Adjustments to Enhance Linked Data Platform Resource-Oriented Discovery

David Wood and James Leigh

3 Round Stones, Washington, DC 20036, USA
{david|james}@3roundstones.com

Abstract. This position paper suggests some desired adjustments to a future version of the Linked Data Platform (LDP) 1.0 specification. Specifically, we argue for the inclusion of features to enhance resource-oriented discovery of LDP services and LDP RDF Sources (LDP-RS), and to adjust the LDP Indirect Container definition based on our experience developing and using the Callimachus Project.

Keywords: Linked Data Platform, LDP, RDF, Linked Data, W3C, Callimachus

1 Introduction

This position paper suggests some desired features to be added to a future version of the Linked Data Platform 1.0 specification[1]. Specifically, we argue for the inclusion of features to enhance resource-oriented discovery of LDP services and LDP RDF Sources (LDP-RS), and to adjust the LDP Indirect Container definition based on our experience developing and using the Callimachus Project[4].

Callimachus supports iterative resource-oriented discovery of "folder" and "file" resources, which are roughly equivalent to LDP Containers and LDP Resources. Callimachus folders may be discovered and traversed using the Callimachus REST API[2] in a manner that is similar in concept to LDP. The Callimachus REST API was extended in late 2014 to expose Callimachus folders as LDP Indirect Containers (LDP-IC), and support the description of Callimachus file resources as LDP Resources (LDPR).

General metadata regarding a Callimachus instance may be found by requesting an HTTP OPTIONS response from the top-level URL. HTTP *link:* headers with relationship (*rel*) attributes provide a list of URLs necessary for recursive resource-oriented discovery of both LDP Containers and Resources. HTTP GET requests on URLs returned by the OPTIONS request allows one to retrieve a list of top-level resources via either an XML atom feed or an RDF description.

The Callimachus resource-oriented discovery mechanism is described in the Callimachus Reference Documentation[3].

The Callimachus team has noted practical issues of resource-oriented discovery when using LDP 1.0. The following section suggests several slight modifications to the LDP specification.

2 Suggested Additions to the LDP Specification

Both LDP 1.0 and the current Callimachus implementation support, but do not mandate the use of the Vocabulary of Interlinked Datasets (VoID)[5] to present commonly defined RDF metadata regarding a service to the Web. It would be useful for a future version of LDP to mandate (via an RFC 2119 **MUST**) so that any LDP client would know exactly where to look to discover an extensible metadata description of both a given LDP service and to descend in a resource-oriented fashion through LDP Containers.

VoID is an RDF Schema vocabulary, which is already suggested for us by LDP implementations. It is noted that some VoID extensions specific to LDP are likely to be necessary.

We suggest that the use of VoID with appropriate extensions could fulfill the same requirement as Callimachus' link headers in an HTTP OPTIONS response. Alternatively, we would be equally as happy if a future version of LDP adopted the existing Callimachus resource-oriented discovery mechanisms or an iteration of them.

Secondly, we suggest that LDP-RS resources be allowed to be directly discoverable via some mechanism. The LDP 1.0 specification already allows corresponding Non-RDF Sources (LDP-NR) to return a discoverable reference to its associated LDP-RS in section 5.2.8.1 of the LDP 1.0 specification.

We propose that an LDP-RS should be discoverable via a *link:* header in an OPTIONS request to their primary topic.

Thirdly, we propose that the URL for the primary topic of an LDP-RS be present in HTTP responses when the LDP-RS URL is resolved. We propose that the primary topic URL be provided in a *link* header using a reverse relationship (*rev*) attribute. Callimachus currently provides this two way linking via link headers to facilitate automated discovery of relevant URLs.

Finally, it would be convenient for implementors of LDP Indirect Containers if such containers did not require the use of the ldp:contains predicate. The ldp:contains predicate would be redundant and unnecessary if an LDP-RS could be discovered from the objects of the ldp:hasMemberRelation properties (i.e. the primary topics of a LDP-RS).

An example of similar handling in the current Callimachus implementation is provided in the last example in section 6.4 “Browsing Content” of the Callimachus Reference Documentation[3].

References

1. Speicher, S., Arwe, J., and Malhotra, A. (eds). Linked Data Platform 1.0. W3C Recommendation, 26 February 2015. Retrieved 19 April 2015 from http://www.w3.org/TR/2015/REC-ldp-20150226/.
2. 3 Round Stones Inc. Callimachus Reference Documentation, version 1.4, Chapter 6. “Callimachus REST API”. Retrieved 19 April 2015 from http://callimachusproject.org/docs/1.4/callimachus-reference.docbook?view#Callimachus_REST_API.

3. 3 Round Stones Inc. Callimachus Reference Documentation, version 1.4, Section 6.4 “Browsing Content”. Retrieved 19 April 2015 from http://callimachusproject.org/docs/1.4/callimachus-reference.docbook?view#Browsing_content.
4. Wood, D. and Leigh, J.: The Callimachus Project. Retrieved 19 April 2015 from http://callimachusproject.org.
5. Alexander, K., Cyganiak, R., Hausenblas, M. and Zhao, J. (eds). Describing Linked Datasets with the VoID Vocabulary. W3C Interest Group Note, 3 March 2011. Retrieved 19 April 2015 from http://www.w3.org/TR/2011/NOTE-void-20110303/.