Title of session: RDF bindings for statistical data and metadata

Day: Thursday

Participants: Achim Wackerow, Wendy Thomas, Eric Prud'hommeaux, Gary Berg-Cross, Guillaume Duffes, Daniella Meeker, Diedre Lungley, Nick Car, Gregg Kellogg

Chair: Achim Wackerow

Note taker: Wendy Thomas

Background information

The discussion covered the current approach used by DDI including criteria and current production model. The RDF/OWL Binding for the DDI Model by Richard Cyganiak was reviewed in order to identify any changes needed. This was followed by a review of the FHIR model.

Approach in organizing the Class Library within the DDI model:

- Group by things that are used together based on example data

Basic production from model to binding:

- PIM (XMI) to PSM (XMI) using a configuration file
- We have identified the relevant vocabularies as OWL/RDF-S and other important vocabularies: DC, DCAT, ORG, PROV, FOAF, SKOS/XKOS, CSVW, PAV, DataCube
- We need to clarify the vocabularies that are introduced in the configuration file

General Discussion

Criteria for selection of RDF vocabularies:

- Being used – the vocabulary is in active use
- Strong Potential – the vocabulary appears to have a strong potential for use within its coverage area
- Endorsed by W3C – might be rare
- Mapping exists between similar vocabularies (i.e. DC and PROV, DC and FOAF)
- Use what is useful to the people doing it
- Only a couple of properties of Dublin Core get used but are very "book" and "publication" centric
- SIO (Scientific Integration Ontology) is another we should look at (see Michel from last year)

Mapping is done at the model level

- The approach needs to be property by property and relationship by relationship
- Iterative process of finding what is a clear relationship
    - May have to edit the model to get a clean match
    - Remaining items:

        - Check on-line for a vocabulary
        - Create a primary DDI Vocabulary
            - Should be an extension/qualification of an existing vocabulary if possible
- Order for preference: RDF, RDF-S, OWL, PROV,
    - If you put in PROV Person, it will infer that it is FOAF Person; but many sparkle queries may not do that
- As you go along you want to extend existing broader or looser concepts and relate any DDI vocabulary terms
- If it is equivalent, use the OWL as it is the most common
- Most are used without any inference at all
- Recommendation: Model the selected RDF classes in the UML model in a separate package of the model which could then be related to the classes
- Use in Model:
    - Make sure they are using inference
    - Leverage the same fragments and determine if we are saying something sensible
    - If we aren't doing inference which vocabulary term is going to best serve the populace
- How much of an open world do you expect to use DDI within; where you need to intersect with specific vocabulary terms?
- Mapping:
    - If there is a one to one relationship the configuration file needs
    - Pushing it from the configuration file to the UML
        - Reduces duplication between XMI and RDF configuration file
    - By putting out overlap lists (PROV and FOAF for example)
        - Put out list of classes used in each vocabulary including overlap
        - Equivalencies are difficult to maintain and can have unintended consequences
        - Rely on external mappings rather than express internally
        - Everywhere you are using FOAF Agent you can use DC Agent
            - If you have a new thing you have to say that it is a sub-class of both a FOAF Agent and DC Agent
- Map out an example to PROV which, unlike many other vocabularies, assumes or depends on inference
- Most people will do inference free sparkle queries
    - You'd have to go through extra steps to resolve into PROV
- Configuration:
    - UML (PIM) transform to PSM (OWL/RDF) transform to syntax
    - Configuration files exist for each transformation
    - If the PSM is the equivalent to the PIM do you need the first configuration?
        - Annotation could come out at a separate stream
- DDI use of these vocabularies is for the purpose expressed by the DDI namespace, not to be a PROV centric
- Is there a reason for having a PSM?
    - Consistency in production

    - If it’s easy its cheap, but if it’s easy is it needed?
    - Should the rules be imbedded in the program or are the rules in the configuration file?
    - Consistency in rules update
- What types of changes are you trying to future-proof against?
    - What structures in the PIM mean?
    - Modeling change in terms of structure (not content)
- Round-trip issues for lossless transfer
    - In XML land you are passing a document boundary (instance to instance, document to document)
    - In general, a relationship goes from on object to another
        - Information resources are a set of objects(?)
    - In XML there are references (external/internal). While there is nothing stopping you imbedding a new document or including it by reference. This has a consequence in XML. In RDF you have chains of triples
    - All the questions that you might want to ask can only be asked in our triple store. In other instances, you want to go into an external store. Certain engines can do this but often it is documentation

Review of the RDF document by Richard Cyganiak

Do the RDF names spaces need to be aligned with the XML

- If you have to resolve the namespace
- You can use the same one because
    - Identify a namespace and resolution via namespace

Must it be resolvable?

- Should be resolvable with content negotiation

Classes – start with upper case

Abstract classes –

- We have abstract classes in UML but abstract is not supported by RDF
- Object oriented is used in a strict sense
- Abstract usage in the bindings
    - Abstact nature gets lost, it never gets used as an abstraction but as a means of identifying the allowed subtypes (similar to how we have done this in the XML bindings)
    - You can declare it in the OWL ontology as say a label (non-executable) can write a SHACL or ShEx rule.
        - Extended validation of documents
            - SHACL business based
            - ShEx academic based
- Use of cardinality
    - Semantic way – no cardinality constrains (theoretical principal)

    - Real world – should you do in OWL? Yes, but it is not enforced
    - Put them in at least as a theoretical expression of what you want even if they cannot be currently enforced
        - SHACL and ShEx should eventually provide some enforcement
- XSD data types
    - Yes, these can be used directly in RDF
    - Supports full sub-types
- Mapping to other vocabularies
    - Shouldn't use 'same as' because you are making a strong claim (including restrictions)
    - Equivalent is also very strong
    - "subclassOf" or "subpropertyOf"
    - Don't do secondary or tertiary languages unless
    - Should avoid others like "similar"
- There a requirement of ordering within collections
    - OrderList ontology – miniature ontology – just a sequence
    - Will need to make our own as there doesn't seem to be a language to do that
    - DISCO used a SKOS construct – but we should use a native based ontology order
- Jena's rdfcat to merger generated RDF/XML with the vocabulary mappings and write result to Turtle
    - Merge to a single store (Jena is an example of a tool to do this)

FHIR perspective

Use of references (relationships)

- Embedding versus reference
- If these are separate documents, then reference can provide you with extra information
- Is it stupid to go through an extra arch to get to the information?
- When dealing with documents that reference other documents are they first class objects that have multiple properties with them (UUID's etc.)
    - DDI does have the requirement of additional information
    - DDI has objects that get referenced by multiple locations
- May stick them in a graph
- Use a name-graph store (have to put in both graph identification and object)
- There are RDF stores where the default draft is the unit of all the named graphs
- Named graphs help support grabbing everything in a graph but slows down querying across graphs
- Rest protocol used
- A data set could have a default graph and any number of named graphs – you can pack multiple documents in one document
- You need something like TRIG to create collections of files

Lossless round-tripping

- From Binding A to Binding B and back again
- Both JSON XML go from an ordered tree to a graph
    - Create a Tree root and node role (hierarchy)
- If you try to do this without going to the model, you would probably find you did go through the model (you need to know the schema in order to spit the graph content into the right tree order)
    - If sequencing is not an issue you can forget about a lot of things such as putting ordinals on things, consulting the schema for order (RDF to XML),
    - In RDF if you have an order (ordinal) you need to use the DDI ordering as the general RDF ordering does not translate (as the XML has no order)
    - Preference of No Order option if available in any binding
- Issue of multiple tree roots
    - If you go in from an object (i.e. Question) up to the Tree Root a single object might have multiple Tree Roots because it is included in multiple documents by external reference
    - Decide what the root element is…derive the root element
    - A “reference” in this case is one that is an arc reference rather than an XML version of a reference i.e. nesting
    - We may not need tree root because it is clear by arching
    - You can actually provide a noderole
        - Fhir:noderole value fhir:treeroot
- Materializing datatypes
    - Use RDF friendly datatypes
    - xsd datatypes
    - Sparql uses only datetime for comparison, not between various date – the behaviors are defined but not required in implementations
- Other risks involving round trips
    - Using XML, RDF and JSON
    - Couple of program libraries
    - We will need a notion of equivalence
        - An expensive tool that does canonicalization (those that do the same signature)
            - A whitespace sensitive way of doing this
            - www.w3.org/TR/xml-c14n   section 3.x
        - Can’t really do this with Turtle – what do you do with a blank node, giving it an id
            - End of line blanks, different end of line, etc. need to be addressed
        - Use canonicalization as a black box
    - FHIR did a lot of reverse engineering to get this legible for users
        - hl7-fhir.github.io/ovservation-example.jason.html
- Extensibility
    - Mapping is a problem as RDF allows writing extra triples; detecting these triples is hard because is it something we know about and the XML works differently
    - FHIR is not good

Relevant Documents

Linked Open Vocabularies **http://lov.okfn.org/dataset/lov**

RDF Schema/OWL Binding for the DDI Model - Richard Cyganiak

Recommendations/Concensus areas:

Criteria for selection of RDF Vocabularies:

- The vocabulary should be in active use and/or have strong potential for use within the coverage area
- Endorsed by W3C - nice to have
- Existing mapping between the vocabulary and similar vocabularies

Vocabulary Selection:

- Order of preference: RDF, RDF-S, OWL, PROV
- Note that FOAF is disallowed by the U.S. Federal Government
- If vocabulary object is equivalent use OWL
- DDI vocabulary items should extend existing vocabulary items whenever possible

The review of the Cyganiak paper was positive with some specific recommendations:

Abstract usage in bindings:

- You can declare it in the OWL ontology as say a label (non-executable) can write a SHACL or ShEx rule.
    - Extended validation of documents
        - SHACL business based
        - ShEx academic based

Use of cardinality:

- Put them in at least as a theoretical expression of what you want even if they cannot be currently enforced
    - SHACL and ShEx should eventually provide some enforcement

XSD data types:

- Not a problem, support for full sub-types

Mapping:

- Shouldn't use 'same as' because you are making a strong claim (including restrictions)
- Equivalent is also very strong
- "subclassOf" or "subpropertyOf"

- Don’t do secondary or tertiary languages unless
- Should avoid others like “similar”

Actions needed:

- We need to clarify the vocabularies that are introduced in the configuration file
- Review important vocabularies in light of the noted criteria - Broad list includes OWL/RDF-S, DC, DCAT, ORG, PROV, FOAF, SKOS/XKOS, CSVW, PAV, DataCube, SIO
- Map at the level of property-by-property and relationship-by-relationship
- Follow an iterative process of finding a clear relationship:
- May have to edit the model to get a clean match
- Remaining items:
- Check on-line for a vocabulary
- Create a primary DDI Vocabulary
- Should be an extension/qualification of an existing vocabulary if possible

<u>Issues requiring further discussion</u>

The following areas should be reviewed for further discussion in support of arriving at a final approach to an RDF binding.

Editing model to attain clean match with RDF

We have a rule about avoiding modeling to a specific binding and have continued to review model to remove XML centric modeling. Is there a conflict here in modeling to match RDF?

- Recommendation: Model the selected RDF classes in the UML model in a separate package of the model which could then be related to the classes
- Use in Model:
  - Make sure they are using inference
  - Leverage the same fragments and determine if we are saying something sensible
  - If we aren’t doing inference which vocabulary term is going to best serve the populace
- How much of an open world do you expect to use DDI within; where you need to intersect with specific vocabulary terms?
- Mapping:
  - If there is a one to one relationship the configuration file needs
  - Pushing it from the configuration file to the UML
    - Reduces duplication between XMI and RDF configuration file
  - By putting out overlap lists (PROV and FOAF for example)
    - Put out list of classes used in each vocabulary including overlap
    - Equivalencies are difficult to maintain and can have unintended consequences
    - Rely on external mappings rather than express internally
    - Everywhere you are using FOAF Agent you can use DC Agent

            - If you have a new thing you have to say that it is a sub-class of both a FOAF Agent and DC Agent
- Configuration:
    - UML (PIM) transform to PSM (OWL/RDF) transform to syntax
    - Configuration files exist for each transformation
    - If the PSM is the equivalent to the PIM do you need the first configuration?
        - Annotation could come out at a separate stream
- Is there a reason for having a PSM?
    - Consistency in production
    - If it's easy it's cheap, but if it's easy is it needed?
    - Should the rules be imbedded in the program or are the rules in the configuration file?
    - Consistency in rules update

Round-tripping between bindings:

The current agreement in the Modeling Team was that round-tripping went through the model rather than direct binding-to-binding. This may require additional discussion.

- Round-trip issues for lossless transfer
    - In XML land you are passing a document boundary (instance to instance, document to document)
    - In general, a relationship goes from on object to another
        - Information resources are a set of objects (?)
    - In XML there are references (external/internal). While there is nothing stopping you imbedding a new document or including it by reference. This has a consequence in XML. In RDF you have chains of triples
    - All the questions that you might want to ask can only be asked in our triple store. In other instances, you want to go into an external store. Certain engines can do this but often it is documentation
- If you try to do this without going to the model, you would probably find you did go through the model (you need to know the schema in order to spit the graph content into the right tree order)
    - If sequencing is not an issue you can forget about a lot of things such as putting ordinals on things, consulting the schema for order (RDF to XML),
    - In RDF if you have an order (ordinal) you need to use the DDI ordering as the general RDF ordering does not translate (as the XML has no order)
    - Preference of No Order option if available in any binding
- Issue of multiple tree roots
    - If you go in from an object (i.e. Question) up to the Tree Root a single object might have multiple Tree Roots because it is included in multiple documents by external reference
    - Decide what the root element is…derive the root element
    - A "reference" in this case is one that is an arc reference rather than an XML version of a reference i.e. nesting

    - We may not need tree root because it is clear by arching
    - You can actually provide a noderole
        - Fhir:noderole value fhir:treeroot

Use of relationships (references)

Current model and approach

- DDI has objects that get referenced by multiple locations
- All of these objects are class A and have identification

Explore use of name-graph

- Use a name-graph store (have to put in both graph identification and object)
- There are RDF stores where the default draft is the unit of all the named graphs
- Named graphs help support grabbing everything in a graph but slows down querying across graphs
- Rest protocol used
- A data set could have a default graph and any number of named graphs – you can pack multiple documents in one document
- You need something like TRIG to create collections of files

Is there a need for a continuation of this discussion?

Yes, among the Modeling Team with expert internal and external input as needed including review of final approach.

Is there a need for a longer document to continue this discussion?

No, relevant document would be decisions coming out of a review of this by the Modeling Team resulting in guidelines and documentation for creating and using an RDF binding.