Joachim: any reason not to include the abstract classes?

Dan: it’s a requirement. We need to be able to say we’ll link to an Agent.

Arofan: you end up with insane complexity compared to other OWL ontologies

ericP: i think Dan’s argument trumps that. Also that we’re an application which is necessarily more complex than ontologies.

PROPOSED: include abstract classes in the RDF model

APPROVED

Joachim: alter the PIM->PSM workflow?

… it was invented last year, before we had just a model binding.

… one reason is to map to existing vocabs in the PSM

Guillaume: we ran into these requirements in mapping GSM UML to OWL because we couldn’t give different ranges to properties

[discussion of OMG ODM ([domain]property][range]) and OWL vs. RDFS]

Joachim: the PIM->PSM framework allows one to generate different kinds of bindings for RDF

Dan: Joachim asked “should the mappings to other ontologies be done within the PSM?”. If we have a particular binding, folks could use the PSM to bind to others.

[discussion about standardizing single RDF vocab vs. creating multiple bindings]

@@how did foaf get blacklisted?

FOAF inverse function datatype properties (i.e. people identified by SSNs)

Arofan: can we stay ahead of the fickle RDF community?

ericP: it’s a question of re-using familiar ontologies. Better than totally new ontology.

PROPOSED: do not attempt to be OBO or SIO-compliant

APPROVED

Kelly’s Notes:

Oliver: Not creating any PSM to focus on the PIM bc every reduction would mean a loss of information.

Achim: yes/no other reasons to have PSM for RDF

O: Getting rid of the inheritance chain and abstracts. Nothing to change for programming languages

A; Should reflect what is realized in the binding syntax on model level.

O: we have already done that by definition of UML restrictions and XML style, because that is close to common things between programming languages and XML.

Arofan; Use case of RDF and XML roundtrip - different ways of identifying things (URNS and URIs), to inject the URN into the RDF representation, assign URIs, injecting stuff into one binding that isn’t in the original source.

O: That is an instance issue not a modeling issue. An extra field in the RDF that is not available in XML and vice versa. Typed user-added file predefining its RDF url and XML urn as the presets. User ID could always be used for that to pass over.

Dan: We have a document that speaks about that.

Arofan takes over notes.

## **Discussion with Oliver Hopt regarding existing PIM-> PSM Transform**

There was no compelling argument for avoiding abstract classes in the RDF binding, except that it simplified the vocabulary.

Do we need the PSM? Now – if we keep the abstract classes, the PIM-PSM transformation does nothing.

There are two arguments for keeping abstracts in the XSD – there are those who would want this (development based on the XSD, not the model) and those who would consult the model during development. The key thing is how these object relations are expressed in the programming language being used (Java, C#, etc.)

## **Using Other Vocabularies**

Vocabularies to consider:

- RDFS/OWL

- PROV-O – would need to restrict PROV-O classes using OWL

- SKOS/XKOS – For which purposes should these be used/reused. DDI controlled vocabularies are a good match. For classifications, codelists SKOS/XKOS should be useful. Things in our collection patterns may require some new extension – need to look at this.

- DCAT – mapping for collection of data sets – we could extend or use DCAT classes. Look at ESTATs application profile.

- DataCube

- CSVW – mapping needs exploration; a little bit redundant against OWL/RDFS – may not be useful if the redundant features are the only ones we would need

- PAV – potential overlaps (see Version note below)

- ORG

- DC-O – would need to restrict DC classes using OWL

DISCO is not here because it will be deprecated when we publish a DIO 4 RDF binding.

Decision: Use DC properties in OWL restriction classes as a way of allowing DDI and DC to be used together. This approach could be used also with some other vocabularies. You could also use sub-classes and sub-properties. The OWL restriction approach is typically a better approach.

(If an external field is too loose, you could use restriction classes in OWL to narrow its use.)

PAV:Version could be used – in a restricted form – to capture the version number (as an example). PAV is one we want to map at a field level.

CSVW – tables (always), but not always dimensional tables (hypercubes). What is the interplay between DDI and CSVW and DDI DataCube? Probably we could map DDI Data Description aggregate data sets against DataCube.

We want to restrict an external class – we will have to maintain a record of where we ae using/restricting the external class. Is this a configuration within the build framework.

Question from Jon:: Has there been a proposal made on absorption of external models / ontologies vs mapping - and is a risk analysis and costing associated with such a decision been done