Representing internal and external links in the VIVO ontology

This page was stimulated by a question from Griffin Weber:

I have a question about the vivo:linkURI property of a vivo:URLLink class. The vivo:linkURI property is a datatype property. However, a related question is how do you store the value of the vivo:linkURI property in a VIVO instance if that value is a URI in that same instance of VIVO? Is it a literal or an entity?

Some context first

xsd:anyURI datatype

(from http://www.schemacentral.com/sc/xsd/t-xsd_anyURI.html):

The type `xsd:anyURI` represents a Uniform Resource Identifier (URI) reference. URIs are used to identify resources, and they may be absolute or relative. Absolute URIs provide the entire context for locating the resources, such as `http://datypic.com/prod.html`. Relative URIs are specified as the difference from a base URI, such as `../prod.html`. It is also possible to specify a fragment identifier, using the # character, such as `../prod.html#shirt`.

The three previous examples happen to be HTTP URLs (Uniform Resource Locators), but URIs also encompass URLs of other schemes (e.g., FTP, gopher, telnet), as well as URNs (Uniform Resource Names). URIs are not required to be dereferencable; that is, it is not necessary for there to be a web page at `http://datypic.com/prod.html` in order for this to be a valid URI.

foaf:page property

From the 2010 FOAF specification, "The page property relates a thing to a document about that thing. As such it is an inverse of the `topic` property, which relates a document to a thing that the document is about. A Document class represents those things which are, broadly conceived, 'documents'. The relationship between documents and their byte-stream representation needs clarification – The `Document`class is currently used in a way that allows multiple instances at different URIs to have the 'same' contents (and hence hash)."

bibo:Webpage class

A bibo:Webpage is a subclass of bibo:Document, which is declared equivalent to a foaf:Document. The VIVO ontology includes the bibo:Webpage class to be able to represent authorship of a web page by a person or organization. A web page is part of a bibo:Website, a subclass of bibo:Collection.

comment: "A web page is an online document available (at least initially) on the world wide web. A web page is written first and foremost to appear on the web, as distinct from other online resources such as books, manuscripts or audio documents which use the web primarily as a distribution mechanism alongside other more traditional methods such as print."

usage: bibo:Website dcterms:hasPart only bibo:Webpage

vivo:webpage and vivo:webpageOf object properties

The *vivo:webpage* property relates to an intermediate node (of type vivo:URLLink) that allows specifying the homepage URL as a literal (via the *vivo:linkURI* data property), as well as a human-readable label ("anchor text") for the link (*vivo:linkAnchorText*).

In the ISF the vivo:webpage property and vivo:webpageOf properties are deprecated in favor of isf:has_url_link and isf:url_link_for object properties.

vivo:URLLink class

The definition of a vivo:URLLink is listed as, "The Uniform Resource Locator (URL) specifies where an identifier resource is available and the mechanism for retrieving it." An example is given as http://info.slis.indiana.edu/~katy/. A vivo:URLLink is a subclass of an IAO 'information content entity', which is replacing vivo:InformationResource as we transition to the Integrated Semantic Framework, or ISF.

A vivo:URLLink has a data property *vivo:linkURI (*sub property of *vivo:hasValue)* for storing the URL, which expects a datatype *xsd:anyURI* as its value. The VIVO 1.5 ontology *vivo:linkAnchorText* data property appears to be deprecated in favor of rdfs:label with the transition to the ISF.

Getting back to the questions

The value of link:URI is a URI, right? So, why isn't it an object property?

The vivo:URLLink class with its vivo:linkURI value of an xsd:anyURI was set up to be able to be able to capture the URL as a string literal in the VIVO editor and to associate a label with a URI so that two different entities in VIVO could reference the same URI and use different labels. Both of these reasons have more to do with the VIVO application than the ontology.

At root the question should be why we need the vivo:URLLink class at all – can't we just enter the URI of the remote resource as the object of the has_url_link property? If a label is found for the URI in the VIVO triple store, the URI could be rendered on the page with anchor text, or otherwise as an unadorned URL.

We support the URLLink in the application not only to be able to capture the anchor text but a rank value that is used to order a person or organization's web pages. URLLink presently has one subclass – a Faculty of 1000 link -- and there have been requests to be able to differentiate full text links as well.

We also have a need for the application to treat the URI of a resource in VIVO differently from a resource known to be native to another application (so far another VIVO, but it could be Profiles or anything else). The VIVO application can be configured to treat certain namespaces as external namespaces to redirect to rather than render locally. This configuration was developed to allow storing the URIs, rdf:type statements, and rdfs:labels of Weill Cornell Medical College people and organizations in the Cornell Ithaca VIVO instance while ensuring that these URIs were redirected to the Weill Cornell VIVO for rendering, since the Weill Cornell VIVO has much more complete information about that URI.

When rendering the object of an *isf:has_url_link* property, however, we change the application to link directly to the URI of the URLLink itself rather than looking for a *vivo:linkURI* data property of the URLLink entity. This would require some data migration in the transition to VIVO 1.6 and would require making the application robust enough to display an rdfs:label if present but just render the URI otherwise; an rdf:type statement would also have to be made optional, and indeed we don't know anything about the type of the URI so should not infer that it is a document, person, organization, or anything else.

By doing this we would lose two features that are important for the VIVO application if the same URI is referenced from two different entities in VIVO and wants to be ordered or labeled differently when displayed on each entity's individual page. For example, if the same URI is the object of multiple *isf:has_url_link* properties there could be multiple rdfs:labels and multiple *vivo:rank* properties for the same URI in the triple store and the application would not have any way of knowing which to use.

This is a case where the needs of the application have been allowed to trump the most straightforward way of representation in the ontology. However, if we know what the remote URI is, it seems like we should type it as a Person or Document or Organization, however, and use a more semantically meaningful property relationship than the generic *isf:has_url_link* property?

How do you store the value of the vivo:linkURI property in a VIVO instance if that value is a URI in that same instance of VIVO?

The application doesn't attempt to parse the xsd:anyURI strings in a vivo:URLLink entity in any special way if you set a vivo:linkURI property that points to the same VIVO application. That link will break if resource URIs get changed, just like any other link.

If we adopted the strategy described just above of using the desired URI as the URI of the vivo:URLLink rather than storing the URI in the vivo:linkURI property, a VIVO URI would be updated if that entity's URI was modified through the VIVO application (although this can only be done by administrators).

This introduces the philosophically challenging notion of whether the URI refers to the person or the webpage of a person, however – normally a reference to a URLLink references a webpage and a different object property would be to link an organization to a person, for example.