Chandler, "Linked Data for Libraries: Benefits of a Conceptual Shift from Library-Specific Record Structures to RDF-Based Data Models" (conference paper, IFLA Council and General Conference, Helsinki, Finland, August 11–17, 2012); Bernhard Haslhofer, Elaheh Momeni, Bernhard Schandl, and Stefan Zander, "Europeana RDF Store Report," Europeana Connect, Results and Resources, March 8, 2011, www.europeanaconnect.eu/documents/europeana_ts_report.pdf.
4. Tim Berners-Lee, "Linked Data," W3C website, last updated June 18, 2009, www.w3.org/DesignIssues/LinkedData.html.
5. Karen Coyle, "Linked Data Tools: Connecting on the Web," *Library Technology Reports* 48, no. 4 (May/June 2012); Bernhard Haslhofer and Antoine Isaac, "data.europeana.eu: The Europeana Linked Open Data Pilot," in *Metadata Harmonization: Bridging Languages of Description: 21–23 September 2011, the Hague, Netherlands*, ed. Thomas Baker, Diane I. Hillmann, and Antoine Isaac (Dublin, OH: Dublin Core Metadata Initiative, 2011), 94–104; Library of Congress, "The BIBFRAME Model: Vocabulary Updates," Bibliographic Framework Initiative website, accessed April 20, 2013, http://bibframe.org/vocab.
6. Berners-Lee, "Linked Data."
7. Tim Berners-Lee, "Tim Berners-Lee on the Next Web," TED video, 16:17, filmed February 2009, posted March 2009, www.ted.com/talks/tim_berners_lee_on_the_next_web.html.
8. Frank Manola and Eric Miller, eds., "RDF Primer," W3C Recommendation, February 10, 2004, www.w3.org/TR/rdf-primer.
9. Liyang Yu, "The Building Block for the Semantic Web: RDF," chapter 2 in *A Developer's Guide to the Semantic Web* (Heidelberg; New York: Springer, 2011), 19–86.
10. Ibid., 72.
11. Dan Brickley and R. V. Guha, eds., "RDF Vocabulary Description Language 1.0: RDF Schema," W3C Recommendation, February 10, 2004, www.w3.org/TR/rdf-schema.
12. Herbert Van de Somple and Carl Lagoze, "The Santa Fe Convention of the Open Archives Initiative," *D-Lib Magazine* 6, no. 2 (February 2000), doi:10.1045/february2000-vandesompel-oai.
13. Martin Klein, Robert Sanderson, Herbert Van de Somple, Simeon Warner, Bernhard Haslhofer, Michael Nelson, and Carl Lagoze, "ResourceSync Framework Specification—Beta Draft," February 1, 2013, www.openarchives.org/rs/0.5/resourcesync.
14. Michael Witt, "Object Reuse and Exchange (OAI-ORE)," *Library Technology Reports* 46, no. 4 (May/June 2010): 9.
15. Antoine Isaac and Robina Clayphan, eds., "Europeana Data Model Primer: Europeana 1.0," Europeana Professional website, October 26, 2011, http://pro.europeana.eu/documents/900548/770bdb58-c60e-4beb-a687-874639312ba5.
16. Witt, "Object Reuse and Exchange."
17. Dathan, "HTML5 Microdata, Microformats, and RDFa Tutorials and Resources," *WebsitesMadeRight.com* (blog), May 1, 2011, http://websitesmaderight.com/2011/05/html5-microdata-microformats-and-rdfa-tutorials-and-resources; Yu, *A Developer's Guide to the Semantic Web*.
18. Harry Halpin and Ian Davis, eds., "GRDDL Primer," W3C Working Group Note, June 28, 2007, www.w3.org/TR/grddl-primer.
19. Markus Lanthaler and Christian Gütl, "On Using JSON-LD to Create Evolvable RESTful Services," in *WS-REST '12: Proceedings of the Third International Workshop on RESTful Design* (New York: ACM, 2012), 25–32, doi:10.1145/2307819.2307827; Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, and Niklas Lindström, "JSON-LD 1.0: A JSON-Based Serialization for Linked Data," W3C Last Call Working Draft, April 11, 2013, www.w3.org/TR/json-ld-syntax.
20. Haslhofer et al., "Europeana RDF Store Report," 94.
21. Hal Abelson, Ben Adida, Mike Linksvayer, and Nathan Yergler, "ccREL: The Creative Commons Rights Expression Language," W3C Member Submission, May 1, 2008, www.w3.org/Submission/ccREL.
22. Tom Heath and Christian Bizer, *Linked Data: Evolving the Web into a Global Data Space*, Synthesis Lectures on the Semantic Web: Theory and Technology, ed. James Hendler (San Rafael, CA: Morgan & Claypool, 2011), doi:10.2200/S00334ED1V01Y201102WBE001; Yu, *A Developer's Guide to the Semantic Web*.