Terminology binding in FHIR-RDF

Tony Mallia 1/15/2015
Edmond Scientific Company

1 Introduction

There are a couple of requirements which may be easy to meet but have some difficult areas. These two requirements are

1. To have achievable round trip transformation of FHIR payloads to RDF
2. To be RDF tool friendly in the constructed RDF

For much of the transformation a verbatim (or transliteral) translation will work but there are difficult areas of which at least four have been identified: Terminology, Extension, Valueset and Resource Reference.

This paper is an attempt to find a middle ground on terminology which can meet both these requirements so that we do not have to develop two versions of RDF.

2 Terminology differences

In FHIR the binding to terminology uses the FHIR Coding Type which has various values as uri, strings and code. There is no concept of including the terminology model.

RDF has representations of some terminologies – SNOMED CT and ICD-11. Both these representations use OWL Classes to represent the concepts in a hierarchy. A term is represented as a URI. In SNOMED CT there are also a number of Object Properties which can be used to express post coordinated codes in pre-coordinated form expressions.

The bindings to a term of an element in FHIR and RDF are therefore fundamentally different.

FHIR provides the code/system for a CTS to lookup the term. FHIR carries the display value in the instance payload to avoid having to lookup the term.

RDF has a relationship between the element (rdfs:Resource – also called an owl:Named Individual) and the term class. The only relationship available is rdf:type and therefore the element must declare itself or a related individual as member of the Term OWL class.

3 Binding of instance

Example of House Dust Allergen as a FHIR substance resource (from the FHIR DSTU examples).

```
<?xml version="1.0" encoding="UTF-8"?>
<Substance xmlns="http://hl7.org/fhir">
  <name value="House Dust"/>
  <type>
    <coding>
      <system value="http://snomed.info/id"/>
      <code value="406466009"/>
      <display value="House dust allergen"/>
    </coding>
  </type>
</Substance>
```

In an RDF instance binding to a SNOMED concept (defined by IHTSDO) would look like:
rec:xxxx rdf:type <http://snomed.info/id/406466009>

rdf:type is concatenation of “system” and “code” from the FHIR payload

The rec: is the instance namespace identification.
The RDF can now obtain closure to the concept in the SNOMED ontology if imported.

What is the instance? There have been various approaches to this which range from making the substance the instance with the term type applied to it to a verbatim approach to declare coding and its components as complex types but eventually as primitives. There was resistance to making the substance a type due to the difficulty of round tripping back to the FHIR structure. There was resistance to the verbatim approach since OWL does not match on strings only on URIs so it is not possible to get closure to imported terminologies.

There is a middle ground to make "coding" an instance where the Substance instance has an object property "Substance.type" to the Coding instance. Coding is mapped verbatim with specific additions marked with * to make an rdfs:type "fhir:Coding".

There are two ways to arrive at the fully populated RDF:

1. Assemble the entities from the FHIR Resource and then compute the * additions
2. Assemble the entities from a minimal RDF (marked with *) and compute the FHIR structure.

Here are RDF target substance and coding instances represented in Turtle (from Protégé). (rdf:type owl:NamedIndividual are omitted)

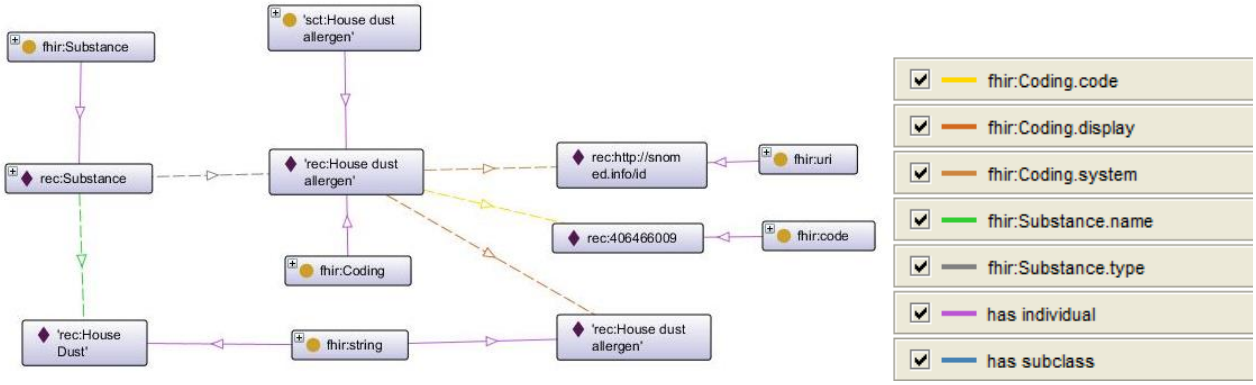
```
<http://record#01338> rdf:type fhir:Substance ; *
                      rdfs:label "Substance"@en ; *
                      fhir:tag "Substance"^^xsd:Name ;
                      fhir:Substance.type <http://record#01339> ;
                      fhir:Substance.name <http://record#01343> .

<http://record#01339> rdf:type fhir:Coding ,*
                               <http://snomed.info/id/406466009> ; *
                      rdfs:label "House dust allergen" ; *
                      fhir:tag "coding"^^xsd:Name ;
                      fhir:Coding.system <http://record#01340> ;
                      fhir:Coding.code <http://record#01341> ;
                      fhir:Coding.display <http://record#01342> .

<http://record#01340> rdf:type fhir:uri ;
                      rdfs:label "http://snomed.info/id"@en ; *
                      fhir:value "http://snomed.info/id"^^xsd:string ;
                      fhir:tag "system"^^xsd:Name .

<http://record#01341> rdf:type fhir:code ;
                      rdfs:label "406466009"@en ; *
                      fhir:value "406466009"^^xsd:string ;
                      fhir:tag "code"^^xsd:string .

<http://record#01342> rdf:type fhir:string ;
                      rdfs:label "House dust allergen"@en ; *
                      fhir:value "House dust allergen"^^xsd:string ;
                      fhir:tag "display"^^xsd:string .

<http://record#01343> rdf:type fhir:string ;
                      rdfs:label "House Dust"@en ;
                      fhir:Substance.name.value "House Dust"^^xsd:string .
```

Record#01338 is the substance instance (OWL Named Individual). The name.value field containing "House Dust" is mapped to the rdfs:label annotation so it will show up as that in the OWL tool.
Its "type" as an Object Property identified as Substance.type points to record#01339

Record#01339 is an instance of fhir:Coding and has an rdfs:label of the Coding.display.value so it shows up (displays) in the OWL tool. The “code” and “system” are concatenated into the standard URI for the term which is also applied as a rdf:type to the coding thus providing the graph closure to the terminology if imported. If the terminology is not imported the system and code are still retained independently.

rdf:type are explicitly declared here but may be inferred by the reasoner based on the domain and range of the object properties (e.g. Substance.type) or from the fhir:tag which is the tag applied to the instance element and populated according to Ontology constraints.

Below is the graph of the example shown in Protégé Ontograph with closure to types and to SNOMED (sct:) where its label (in this case) is identical to the label of the Coding instance..



4 Mapping of Coding ComplexType (Model)

The mapping of the Coding type from FHIR Coding to OWL is described as follows:

FHIR Model element	OWL Type/Property	Comment
Coding complex type extends Element	Class– subclass of Element	In line with transliteral mapping from Complex Object. Object properties have to be gathered from Element
Coding.Id (Element.Id)	record: <URI>	If element.id is empty it will be generated on the RDF side as it is required.
Coding.extension		Tbd – will cover this in discussion of extensions
Coding.system	Object property – links to Class fhir:uri	
Coding.version	Object property – links to Class fhir:string	
Coding.code	Object property – links to Class fhir:code	

Coding.display	Object property – links to Class fhir:string	
	Coding.display.value Mapped to Coding rdfs:label	Allows tool to show display
Coding.primary	Datatype Coding.primary (bool)	Verbatim
Coding.system Coding.version Coding.code	Mapped to Coding rdf:type	Allows the Coding instance to bind to the imported Ontology if available.
Coding.valueset	Class – represents valueset Resource	More work is to be done here to match valueset to terminology concepts