Identity Crisis in Linked Data

Ora Lassila (Nokia)
Ryan McDonough (Nokia)
Susan Malaika (IBM)

October 31, 2011

1 Introduction

One of the cornerstones of Web Architecture [8] is the ability to *identify* resources (for purposes of description, manipulation, etc.). W3C's guidance on this matter is very clear: All resources "worth talking about" should have a URI as their globally unique identifier. This is an architecturally clean approach wrought with naïve idealism. In this paper, we explore the issue of identity, specifically in the context of the Semantic Web [4] and Linked Data [5], although – as we will demonstrate – the issue is not limited to these technologies.

Some of the specific issues we have observed are described below. Obviously, the issues are not entirely independent of one another and there is some overlap between them.

2 Identity vs. Location

URIs as IDs vs. URLs as *queries*: The best practice in Semantic Web and Linked Data is to use "http:" URIs as resource identifiers, regardless whether these resources are accessible over HTTP.[1] This leads to the confusion whether these URIs are actually URLs used to *access* said resources (i.e., whether they are *locators* or queries).

This issue is a source of much confusion, especially in the context of RESTful services [6] where (typically) the "identity" of a resource is embedded into the access URL. The industry would benefit from some kind of guidance or "best practices" documentation about this topic.

We observe many different ways how *identities* and *queries* are conflated and confused with one another:

Query Parameter Identity with Internal ID: On many sites, you see URLs fashioned in the following way: `http://example.com/things?id=13`. Here, the application is typically exposing a URL that queries a datastore for the identity of a thing by it's internally ID via a URL query parameter.

[1]One might argue that one contributing factor to this problem is the inclusion of the *access protocol* to resource addresses (e.g., HTTP vs. HTTPS).

URL as a Query: Using a path parameter: `http://example.com/things/13`. The query string is replaced by the last path segment in the URL. This syntax can be used as both a URI and a URL, if there is a representation for the URL at the given host. This syntax *might* be acceptable to the Linked Data community.

Query URLs are not opaque. The consumer constructs the URL according to the query specification for the relevant resources, incorporating identifiers or properties that represent the desired resources.

Query Parameter Identity with URI: This one has two forms: *internal* and *external*, we'll start with external first; if we are storing data with a known URI (e.g., DBPedia data) in our datastore, but we want to expose it via some URL, we could use the following URL:

`http://somesite.com/browse?desc=http%3A//example.com/things/13`

To many, it is obvious that this URL is a query into the system for data that was created or *identified* by some third party. What trips people up is when they have to create a URL for their own data using the same convention:

`http://example.com/browse?desc=http%3A//example.com/things/13`

Because both the (query) URL and the (identity) URI start with "http", the immediate assumption is that that they both *must* be dereferenceable. Web developers typically have an issue with this kind of URL syntax, since it just "feels wrong" to have an HTTP URL point to what looks like another HTTP URL. If we were using URNs as identity, rather than HTTP URIs:

`http://example.com/browse?desc=urn:com:example:things:13`

this might be less of an issue. Using "protocol URIs" leads to confusion.

3 Missing or Ambiguous Identification via URI

Many resources that need to be described (say, using Semantic Web or Linked Data technologies) in fact *do not* have a URI, or if a URI is selected it is not obvious how and does not lead to a *single* URI being picked for any particular resource [9]. Again, it may be idealistic to assume that every resource has a URI, but most objects in this world can be addressed *somehow*, either via an *intentional* address (one that sufficiently describes some subset of the object's attributes) or via some other unique identifier that happens not to be a URI, e.g., a book's ISBN number, a person's SSN [7]. Note, though, that in the case of "unique" identifiers, these identifiers may not in fact be completely unique.[2]

In the Semantic Web context, OWL's *inverse functional properties* [10, section 3.3.5] can be utilized if some other kind of unique identifier than URI is available for objects. Use of IFPs, however, is predicated on some level of reasoning support. It is also unclear how this approach could be used in the context of REST services.

[2]For example, ISBN numbers are being "recycled".

4 Versioning

Versioning has largely been ignored in many W3C specifications. In case of resources that must support versioning, should the resource identifier contain the version somehow, or is it an orthogonal attribute. More broadly, there are “lifecycle” issues with resources (e.g., moving an active resource to an archive at the end of its life), and these issues affect how resources are identified and/or located. It is unclear, however, whether version numbers (or other version identifiers) should be part of URIs or more broadly part of any identification scheme.

5 Stability (or Lack Thereof) of URIs

W3C also idealistically proclaims that “Cool URIs don’t change” [3], but in reality, they do. It is therefore necessary to consider the stability of addresses and possible mitigation approaches (PURLs and other redirect mechanisms). In fact, the general idea of “resolution” of addresses may be worth our attention[3] and may indeed help with many issues, including the “lifecycle” changes of a resource.

Redirect mechanisms such as PURL typically take effect on resource access through the network. However, identifiers such as URIs have to be stored in databases that underpin the resources. Databases have their own query interfaces. If URL’s rather than URI’s are used, then redirect mechanisms are needed for database query content, e.g., if a URL is referenced, and for the stored data to ensure accurate execution of database queries.

Note that OWL’s identity relation `owl:sameAs` [10, section 4.2] could be used as a way to *declare* that some URIs could be “mapped” onto one another, but similarly to IFPs, the use of this require some inference support.

Funny enough, ”Cool URIs don’t change” [3] really should have been called ”Cool *URLs* don’t change” because all the examples given are URLs, not so much URIs. URLs, and also URIs for that matter, are at the mercy of the entity that owns the domain name. Company acquisitions and reorganizations will inevitably cause a URL to change (due to consolidation of web properties, etc.), but do the URIs need to change? If we identified data using URIs, the URIs could live on, but publicly facing URLs might not.[4]

The problem is amplified with personal data, and in some “semantic desktop” systems [11, for example] URIs were being generated using a pattern like ”`file:///C:/My Documents/`...” Obviously, many users will have that ID prefix, so some other URI scheme is needed.

Also related to URL redirection is the well-known issue of “hash vs. slash” [2] and the related W3C TAG decision known as “httpRange-14”.

6 Conclusions

The ambiguity in handling identity delays Linked Data and Semantic Web deployments as implementers try to comply with W3C guidelines and fill in any gaps themselves, resulting in

[3]It is a well-known fact that “*we can solve any problem by introducing an extra level of indirection*” [1].

[4]E.g., consider various identitied of Adobe’s “Flash” technology: `http://www.futurewave.com` vs. `http://macromedia.com/software/flash/` vs. `http://adobe.com/flash`.

a variety of approaches – potentially in a non-interoperable way. Additional frameworks and infrastructures are sometimes used to create persistent URL systems, but there is no agreement on how and when these frameworks should be used, and how they affect or are affected by REST query mechanisms, versioning, etc.

References

[1] http://en.wikipedia.org/wiki/Fundamental_theorem_of_software_engineering.

[2] http://www.w3.org/wiki/HashVsSlash.

[3] “Cool URIs don’t change”. http://www.w3.org/Provider/Style/URI.

[4] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001.

[5] C. Bizer, T. Heath, and T. Berners-Lee. Linked Data – The Story So Far. *International Journal of Semantic Web and Information Systems*, 5(3):1–22, July-September 2009.

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

[7] T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool, 2011.

[8] I. Jacobs and N. Walsh. Architecture of the World Wide Web, Volume One. W3C Recommendation, World Wide Web Consortium, Cambridge, MA, December 2004.

[9] O. Lassila. Identity Crisis and Serendipity. Invited talk at the W3C Advisory Committee meeting in Edinburgh, Scotland, May 2006.

[10] D. L. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview. W3C Recommendation, World Wide Web Consortium, Cambridge, MA, Feb. 2004.

[11] L. Sauermann. The Gnowsis: Using Semantic Web Technologies to Build a Semantic Desktop. Master’s thesis, Technische Universität Wien, 2003.