Chapter 21
Metadata Architecture in RESTful Design

Antonio Garrote Hernández and María N. Moreno García

Abstract Metadata is a key component of the REST architecture that can be used to provide additional information about web resources. The ultimate goal of metadata is to transform web resources into self describing information units that can be automatically processed by software agents. We review the main options present in the HTTP standard to provide metadata for web resources. We also review the main mechanisms proposed by standard organizations like the W3C and the IETF as well as by groups of practitioners to provide additional ways of associating metadata to resources. The connection between metadata and semantic web technologies is also explored. Finally the notion of resource and metadata discovery is also introduced and the main discovery technologies are reviewed.

Introduction

Metadata is one of the data elements in the REST web architectural style along with resources, resource identifiers, representations and control data (Fielding and Taylor 2000). In this context, metadata can be defined as "machine understandable information about web resources" (Berners-Lee 1998). This brief definition remarks the importance of metadata as the element of RESTful design enabling automatic processing of web resources. This aspect is often overlooked in the design of RESTful web services where the role of metadata is many times restricted to provide information about the syntax used in the resource representation. This "representation metadata" encoded as a media data type in a HTTP header is exchanged between HTTP parties in the content negotiation process in order to select a suitable representation for a certain web resource.

A. Garrote Hernández (✉)
University of Salamanca, Avenida Italia 29 4-A, Plaza de los Caídos, s/n, 37008, Salamanca, Spain
e-mail: agarrote@usal.es; antoniogarrote@gmail.com

E. Wilde and C. Pautasso (eds.), *REST: From Research to Practice*,
DOI 10.1007/978-1-4419-8303-9_21, © Springer Science+Business Media, LLC 2011

Nevertheless, effective metadata for a resource should not be restricted to the automatic selection of the parsing mechanism for a resource representation. It should provide a full description of the semantics of the resource that would make possible for a HTTP agent to automatically choose a way of processing the resource to accomplish some kind of functionality, sometimes different from the original functionality devised by the creator of the resource. The ultimate goal of the metadata layer is to transform web resources into self describing information units (Berners-Lee 1998). This same aim can be found in the core of metadata proposals like the W3C's semantic web stack of technologies or the microformats initiative.

From a RESTful point of view, the main concern about the resource metadata layer is to find suitable ways of integrating this new kind of data into the architectural components of the REST style of building web systems. Metadata must be regarded as any other kind of resource data, therefore it must be exposed in the expected manner to REST components and connectors.

On top of this RESTful foundation, further metadata based features can be built, for instance, in the same way HTTP provide a common mechanism to access resources, metadata authors can agree in the vocabulary used in the metadata of a resource and in the ways these metadata are encoded into resources. These features have the potential to grant major advantages in web design, like improved data interoperability and better functional and conceptual reusability of web resources.

Metadata in RESTful web services is getting increasingly important as data APIs built following RESTful architectural principles are becoming a central component in many modern web applications. These APIs are facing the same data interoperability and reusability challenges that metadata have the potential to solve.

Metadata in the Hyper Text Transfer Protocol

The HTTP protocol makes metadata a first class object in the protocol specification. The main place to store metadata in HTTP messages is the collection of HTTP headers sent in every HTTP request and response.

Entity headers can be classified in different categories. Some headers contain meta information about the representation of the requested resource being transferred in the entity body of the HTTP response. The most important representation HTTP header is the "Content-Type" response header that specifies the media type for the representation. A different kind of entity headers expose meta data about the resource rather than the representation being transferred. For example the "Allow" HTTP header contains the list of HTTP methods supported by the resource. Finally, control data headers contain information necessary for the correct interaction between client and server. This information is not directly related to the representation of the resource. The Cache-Control header is an example of this kind of headers.

The main mean to provide the semantics of the resource representation retrieved by the client is the media type returned in the "Content-Type" HTTP header. This

header is used in the context of the content negotiation mechanism specified in the HTTP protocol (Fielding et al. 1999). Using this mechanism a HTTP agent can expose the list of preferred representations it is willing to accept using the "Accept" HTTP header and the HTTP server can return the list of available representations for the resource using the "Content-Type" HTTP header. Examining this information both parties can agree which representation for the resource will be finally transferred from server to client.

Each media type imposes a certain syntax for the representation of the resource and hints some of the semantics of the resource being requested. media types can be classified into application specific media types and generic media types (Allamaraju 2010). Different media types provide a different degree of semantic information about the resource. Application specific media types specify well defined semantics for the representation of the resource encoded in the HTTP entity body. For example, the media type "image/jpg" provides enough semantics for HTTP agents to process the HTTP resource representation and visualize the entity data according to the specification of the JPEG image format. Application specific media types often have a very limited support for adding arbitrary metadata about the resource being encoded in the provided representation. On the other hand, media type headers for more generic media types, like "application/xml" or "application/json" provide very little semantic information about the resource but the format of the associated representation can contain any kind of metadata and information about the resource being retrieved.

A common practice for better describing the semantics of a representation is to provide an augmented media subtype in the Content-Type HTTP header. This way a particular vocabulary and semantics are stated to be used besides a generic encoding mechanism. media types like "image/svg+xml" or "application/atom+xml" give the agent a better understanding of the semantics of the representation built on top of a generic description mechanism (Allamaraju 2010).

The list of public media types is supported by IANA and is publicly available `http://www.iana.org/assignments/media-types/`. When designing a RESTful API is a good practice to look for a public representation format that suits the resources being exposed through the API. If no public media type matches the intended use of the representation at the application level, designers can consider the creation of new media types that will give HTTP agents a hint of the semantics for the provided representation of the resource.

The support for media types in the HTTP protocol metadata provides a mechanism to transfer the syntax and, to a certain degree, the semantics of the resource representation together with the representation in HTTP messages. Taking this into account, HTTP messages can be considered to be self-describing. Any HTTP connector can inspect the metadata of the HTTP message and take decisions about how to process the content of the message as an opaque packet of data. In a similar way, HTTP agents can decide the best way to process the message data based on the semantics of the stated media type.

Nevertheless, the level of semantic description of the content enabled by the use of media types is not enough to automate complex tasks involving the processing

of web resources. Media types just provide HTTP agents with a reference to the semantics of the representation but it does not support a mechanism for describing these semantics. HTTP agents must have support in advance for the media type of the resource representation since it is impossible for agents to acquire support for an unknown type just from the media type declaration present in the HTTP "Content" header.

An additional problem is the rigidity of the media type standard to provide custom semantics for a specific resource. Private custom media type headers can be used for particular applications but their semantics cannot be automatically retrieved and processed by third party HTTP agents.

Different solutions to provide the semantics of the resource representation have been proposed. IETF RFC2068 (Fielding et al. 1997) of the HTTP protocol, superseded by IETF RFC2616, proposed the inclusion of a "Link" header (Conolly 1999) that could be used to link an associated resource to the resource representation being retrieved. This header has been used by different metadata retrieval proposals to associate metadata with a web resource. The use of an additional HTTP header has the advantage of not requiring the HTTP agent to retrieve the whole document in order to check and process the associated metadata. This can be accomplished with a single HEAD HTTP request that will retrieve the headers of the HTTP message. One major drawback of using the HTTP HEAD method is that it is not widely supported by server and client implementations of the HTTP protocol.

Using fixed, well known URIs where information about web resources in a domain could be retrieved, as proposed in IETF RFC5785 (Nottingham and Hammer-Lahav 2010), is another possible alternative for the association of metadata to resources that is being used in different metadata mechanisms.

WEBDAV extensions to the HTTP protocol introduced a different approach to the retrieval of metadata for a resource using an additional HTTP verb "PROPFIND" that make WEBDAV (Goland et al. 1999) enabled servers return all the metadata information associated with a resource.

Nevertheless, none of these mechanisms offers a solution for all the possible use cases involving the retrieval of metadata. In these cases, the common approach is to embed metadata in the resource representation or link the metadata from the representation if the resource representation supports hyperlinking. This approach can be problematic because HTTP agents must retrieve the full representation and process it to retrieve the metadata.

Metadata as a Formal System

There are different proposals to expose the description of the semantics of a web resource. Some of them have been proposed by standards organizations like the W3C, others have originated in the industry and among practitioners. As a consequence, designers of RESTful APIs face many different, often overlapping, options when choosing a mechanism to add semantic metadata to the representation

of the resources exposed in web services.Nevertheless, there are some important points that must be taken into consideration and that should be addressed by any description mechanism.

First, semantic metadata should be regarded as a set of formal assertions about the resource being described (Berners-Lee 1998). Metadata must conform to a formal logic system with its own semantics that impose a certain trade-off between expressivity and processing complexity. This is a mandatory requirement to build truly extensible mechanisms for semantic description.

Any metadata proposal lacking formal soundness is unsuitable for the development of automated HTTP agents involving complex tasks like logic inference. More simple tasks like integration of resource information from different sources will also benefit from the coherence imposed by a formal description system.

Another major feature of a good semantic description mechanism is its capacity to interact with web technologies and follow RESTful architectural principles. One well known REST principle is "hypermedia as the engine of the application state" (Fielding 2000). It is important for any description mechanism to use the capacity of hyperlinking to reference descriptions from resources using URIs. URIs also offer a good namespace for creating unique identifiers for metadata that can be shared between agents in a web scale. The capacity of linking these descriptions from different resource descriptions, allows agents to retrieve the description of the semantics of a resource from the representation of the resource in a standard and RESTful way.

Finally, another important feature of any metadata description mechanism that must be taken into account is its openness and extensibility. The web itself is a system with a great degree of openness and extensibility arising from their basic components like the use of URIs and hyperlinks.

W3C standards for the semantic web meet all these requirements. Other proposals like the Microformats `http://microformats.org` initiative offer a simpler metadata mechanism at the price of limiting the expressivity and the extensibility of the solution. Nevertheless, microformats have been applied successfully to different application domains and have obtained broad adoption.

Embedding Metadata in Web Resources Using Microformats

Microformats, as a mechanism for the description of resource semantics, is an extension of the idea of semantic markup. Semantic markup is a set of design guidelines enforcing the use of HTML building blocks to express the meaning of the data contained in the document rather than the presentation information of those data.

To accomplish that goal, semantic markup principles enforce the use of HTML tags with precise semantics for each information element in the document. If there is no standard HTML tag with the required semantics for the information included

in the document, a generic HTML container block like “div” or “span” can be used and the semantics of the information could then be added as the value of standard HTML attributes.

The Microformats proposal ultimate goal is to define standard ways of structuring HTML tag elements and property vocabularies in order to describe semantic information so it can be easily reused by humans in the design process of HTML documents and by software agents automatically processing web resources.

The main characteristics of the microformats initiative can be summarized as follows:

- Use of HTML structure plus a plain vocabulary to define semantics
- Community driven
- Embeddable in HTML, XHTML, Atom, RSS, and XML documents
- Focus on simplicity, reuse and minimalism

Currently, there is a list of ten microformats considered to be stable, including hCalendar for expressing calendar events, hCard used to represent people and organizations, or rel-license to state content licenses in a document.

The following example shows sample HTML code including hCalendar microformat markdown. The “event”, “summary”, “dtstart” and “location” property values are part of a controlled vocabulary used by the microformat to add semantic information to the data contained into standard span HTML tags. The structure of the HTML tags containing the vocabulary values in the class HTML properties is also prescribed by the hCalendar specification.

```
<span class="vevent">
 <span class="summary">The microformats.org site
 was launched</span>
 on <span class="dtstart">2005-06-20</span>
 at the Supernova Conference
 in <span class="location">San Francisco, CA, USA</span>.
</span>
```

In order to make microformatted HTML document self-describing, an HTML profile can be linked to the document describing the microformat, using the XMDP microformat itself. Profiles can be declared in the head of the HTML element using the “profile” attribute or the link tag with a “rel” attribute with value “profile”. They can be just referenced in the HTML document body using a HTML anchor element with a “rel” property with value “profile”.

Microformats provide simple mechanism to add semantics to HTML documents. They are easy to use with present technologies and have gained wide adoption. Unfortunately, microformats have important limitations. The main issue of the Microformats proposal is the use of plain literals to express the properties and relations of the HTML data. A literal used in one microformat to express a property can collide with any other use of the same literal in a different microformat with a similar or completely different meaning. Literal properties are not unique and they must be defined in a single flat namespace. The use of URIs would have avoided this

problem since XML namespaces could be used to prevent name collisions among metadata identifiers but present important difficulties for their integration into plain HTML documents.

Another problem with the microformats is extensibility. It is impossible to add a custom microformat to a HTML document. The only description mechanism available for metadata is XMDPP and at the present moment is more suitable for documenting microformats for humans than for description of arbitrary microformats that could be automatically parsed by agents. Besides, the lack of unique identifiers for properties makes difficult to reuse properties between different microformats.

Resource Description Framework in Attributes and the W3C Semantic Stack of Technologies

Resource Description Framework in attributes (RDFa) (Pemberton et al. 2008) is the standard mechanism proposed by the W3C to embed semantic metadata into XHTML and HTML (Adida et al. 2010) documents. It has a similar aim to the Microformats proposal but is built on top of the stack of semantic technologies proposed by the W3C.

The foundation of the W3C semantic specifications is the Resource Description Framework (RDF) (Beckett 2004). RDF provides a mechanism to make statements about resources, understanding as a resource everything that can be identified using a URI (Miller and Manola 2004). RESTful resources are just a small subset of the resources that can be addressed in RDF. RDF is simultaneously many things:

- An abstract data model to describe metadata as a labeled graph of resources and properties
- An extensible vocabulary for data description based on the use of URIs
- A formal system with well defined semantics

RDF introduces the notion of properties, identified by URIs, that serve as a relation between resources acting as subject and object of the property. A collection of these statements, known as triples, defines a graph of relations between resources that can be serialized to different concrete syntaxes: XML, N3, TTL, etc.

RDFa is a standard proposal for a concrete syntax for the RDF data model that makes possible embedding RDF graphs in XHTML and other XML based documents. The final result is similar to the microformats proposal with some important differences:

- RDFa properties are stated as full URIs instead of plain literals. RDFa map URIs to literal strings using XML namespaces to build “compact URIs” or CURIES. The use of XML namespaces in the name of the relations prevents the clash between relation names. It also makes possible to reuse existing RDF vocabularies and allows an easy extension of the assertions contained in RDFa annotated XML documents.

- RDFa is currently defined on top of XHTML since it requires the extensibility capacities of this language. RDFa annotations can be used in HTML documents but these documents will not validate. Specification of RDFa annotations for HTML documents is an undergoing effort.

Since RDFa is just an encoding for RDF, the whole stack of semantic technologies standardized by the W3C can be used to provide semantics for the resource representation.

Schema and ontology languages like RDFS and OWL can be used to describe and link the description of the metadata used in the RDFa annotated document. For instance, vocabularies like the Dublin Core Metadata Initiative vocabulary `http://dublincore.org/`, or Friend of a Friend (FOAF) `http://www.foaf-project.org/` can be used in RDFa annotated documents in a similar way as microformats, like XFN or rel-license, are used in annotated HTML documents.

The following listing shows the example previously introduced for the hCalendar microformat modified to use RDFa and the iCalendar vocabulary.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.0//EN"
         "http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd">
<html xmlns:cal="http://www.w3.org/2002/12/cal/ical#">

<span typeof="cal:Vevent">
 <span property="cal:summary">The microformats.org site
   was launched</span>
 on <span property="cal:dtstart" content="20050620">
   2005-06-20</span>
 at the Supernova Conference in
 <span property="cal:location">San Francisco, CA, USA</span>.
</span>
```

The vocabulary referenced by the URI “http://www.w3.org/2002/12/cal/ical#” has been added as a XML namespace declaration in the HTML tag. RDF properties from this namespace have been used as values for HTML attributes like “instanceof” or “property”. The values are specified as CURIES using the prefix “cal” previously declared in the XML namespace declaration.

RDFa tries to maintain the appeal of the Microformats proposal as a simple mechanism for adding semantics to XHTML documents while preserving the formal semantics and data model of the W3C stack of semantic technologies.

Extracting Metadata from Representations Using Transformations

Microformats and RDFa propose an approach to the description of metadata consisting of embedding metadata in the resource representation. Once a HTTP agent has retrieved the annotated representation, it can use a well defined algorithm

to extract the actual metadata from the representation. These metadata can link to additional metadata, for example, a document containing the OWL description of the class and properties used to annotate the representation.

Gleaning Resource Description from Dialect of Languages (GRDDL) (Connolly 2007) is a W3C recommendation describing an alternative mechanism to add metadata to web resources. The starting point of GRDDL is the existence of a variety of possible representations for web resources. Many of them are XML based: plain XHTML documents, Atom feeds, etc. In many occasions, modifying these representations to embed semantic metadata using microformats or RDFa is not possible. One possibility to add semantic metadata to the resource is to provide an additional representation for the resource containing only the metadata for the resource being exposed, for example, a RDF document, that can be retrieved by HTTP agents using HTTP content negotiation. This approach has the drawback of creating and maintaining the additional representation.

Resource authors using GRDDL link an algorithmic transformation capable of generating a faithful rendition of the XML representation in RDF, instead of directly linking the metadata of the resource. GRDDL recommended way of describing transformations is using XSL Transformations (XSLT).

Linking GRDDL transformations from XML based resource representations can be accomplished just adding the GRDDL namespace declaration to the document and adding a "grddl:transformation" property pointing at the transformation. Transformations for whole XML dialects can be linked using the "grddl:namespaceTransformation" property. XHTML documents can be used with GRDDL adding the GRDDL namespace as a metadata profile and linking the transformation using a link tag with a "transformation" value for the "rel" attribute.

One of the main use cases for GRDDL is to transform XHTML documents annotated using microformats into RDF representations using some equivalent vocabulary. The following example shows a variant of the hCal microformat example considered before. In this example, a GRDDL transformation renders the same RDF graph that was embedded in the RDFa annotated version.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
  lang="en">
  <head profile="http://www.w3.org/2003/g/data-view">
    <link rel="transformation"
          href="http://www.w3.org/2002/12/cal/glean-hcal"/>
  </head>
  <body>
    <span class="vevent">
     <span class="summary">The microformats.org site
     was launched</span>
     on <span class="dtstart">2005-06-20</span>
     at the Supernova Conference
     in <span class="location">San Francisco, CA, USA</span>.
    </span>
  </body>
</html>
```

GRDDL offers also a good opportunity for reusability. The GRDDL community has collected transformations for many existing microformats as well as for other non HTML based dialects e.g. the Atom format that can be directly linked by authors of these representations. GRDDL can be used by Microformats publishers as an easy way to provide an alternative representation for the metadata of a resource that can be used by agents working with W3C semantic technologies.

Resource and Metadata Discovery

Services discovery can be described as the process allowing two automated agents to start some kind of useful interaction. In the process both parties discover which kind of services are offered by the other. In the context of web resources, we can talk about two kind of discovery process: service discovery and descriptor discovery (Hammer-Lahav 2010). Service discovery deals with agents looking for services with a certain capability. Descriptor discovery involves a software agent trying to discover the capabilities supported by a resource. The availability of metadata is one of the services that can be detected in the service discovery mechanism, metadata themselves can also be used to make possible the discovery of other kind of capabilities as well as enabling service discovery.

In previous sections, different ways of adding metadata to a resource have been examined. Previously reviewed mechanisms like the embedding of metadata into resource representations using microformats and RDFa or linking a GRDDL transformation capable of generating metadata from a representation, present the problem of not being automatically discoverable by HTTP agents. Agents must obtain the full representation of the HTTP resource and process it in order to detect the presence of metadata description mechanisms.

Metadata discovery protocols and specifications try to solve these limitations providing two features:

- Standard protocols for the automatic retrieval of resource's metadata.
- Shared vocabularies for the description of resource services.

Discovery mechanisms must have certain desirable features that are also common to any other metadata mechanisms (Umbrich et al. 2009):

- Self declarative: the resource must be capable of linking the resource description.
- Direct accessible: the resource description must be retrievable without requesting the resource being described.
- Compliant with web architecture.
- Scale to web size.
- Extensible: the description mechanism must allow authors of resource descriptors to add arbitrary metadata in the description.
- Granular: a resource descriptor can be used to describe a single resource or a set of resources.

The Protocol for Web Description Resources (POWDER) is a W3C standard recommendation for the discovery of metadata associated with a web resource. POWDER documents consist of two different parts: an attribution block describing the author, date and validity of the description and a collection of "description resources" containing the actual metadata. A single POWDER document can contain metadata for different resources in a single domain. Each description resource is composed of two parts, a set of URIs being described and a collection of assertions. The assertions contain a collection of plain tags or a RDF fragment.

POWDER documents can be linked using the "describedby" property from the POWDER namespace. HTML documents can link a POWDER description using link tags located in the head of the document. In order to avoid the necessity of processing the whole resource representation, a different recommended mechanism to link a POWDER profile is to use the not official "Link" HTTP header.

An alternative mechanism for metadata discovery is the combination of the LRDD/XRD standards. Link-based Resource Descriptor Discovery (LRDD) protocol is an IETF draft standard proposal for linking easily discoverable metadata to web resources.

LRDD defines three different metadata sources:

- Hyperlinks using the "link" tag in the representation of a resource.
- The "Link" HTTP header.
- Host metadata situated in standard locations.

LRDD shares with POWDER the hyperlink and "Link" header mechanisms for metadata discovery. It also adds the possibility of inserting metadata at standard locations for each domain. The IETF "host-meta" standard proposal specifies a single point for each domain to add metadata for resources located at that domain. The entry point URI is built from the "host-meta" suffix added to the standard "/.well-known/" prefix defined in the IETF RFC 5785. LRDD profiles collected from these three link sources must be described using the OASIS standard draft Extensible Resource Descriptor (XRD) as the format for the metadata of the described resources.

Conclusions

The current state of the metadata architecture in the design of RESTful web services is still a work in progress.

As RESTful APIs are becoming more and more usual and a higher degree of automation and interconnection is required, the necessity of a standard metadata layer is becoming more evident. In this chapter we have reviewed some of the main technologies trying to address the architectural issues introduced by the integration of semantic metadata in the HTTP protocol.

Four main techniques have been introduced to associate metadata with resource representations:

- Providing metadata as an alternative representation for the resource that can be retrieved using content negotiation.
- Embedding metadata within the resource representation.
- Linking metadata annotations from the resource representation.
- Linking metadata annotations from the headers of the HTTP message or a well known URI location.

Microformats have been used as a simple mechanism to embed metadata in HTML documents. Microformats is the most extended technology to add explicit semantics to HTML representations but this technology is lacking in extensibility and presents a serious drawback due to the use of a flat namespace to describe properties instead of standard XML namespaces and URIs. To solve these problems, RDFa presents an alternative mechanism to embed metadata in XHTML documents in a compliant way with W3C standards for the semantic web. RDFa allows publishers of web resources to add the full potential of semantic web technologies to their service APIs, like the ontology description language OWL and standard vocabularies like FOAF at the same time that it preserves the simplicity and low entry barrier of the Microformats proposal.

A bridge between both semantic annotation mechanisms can be found in the GRDDL W3C recommendation. GRDDL provides the means for linking an algorithmic transformation to a resource representation that will render as a result of its application, the equivalent RDF triples graph. GRDDL transformations can be reused by resource publishers and a whole collection of GRDDL transformations for many microformats is already available. GRDDL makes possible the integration of annotation mechanisms, metadata vocabularies and description mechanisms.

Metadata discovery is another open problem in order for autonomous HTTP agents to be able to identify the available metadata in the services exposed by API providers. Automatic discovery of these metadata will open new ways of interaction between agent and servers. The POWDER W3C recommendation and the XRD/LLDR protocol stack try to offer solutions to this problem, specifying linking mechanisms that do not require the HTTP client to retrieve and process the full representation of the resource. Metadata can be linked to the HTTP message using the Link HTTP header that can be retrieved using the HEAD HTTP method without downloading the HTTP message body or can be placed into well defined standard URIs that can be queried by clients, for example, in the ".well-known/host-meta" path. They also define standard ways of adding arbitrary metadata to services and the mechanism for its retrieval.

Metadata is already making possible the automatic interaction between HTTP agents in web protocols like the OAuth authentication mechanism. In the nearly future, better metadata support in RESTful APIs will make possible to automate new kind of interactions offering important benefits to users. The emergent properties as well as the interoperability capacities offered by semantic metadata will also make possible to build more robust HTTP agents and use these APIs in new ways not anticipated by their original designers.

References

B. Adida, M. Birbeck, and S. Pemberton. HTML+RDFa 1.1, support for rdfa in html4 and html5. W3C working draft, W3C, October 2010. http://www.w3.org/TR/rdfa-in-html/.

S. Allamaraju. *RESTful Web Services Cookbook*. O'Reilly, February 2010.

D. Beckett. RDF/xml syntax specification (revised). W3C recommendation, W3C, February 2004. http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/.

T. Berners-Lee. Design issues of web architecture. 1998.

D. Connolly. Gleaning resource descriptions from dialects of languages (GRDDL). W3C recommendation, W3C, September 2007. http://www.w3.org/TR/2007/REC-grddl-20070911/.

H. Conolly. An Entity Header for Linked Resources, October 1999.

R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, Irvine, California, 2000.

R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2068 (Proposed Standard), January 1997. Obsoleted by RFC 2616.

R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), June 1999. Updated by RFCs 2817, 5785.

R. T. Fielding and R. N. Taylor. Principled design of the modern web architecture. In *ICSE '00: Proceedings of the 22nd international conference on Software engineering*, pages 407–416, New York, NY, USA, 2000. ACM.

Y. Goland, E. Whitehead, A. Faizi, S. Carter, and D. Jensen. HTTP Extensions for Distributed Authoring – WEBDAV. RFC 2518 (Proposed Standard), February 1999. Obsoleted by RFC 4918.

E. Hammer-Lahav. LRDD: Link-based Resource Descriptor Discovery, Draft rev 6. Internet Draft, May 2010.

E. Miller and F. Manola. RDF primer. W3C recommendation, W3C, February 2004. http://www.w3.org/TR/2004/REC-rdf-primer-20040210/.

M. Nottingham and E. Hammer-Lahav. Defining Well-Known Uniform Resource Identifiers (URIs). RFC 5785 (Proposed Standard), April 2010.

S. Pemberton, B. Adida, S. McCarron, and M. Birbeck. RDFa in XHTML: Syntax and processing. W3C recommendation, W3C, October 2008. http://www.w3.org/TR/2008/REC-rdfa-syntax-20081014.

J. Umbrich, M. Hausenblas, E. Hammer-Lahav, and E. Wilde. Discovering resources on the web. DERI technical report, DERI, August 2009.