Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Technical Memo
TM-07-01

Cool URIs for the Semantic Web

Leo Sauermann
DFKI GmbH
leo.sauermann@dfki.de

Richard Cyganiak
Freie Universität Berlin
richard@cyganiak.de

Max Völkel
FZI Karlsruhe
voelkel@fzi.de

February 2007

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
67608 Kaiserslautern, FRG
Tel.: + 49 (631) 205-3211
Fax: + 49 (631) 205-3210
E-Mail: info@dfki.uni-kl.de

Stuhlsatzenhausweg 3
66123 Saarbrücken, FRG
Tel.: + 49 (681) 302-5252
Fax: + 49 (681) 302-5341
E-Mail: info@dfki.de

WWW: http://www.dfki.de

Cool URIs for the Semantic Web

Leo Sauermann
DFKI GmbH
leo.sauermann@dfki.de

Richard Cyganiak
Freie Universität Berlin
richard@cyganiak.de

Max Völkel
FZI Karlsruhe
voelkel@fzi.de

DFKI-TM-07-01

This work is the result of collaboration of the Freie Universität Berlin, FZI Karlsruhe and DFKI Kaiserslautern. It was created to help educating the use of the Semantic Web, we wish to thank the various reviewers, especially Chris Bizer and Gunnar Aastrand Grimnes. This work is partly supported by by the European Union IST fund (Grant FP6-027705, project Nepomuk).

ISSN 0946-0071

Cool URIs for the Semantic Web

Leo Sauermann
DFKI GmbH
leo.sauermann@dfki.de

Richard Cyganiak
Freie Universität Berlin
richard@cyganiak.de

Max Völkel
FZI Karlsruhe
voelkel@fzi.de

29.11.2006

Zusammenfassung

The *Resource Description Framework* RDF allows you to describe web documents and resources from the real world—people, organisations, things—in a computer-processable way. Publishing such descriptions on the web creates the *semantic web*. URIs are very important as the link between RDF and the web. This article presents guidelines for their effective use. We discuss two strategies, called *303 URIs* and *hash URIs*. We give pointers to several web sites that use these solutions, and briefly discuss why several other proposals have problems.

1 Introduction

The semantic web is envisioned as a decentralised world-wide information space for sharing machine-readable data with a minimum of integration costs. Its two core challenges are the distributed modelling of the world with a shared data model, and the infrastructure where data and schemas can be published, found and used. A basic question is thus how to publish information about resources in a way that allows interested users and software applications to find them.
On the semantic web, all information has to be expressed as *statements* about *resources*, like *who are the members of a company* or *what are their telephone numbers* or *who made this web page*. Resources are identified by *Uniform Resource Identifiers* (URIs, [3]). This modelling approach is the *Resources Description Framework* (RDF, [11]).
At the same time, web documents have always been addressed with *Uniform Resource Locators* (URLs). Confusingly, URIs and URLs share the same syntax, every URL is also a URI. This is good because it means we can easily make RDF statements about web pages, but it is also dangerous because we can easily mix up web pages and the things, or resources, described on the page.
In general, what URIs should we use in RDF? As an example, to identify the frontpage of the website of ACME Inc., we may use `http://www.acme.com/`.

But what URI identifies the company as an organisation, not a website? And do we have to serve any content – HTML pages, RDF files – at those URIs? In the remainder of this paper we will answer these questions. We explain how to use URIs for things that are not web pages, such as people, products, places, ideas, and ontology classes. We give detailed examples how the semantic web can (and we think: should) be realised as a part of the web.
We assume that you are familiar with the basics of the RDF data model [11]. We also assume some familiarity with the HTTP protocol [10]. Wikipedia's article [1] serves as a good primer.

2 URIs for Web Documents

Let us begin with an example. Assume that ACME Inc., the fictional company from many Warner Brothers movies, has a web site at `http://www.acme.com/`. Imagine, part of the site is a white-pages service listing the names and contact details of the employees. Alice and Bob both work at ACME. The structure of ACME's web site might thus be:

`http://www.acme.com/` – the homepage of ACME, Inc.

`http://www.acme.com/people/alice` – the homepage of Alice

`http://www.acme.com/people/bob` – the homepage of Bob

Like everything on the traditional web, these are *web documents*. Every web document has its own URI. Note that a web document is not the same as a file: A single web document can be available in many different formats and languages, and a single file, for example a PHP script, may be responsible for generating a large number of web documents with different URIs.
On the traditional web, URIs were used *only* for web documents – to link to them, and to access them in a browser. In short, to *locate* a web document – hence the term *URL* (Uniform Resource Locator). The notion of resource *identity* was not so important on the traditional web, a URL simply identifies whatever we see when we type it into a browser.

2.1 HTTP and Content Negotiation

Today's web clients and servers use the HTTP protocol [10] to request web documents and send back the responses. HTTP has a powerful mechanism for offering different formats and language versions of the same web document: *content negotiation.*
When a user agent (e. g. a browser) requests a URL, it sends along some HTTP headers to indicate what data formats and language it prefers. The server then selects the best match from its hard disk or generates the desired content on demand, and sends it back to the client. For example, a browser could send this HTTP request to indicate that it wants an HTML or XHTML version of `http://www.acme.com/people/alice` in English or German:

```
GET /people/alice HTTP/1.1
Host: www.acme.com
Accept: text/html, application/xhtml+xml
Accept-Language: en, de
```

The server could answer:

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Language: en
```

followed by the content of the HTML document in English. Content negotiation is often implemented with a twist: Instead of a direct answer, the server *redirects* to another URL where the appropriate version is found:

```
HTTP/1.1 302 Found
Location: http://www.acme.com/people/alice.en.html
```

The redirect is indicated by a special *status code*, here `302 Found`. The client would now send another HTTP request to the new URL. By having separate URLs for all versions, this approach allows web authors to link directly to a specific version.
RDF/XML, the standard serialisation format of RDF, has its own content type too, `application/rdf+xml`. Content negotiation thus allows publishers to to serve HTML versions of a document to traditional web browsers and RDF versions to semantic web-enabled user agents. And it allows servers to provide alternative RDF serialisation formats like N3 [5] or TriX [9].

3 URIs for Real-World Objects

On the semantic web, URIs identify not just web documents, but also real-world objects like people and cars, and even abstract ideas and non-existing things like a mythical unicorn. We call all these things *resources*.
Given such a URI, how can we find out what resource it identifies? We need some way to answer this question, because otherwise it will be hard to achieve interoperability between independent information systems. We could imagine a service where we can look up a description for a URI, similar to today's search engines. But such a single point of failure is against the web's decentralised nature. Instead, we should use the web itself – an extremely robust and scalable information publishing system – as a lookup service for resource descriptions. Whenever a URI is mentioned, we can look it up to retrieve a description containing relevant information and links to related data. This is so important that we make it our number one requirement for good URIs:

1. **Be on the web.** Given only a URI, machines and people should be able to retrieve a description about this URI from the web. Such a look-up mechanism is important to establish shared understanding of what a URI identifies. Machines should get RDF data and humans should get HTML. The standard web transfer protocol, HTTP, should be used.

Let's assume ACME Inc. wants to publish contact data of their employees on the semantic web so their business partners can import it into their address books. For example, the published data would contain these statements about Alice, written here in N3 syntax [5]:

```
<URI-of-alice> a foaf:Person;
    foaf:name "Alice";
    foaf:mbox <mailto:alice@acme.com>;
    foaf:homepage <http://www.acme.com/people/alice> .
```

What URI should we use instead of the placeholder `<URI-of-alice>`? Certainly not `http://www.acme.com/people/alice`, because that would confuse a person with a web document, leading to misunderstandings [6][8]: Is the homepage of Alice also named "Alice"? Has the homepage an email address? And why has the homepage a homepage? So we need another URI. Therefore our second requirement:

2. Don't be ambiguous. There should be no confusion between identifiers for documents (URLs) and resource identifiers. URIs are meant to identify only one thing, and one URI can't stand for both a web-retrievable document and another real-world object.

We note that our requirements seem to conflict with each other. If we can't use document URLs as resource identifiers, then how can we retrieve a description from the URI? The challenge is to find a solution that allows us to find the describing documents if we have just the resource's URI, using standard web technologies. This would look somewhat like Figure 1.
Another question is where to draw the line between traditional web documents and other, non-document resources. According to W3C guidelines, we may have a web document if *all its essential characteristics can be conveyed in a message* [13]. This is not a very precise definition. Our recommendation is to *err on the side of caution*: Whenever an object of interest is not clearly and obviously a document, then it's better to use two distinct URIs, one for the resource and another one for the document describing it.

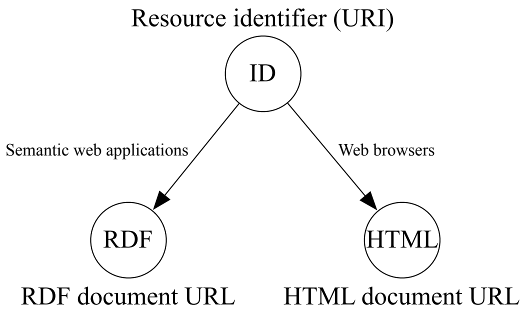

Abbildung 1: A resource and its describing documents

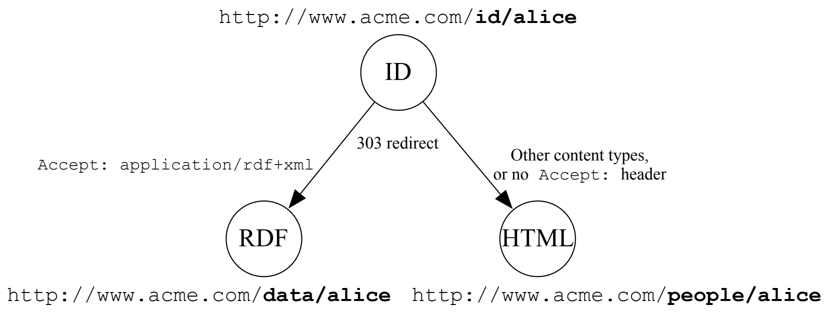
http://www.acme.com/id/alice
ID
303 redirect
Accept: application/rdf+xml
Other content types, or no Accept: header
RDF
HTML
http://www.acme.com/data/alice
http://www.acme.com/people/alice

Abbildung 2: The 303 URI solution

4 Two Good Solutions

There are two solutions that meet our requirements: *303 URIs* and *hash URIs*. Which one to use depends on the situation, both have advantages and disadvantages.

4.1 303 URIs

The first solution is to use a special HTTP status code, "`303 See Other`", to distinguish non-document resources from regular web documents. Since 303 is a redirect status code, the server can also give the location of a document that describes the resource. If, on the other hand, a request is answered with one of the usual status codes in the 2XX range, like `200 OK`, then the client knows that the URI identifies a web document. This practice has been embraced by the W3C's Technical Architecture Group in its *httpRange-14 ruling* [12].
If ACME adopts this solution, they could use these URIs to represent the company, Alice and Bob:

`http://www.acme.com/id/acme` – ACME, the company

`http://www.acme.com/id/bob` – Bob, the person

`http://www.acme.com/id/alice` – Alice, the person

The web server would be configured to answer requests to all these URIs with a 303 status code and a `Location` HTTP header that provides the URL of a document that describes the resource, as seen in Figure 2. The server could employ content negotiation (see Sec. 2.1) to send either the URL of an HTML description or RDF. Requests for HTML would be redirected to the web page URLs we gave in Section 2. Requests for RDF data would be redirected to RDF documents, such as:

`http://www.acme.com/data/acme` – RDF document describing ACME, the company

`http://www.acme.com/data/bob` – RDF document describing Bob, the person

`http://www.acme.com/data/alice` – RDF document describing Alice, the person

Each of the RDF documents would contain statements about the appropriate resource, using the original URI, e.g. `http://www.acme.com/id/alice`, to identify the described resource.

4.2 Hash URIs

The second solution is to use "hash URIs" for non-document resources. URIs can contain a *fragment*, a special part that is separated from the rest of the URI by a hash symbol ("#").
When a client wants to retrieve a hash URI, then it is required to strip off the fragment part before requesting the URI from the server. This means a URI that includes a hash cannot be retrieved directly, and therefore cannot identify a web document. We can use them to identify other, non-document resources, without creating ambiguity.
If ACME adopts this solution, then they could use these URIs to represent the company, Alice, and Bob:

`http://www.acme.com/data#acme` – ACME, the company

`http://www.acme.com/data#bob` – Bob, the person

`http://www.acme.com/data#alice` – Alice, the person

Clients will always strip off the fragment part before requesting any of these URIs, resulting in a request to this URI:

`http://www.acme.com/data` – RDF document describing ACME, Bob, and Alice

At this URI, ACME could serve an RDF document that contains descriptions of all three resources, using the original hash URIs to identify the resources.
Alternatively, content negotiation (see Section 2.1) could be employed to redirect from the `data` URI to separate HTML and RDF documents. This is a somewhat intricate affair because of the way HTML interacts with fragments: The redirect to the HTML document *must* send a `303 See Other` status code (otherwise a client could conclude that the hash URI represents a part of the HTML document). For the redirect to the RDF, `302 Found`, `303 See Other`, and `307 Temporary Redirect` are fine; a recent W3C Best Practices document uses 303 [14].
The hash URI solution, with and without content negotiation, is shown in Figures 3 and 4.

4.3 Choosing between 303 and Hash

Which approach is better? It depends. The hash URIs have the advantage of reducing the number of necessary HTTP requests. A family of URIs can share the same non-hash part. The descriptions of `http://www.acme.com/data#acme`, `http://www.acme.com/data#product123`, and `http://www.acme.com/data# product456` are retrieved with a single request to `http://www.acme.com/data`. There is a

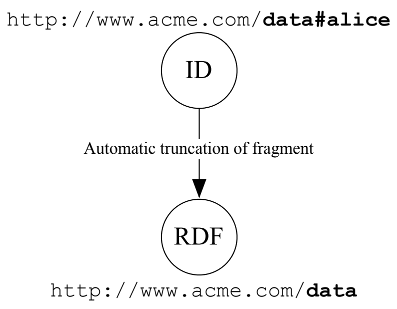
http://www.acme.com/**data#alice**
ID
Automatic truncation of fragment
RDF
http://www.acme.com/**data**

Abbildung 3: The hash URI solution without content negotiation

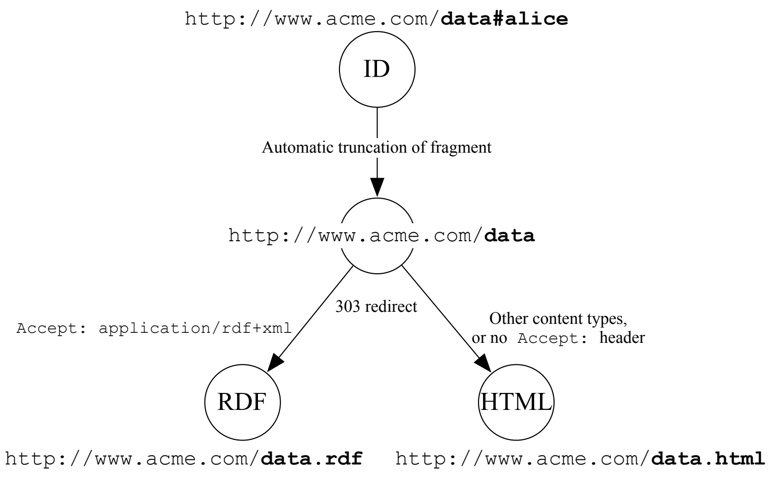
http://www.acme.com/**data#alice**
ID
Automatic truncation of fragment
http://www.acme.com/**data**
303 redirect
Accept: application/rdf+xml
Other content types, or no Accept: header
RDF
HTML
http://www.acme.com/**data.rdf**
http://www.acme.com/**data.html**

Abbildung 4: The hash URI solution with content negotiation

counter-effect, too. A client interested only in `#product123` will inadvertently load the data for all other resources as well, because they are in the same file. 303 URIs, on the other hand, are very flexible because the redirection target can be configured separately for each resource. There could be one describing document for each resource, or one large document for all of them, or any combination in between. It is also possible to change the policy later on. But the large number of redirects may cause higher latency.

Conclusion. Hash URIs should be preferred for rather small and stable sets of resources that evolve together. An ideal case are RDF Schema vocabularies and OWL ontologies, where the terms are often used together, and the number of terms is unlikely to grow much in the future.
Hash URIs without content negotiation can be implemented by simply uploading static RDF files to a web server, without any special server configuration. This makes them popular for quick-and-dirty RDF publication.
303 URIs should be used for large sets of data that are, or may grow, beyond the point where it is practical to serve all related resources in a single document.
If in doubt, it's better to use the more flexible 303 URI approach.

4.4 Cool URIs

The best resource identifiers don't just provide descriptions for people and machines, but are also "cool", a term Tim Berners-Lee uses for URIs designed with simplicity, stability and manageability in mind [4]. More guidelines are in Sections 1 and 3 of [15]:

Simplicity. Short, mnemonic URIs will not break as easily when sent in emails and are in general easier to remember, e. g. when debugging your semantic web server.

Stability. Once you set up a URI to identify a certain resource, it should remain this way as long as possible. Think about the next ten years. Maybe twenty. Keep implementation-specific bits and pieces such as `.php` and `.asp` out of your URIs, you may want to change technologies later.

Manageability. Issue your URIs in a way that you can manage. One good practice is to include the current year in the URI path, so that you can change the URI-schema each year without breaking older URIs. Keeping all 303 URIs on a dedicated subdomain, e.g. `http://id.acme.com/alice`, eases later migration of the URI-handling server.

4.5 Linking

All the URIs related to a single real-world object – resource identifier, RDF document URL, HTML document URL – should be explicitly linked with each other to help information consumers understand their relation. For example, in the 303 URI solution for ACME, there are three URIs related to Alice:

`http://www.acme.com/id/alice` - Identifier for Alice, the person

`http://www.acme.com/people/alice` - Alice's homepage

`http://www.acme.com/data/alice` - RDF document with description of Alice

Two of them are web document URLs. The RDF document located at `http://www.acme.com/data/` might contain these statements (expressed in N3):

```
<http://www.acme.com/id/alice>
    foaf:page <http://www.acme.com/people/alice>;
    rdfs:isDefinedBy <http://www.acme.com/data/alice>;

    a foaf:Person;
    foaf:name "Alice";
    foaf:mbox <mailto:alice@acme.com>;
    ...
```

The document makes statements about Alice, the person, using the resource identifier. The first two properties relate the resource identifier to the two document URLs. The `foaf:page` statement links it to the HTML document. This allows RDF-aware clients to find a human-readable version of the resource, and at the same time, by linking the page to its topic, defines useful metadata about that HTML document. The `rdfs:isDefinedBy` statement links the person to the document containing its RDF description and allows RDF browsers to distinguish this main resource from other auxiliary resources that just happen to be mentioned in the document. We use `rdfs:isDefinedBy` instead of its weaker superproperty `rdfs:seeAlso` because the content at `/data/alice` is authoritative. The remaining statements are the actual white pages data.
The HTML document at `http://www.acme.com/people/alice` should contain in its header a `<link />` element that points to the corresponding RDF document:

```
<html lang="en">
  <head>
    <title>Alice's Homepage</title>
    <link rel="alternate" type="application/rdf+xml"
          title="RDF Version"
          href="http://www.acme.com/data/alice" />
  </head> ...
```

This allows RDF-aware web clients to discover the RDF information. The approach is recommended in Section 9 of the RDF/XML specification [2]. If the information on the web page differs significantly from the RDF version, then we recommend using `rel="meta"` instead of `rel="alternate"`. The three desired links for each resource are shown in Figure 5.

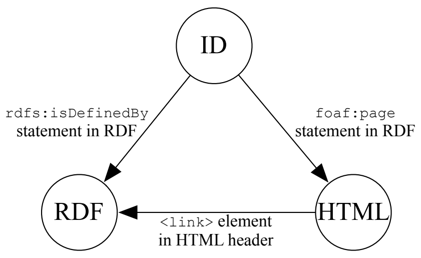

Abbildung 5: The RDF and HTML documents should relate the URIs to each other

4.6 Implementation

The W3C's Semantic Web Best Practices and Deployment Working Group has published a document [14] that describes how to implement the solutions presented here on the Apache web server. The document mostly discusses the publication of *RDFS vocabularies*, but its ideas can also be applied to other kinds of small RDF datasets that are published from static files.

5 Examples from the Web

Not all projects that work with semantic web technologies make their data available on the web. But a growing number of projects follow the practices described here. This section gives a few examples.

ECS Southampton. The School of Electronics and Computer Science at University of Southampton has a semantic website that employs the 303 solution[1] and is a great example of a carefully designed and well-documented[2] semantic web engineering. Separate subdomains are used for HTML documents, RDF documents, and resource identifiers. Take these examples:

`http://id.ecs.soton.ac.uk/person/1650` – URI for Wendy Hall, the person

`http://www.ecs.soton.ac.uk/people/wh` – HTML page about Wendy Hall

`http://rdf.ecs.soton.ac.uk/person/1650` – RDF about Wendy Hall

Entering the first URI into a normal web-browser redirects to an HTML page about Wendy Hall. It presents a web view of all available data on her. The page also links to her URI and to her RDF document.

D2R Server is an open-source application that can be used to publish data from relational databases on the semantic web in accordance with these guidelines [7]. It employs the 303 solution and content negotiation. For example, the *D2R Server publishing the DBLP Bibliography Database*[3] publishes several 100k

[1]`http://www.ecs.soton.ac.uk/`
[2]`http://id.ecs.soton.ac.uk/docs/`
[3]`http://www4.wiwiss.fu-berlin.de/dblp/`

bibliographical records and information about their authors. Example URIs, again connected via 303 redirects:

`http://www4.wiwiss.fu-berlin.de/dblp/resource/person/315759` – URI for Chris Bizer, the person

`http://www4.wiwiss.fu-berlin.de/dblp/page/person/315759` – HTML page about Chris Bizer

The RDF document for Chris Bizer is a SPARQL query result from the server's SPARQL endpoint:

```
http://www4.wiwiss.fu-berlin.de/dblp/sparql?query=
DESCRIBE+\%3Chttp\%3A\%2F\%2Fwww4.wiwiss.fu-berlin.de
\%2Fdblp\%2Fresource\%2Fperson\%2F315759\%3E
```

The SPARQL query encoded in this URI is:

```
DESCRIBE <http://www4.wiwiss.fu-berlin.de/dblp/resource/person/315759>
```

This shows how a SPARQL endpoint can be used as a convenient method of serving resource descriptions.

Semantic MediaWiki is an open-source semantic wiki engine. Authors can use special wiki syntax to put semantic attributes and relationships into wiki articles. For each article, the software generates a 303 URI that identifies the article's topic, and serves RDF descriptions generated from the attributes and relationships [17]. Semantic MediaWiki drives the OntoWorld wiki[4]. It has an article about the city of Karlsruhe:

`http://ontoworld.org/wiki/Karlsruhe` – the article, an HTML document

`http://ontoworld.org/wiki/_Karlsruhe` – the city of Karlsruhe

`http://ontoworld.org/index.php/Special:ExportRDF/Karlsruhe?xmlmime=rdf` – RDF description of Karlsruhe

There is an effort underway that calls for the adoption of Semantic MediaWiki as the software that runs Wikipedia [17]. This would turn Wikipedia into a repository of identifiers with community-agreed descriptions.

6 Other Resource Naming Proposals

Many other approaches have been suggested over the years. While most of them are appropriate in special circumstances, we feel that they do not fit the criteria from Section 3, which are to *be on the web* and *don't be ambiguous*. Therefore they are not adequate as general solutions for building a standards-based, non-fragmented, decentralized semantic web. We will discuss two of these approaches in some detail.

[4] `http://ontoworld.org/`

6.1 New URI schemes

HTTP URIs were originally designed to identify web documents, not other kinds of resources. Shouldn't we create a new URI scheme to identify resources? Then we could easily distinguish resources from web documents just by looking at the first characters of the URI. For example, the *info* scheme can be used to identify books based on a LCCN number: `info:lccn/2002022641`.
Here are examples of such new URI schemes. For a more complete list, see [16].

- **Magnet**[5] is an open URI scheme enabling seamless integration between websites and locally-running utilities, such as file-management tools. It is based on hash-values, a URI looks like this:
  `magnet:?xt=urn:sha1:YNCKHTQCWBTRNJIV4WNAE52SJUQCZO5C`.
- The **info:** URI scheme[6] is proposed to identify information assets that have identifiers in existing public namespaces. Examples are URIs for LCCN numbers (`info:lccn/2002022641`) and the Dewey decimal system (`info:ddc/22/eng//004.6`
- The idea of **Tag URIs**[7] is to generate collision-free URIs by using a domain name and the date when the URI was allocated. Even if the domain changes ownership at a later date, the URI remains unambiguous. Example: `tag:hawke.org,2001-06-05:Taiko`.
- **XRI**[8] defines a scheme and resolution protocol for abstract identifiers . The idea is to use URIs that contain wildcards, to adapt to changes of organizations, servers, etc.
  Examples are `@Jones.and.Company/(+phone.number)` or
  `xri://northgate.library.example.com/(urn:isbn:0-395-36341-1)`.

To be truly useful, a new scheme must also define a protocol how to access more information about the identified resource. For example, the `ftp://` URI scheme identifies resourcse (files on an FTP server), and also comes with a protocol for accessing them (the FTP protocol).
Some of the new URI schemes provide no such protocol at all. Others provide a web service that allows retrieval of descriptions using the HTTP protocol. The identifier is passed to the service, which looks up the information in a central database or in a federated way. The problem here is that a failure in this service renders the system unusable.
Another drawback can be a dependence on a standardization body. To register new parts in the `info:` space, a standardization body has to be contacted. This, or paying a license fee before creating a new URI, slows down adoption. In cases a standardization body is desirable to ensure that all URIs are unique (e.g. with ISBNs). But this can be achieved using HTTP URIs inside an HTTP namespace owned and managed by the standardization organization.
The problems with new URI schemes are discussed at length by Thompson and Orchard [16].

[5] `http://magnet-uri.sourceforge.net/`
[6] `http://www.ietf.org/rfc/rfc4452.txt`
[7] `http://www.taguri.org/`
[8] `http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xri`

6.2 Reference by Description

This approach radically solves the URI problem by doing away with URIs altogether: Instead of *naming* resources with a URI, *anonymous nodes* are used, and are *described* with information that allows us to find the right one. A person, for example, could be described with her name, date of birth, and social security number. These pieces of information should be sufficient to uniquely identify a person.

A popular practice is the use of a person's email address as a uniquely identifying piece of information. The `foaf:mbox` property is used in FOAF profiles[9] for this purpose. In OWL, this kind of property is known as an *Inverse Functional Property* (IFP). When an agent encounters two resources with the same email address, it can infer that both refer to the same person and can treat them as one.

But how to *be on the web* with this approach? How to enable agents to download more data about resources we mention? There is a best practice to achieve this goal: Provide not only the IFP of the resource (e.g. the person's email address), but also an `rdfs:seeAlso` property that points to a web address of an RDF document with further information about it. We see that HTTP URIs are still used to identify the location where to download more information.

Furthermore, we now need several pieces of information to refer to a resource, the IFP value and the RDF document location. The simple act of linking by using a URI has become a process involving several moving parts, and this increases the risk of broken links and makes implementation more cumbersome.

Regarding FOAF's practice of avoiding URIs for people, we agree with Tim Berners-Lee: "Go ahead and give yourself a URI. You deserve it!" [10]

7 Conclusion

Resource names on the semantic web should fulfill two requirements: First, a description of the identified resource should be retrievable with standard web technologies. Second, a naming scheme should not confuse documents and the things described by the documents.

We have described two approaches that fulfill these requirements, both based on the HTTP URI scheme and protocol. One is to use the 303 HTTP status code to redirect from the resource identifier to the describing document. One is to use "hash URIs" to identify resources, exploiting the fact that hash URIs are retrieved by dropping the part after the hash and retrieving the other part.

The requirement to distinguish between resources and their descriptions increases the need for coordination between multiple URIs. Some useful techniques are: embedding links to RDF data in HTML documents, using RDF statements to describe the relationship between the URIs, and using content negotiation to redirect to an appropriate description of a resource.

[9] `http://www.foaf-project.org/`

[10] `http://dig.csail.mit.edu/breadcrumbs/node/71`

8 Acknowledgements

Many thanks to Tim Berners Lee who helped us understanding the TAG solution by answering chat requests[11].
This work was supported by the German Federal Ministry of Education, Science, Research and Technology (bmb+f), (Grants 01 IW C01, Project EPOS: Evolving Personal to Organizational Memories; and 01 AK 702B, Project InterVal: Internet and Value Chains) and by the European Union IST fund (Grant FP6-027705, Project Nepomuk).

9 Copyright Notice

Literatur

[1] Anonymous. Hypertext transfer protocol. Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/HTTP`.

[2] Dave Beckett. Rdf/xml syntax specification (revised). W3c recommendation, W3C, Oct 2004. `http://www.w3.org/TR/rdf-syntax-grammar/`.

[3] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform resource identifier (uri): Generic syntax. RFC 3986, January 2005. `http://www.ietf.org/rfc/rfc2396.txt`.

[4] Tim Berners-Lee. Cool uris don't change. Design Issues, 1998. `http://www.w3.org/Provider/Style/URI`.

[5] Tim Berners-Lee. Notation 3. Design Issues, 1998-2006. `http://www.w3.org/DesignIssues/Notation3`.

[6] Tim Berners-Lee. What http uris identify. Design Issues, 06 2005. `http://www.w3.org/DesignIssues/HTTP-URI2.html`.

[7] Christian Bizer and Richard Cyganiak. D2r server – publishing releational databases on the semantic web. Poster at the 5th International Semantic Web Conference, Athens, USA, Nov 2006. `http://sites.wiwiss.fu-berlin.de/suhl/bizer/pub/Bizer-Cyganiak-D2R-Server-ISWC2006.pdf`.

[11] `http://chatlogs.planetrdf.com/swig/2006-10-29#T17-42-28`

[8] David Booth. Four uses of a url: Name, concept, web location and document instance. Website, Jan 2003. `http://www.w3.org/2002/11/dbooth-names/dbooth-names_clean.htm`.

[9] Jeremy J. Carroll and Patrick Stickler. Rdf triples in xml. In *Extreme Markup Languages 2004 Proceedings*, 2004. `http://www.mulberrytech.com/Extreme/Proceedings/html/2004/Stickler01/EML2004Stickler01.html`.

[10] R. Fielding et al. Hypertext transfer protocol – http/1.1. RFC 2616, Jun 1999. `http://www.ietf.org/rfc/rfc2616.txt`.

[11] E. Miller F. Manola. Rdf primer. W3c recommendation, 2004, W3C, 10 February 2004. `http://www.w3.org/TR/rdf-primer/`.

[12] Roy T. Fielding. [httprange-14] resolved. Mailing list message to www-tag@w3.org, Jun 2005. `http://lists.w3.org/Archives/Public/www-tag/2005Jun/0039.html`.

[13] Ian Jacobs and Norman Walsh. Architecture of the world wide web, volume one. W3c recommendation, W3C, Dec 2004. `http://www.w3.org/TR/webarch/`.

[14] Alistair Miles, Thomas Baker, and Ralph Swick. Best practice recipes for publishing rdf vocabularies. W3c working draft, W3C, Mar 2006. `http://www.w3.org/TR/swbp-vocab-pub/`.

[15] Olivier Théreaux. Common http implementation problems. W3c note, W3C, Jan 2003. `http://www.w3.org/TR/chips/`.

[16] Henry S. Thompson and David Orchard. Urns, namespaces and registries. Draft tag finding, W3C, 2006. `http://www.w3.org/2001/tag/doc/URNsAndRegistries-50`.

[17] Max Völkel, Markus Krötzsch, Denny Vrandecic, Heiko Haller, and Rudi Studer. Semantic wikipedia. In *Proceedings of the 15th international conference on World Wide Web, WWW 2006*, May 2006. `http://www.aifb.uni-karlsruhe.de/WBS/hha/papers/SemanticWikipedia.pdf`.

-07-01

cal Memo