To: Joint Steering Committee for Development of RDA

From: Gordon Dunsire, CILIP representative

Subject: RDF representation of RDA relationship designators: discussion paper

Introduction and background

The RDA glossary defines a **relationship designator** as “A designator that indicates the nature of the relationship between entities represented by authorized access points, descriptions, and/or identifiers”. The term “designator” is not included in the glossary, but the two closest terms suggest that a designator ***identifies*** something.

Intended usage, therefore, requires at least three components: two related entities and one relationship designator.

RDA appendices I, J, and K list relationship designators for relationships between resources and agents, between resources, and between agents, respectively. A **resource** is “a work, expression, manifestation, or item” (i.e. FRBR group 1) and **agents** are “persons, families, and corporate bodies” (i.e. FRBR/FRAD group 2); these terms are used in this document for the sake of brevity. Appendix L is a place-holder for designators for relationships between FRBR group 3 entities.

RDA uses English terms as designators. The terms used are nouns, and each term has an associated definition.

RDF representation

RDA relationship designators can be represented in Resource Description Framework (RDF) in two distinct ways:

1. Element set[1]: as a **property** and/or **class**
2. Value vocabulary: as a **concept**

Each of these three cases is discussed separately below.

The RDA designators are currently represented as RDF properties in element sets:

App	Scope	Element set URI	Qname
I	Resource-Agent	http://rdvocab.info/roles	roles
J	Resource-Resource	http://rdvocab.info/RDARelationshipsWEMI	RDARelationshipsWEMI
K	Agent-Agent	none	none
L	FRBR Group 3	none	none

A **qname** is an abbreviation for the namespace base domain used to reduce the length of URIs in triples by using curies (compact URIs)[2]; for example *roles:authorWork* instead of

[1] The terms “element set” and “value vocabulary” are taken from the W3C Library Linked Data Incubator Group report, available at: http://www.w3.org/2005/Incubator/lld/XGR-lld-vocabdataset-20111025/

http://rdvocab.info/roles/authorWork. These qnames are given in the Open Metadata Registry (OMR), but are just a recommendation for consistency; a user is free to use their own abbreviations with no operational impact. The qname does not need to follow the URI of the base domain.

Discussion: The current registered qnames for all of the RDA element sets are inconsistent. There is no RDA branding in "roles", while "RDARelationshipsWEMI" is overlong and potentially unclear to external communities. Although use of the OMR qnames is optional, many application developers and data publishers prefer a consistent approach and are happy to take them.

It is not bad practice to change the RDA qnames because the element set namespaces are not yet in published status. Changing the qname does not affect the URIs, which can remain as they are.

Recommendation: Change the qnames in the OMR to be consistent and include RDA branding.

Approaches taken by other communities

The introduction to the Vocabulary Mapping Framework (VMF)[3] says:

"… relators have always been widespread in metadata, but until recently have often been disguised as categories or roles. For example, all of the substantial contributor role vocabularies in schemes such as ONIX, MARC and DDEX are actually relators describing the relationship between the resource being described and some contributing party. The main reason for this sometimes superficial confusion lies in naming. For example, the ONIX contributor list (code list 17), which uses a mixture of role terms such as "Actor" and relator terms such as "Abridged by", demonstrates this. "Actor" in this vocabulary is in fact a relator meaning "is actor in" (in context it is not saying that John Smith is an actor by profession, only that he acted in this particular resource). As a further example of the "hidden" prevalence of relators, of the Dublin Core 15 terms, only six describe wholly-owned attributes (title, identifier, description, type, format and coverage) while nine are relators from the resource to other independent entities (creator, contributor, publisher, source, relation, rights, date10, language and subject, although the rights element is normally used in practise as a description and does not point to a particular entity). Metadata statements are increasingly recognized as being bi-directional: Shakespeare is metadata about Hamlet, and Hamlet is metadata about Shakespeare, depending on the starting point of view of any particular scheme, and the role of the relator in characterising relationships between entities is of course fundamental."

The VMF itself is itself an extension of the RDA/ONIX framework[4], and used the RDA relationship designators as one of its source vocabularies.

Searching for RDA designator terms in the Linked Open Vocabularies (LOV) service[5] shows that most communities take the same approach of creating RDF properties to represent roles and other

[2] CURIE Syntax 1.0: a syntax for expressing Compact URIs. Available at: http://www.w3.org/TR/curie/
[3] The Vocabulary Mapping Framework (VMF): an introduction. v1.0, December 12, 2009. Section 2.4.4. Available at: http://www.doi.org/VMF/documents/VocabularyMappingFrameworkIntroductionV1.0%28091212%29.pdf
[4] RDA/ONIX framework for resource categorization. Version 1.0. Released August 1, 2006. Available at: http://www.rda-jsc.org/docs/5chair10.pdf
[5] LOV service is available at: http://lov.okfn.org/dataset/lov/index.html

relationships between entities. These include the Dublin Core Metadata Initiative (for example *dct:creator*) and MARC 21 (for example MARC relator codes).

Element set representation

Properties

An RDF property relates or links:

a) two individual entities.
b) an individual entity and a literal string.

Pattern a) full supports linked data because both subject and object can be linked to other triples; pattern b) results in the termination of a linked data chain because the literal string cannot be machine-matched to any other triple. Pattern a) is therefore generally more useful. Pattern a) corresponds to RDA entities with identifiers (URIs); pattern b) corresponds to RDA entities with descriptions (literal strings). Authorized access points can usually be used in either pattern.

A property forms the middle, predicate part of a metadata statement in the form of a subject-predicate-object data triple, and must be a URI. A property is uni-directional between entities; its domain corresponds to the subject part, and its range to the object part.

Discussion: The representation of RDA designators as properties is the best fit with the Glossary definition; the designator links two entities. This is also consistent with approach taken by communities with close relationships with RDA.

Recommendation: Continue to represent RDA relationship designators as RDF properties.

Domains and ranges

The domain and range of a property consist of zero, one, or more RDF classes. The use of multiple classes tends to occur in complex ontologies or application profiles, and is currently rare in basic namespaces.

Semantic reasoner software infers that the subject of a data triple using the property is an individual member of the domain class(es), while the object is a member of the range class(es). For example, the data triple <*individualURI1*> <*roles:compilerWork*> <*individualURI2*> can generate a new data triple <*individualURI1*> <*is-a*> < *FRBRentitiesRDA:Work*>: the entity identified by *individualURI1* must be a Work because the domain of *roles:compilerWork* is *FRBRentitiesRDA:Work*. No range is specified, so nothing can be inferred about *individualURI2*.

Discussion: The ranges of the properties in the *roles* element set (based on appendix I) have not been registered pending further discussion on the "Agent" class (see the Classes section below). Without the agent class, it would be incomplete to register just one of *FRBRentitiesRDA:Person, FRBRentitiesRDA:Family,* or *FRBRentitiesRDA:CorporateBody* as the range, and it would be false to register more than one. Technical note: multiple domains and ranges are interpreted as unions of each class, whereby the attributes are assumed to be common. This would lead to an expectation, for example, that an attribute specific to a Family, say *ElementsGr2:familyHistory*, is applicable to a Person or Corporate Body.

Recommendation: Add the existing "Agent" super-class for the range of the RDA designator properties in the *roles* element set.

Unconstrained properties

A property with no domain or range is often referred to as "unconstrained", "unbound", "free", etc. to indicate that nothing can be inferred about the subject or object of corresponding data triples, so publishers of linked data can use the property without risk of increasing semantic incoherency in the Semantic Web.

Discussion: A parallel set of unconstrained properties has been added to each of the RDA element sets. In several instances there is redundancy because an unconstrained property always has a broader semantic than the constrained original, leading to semantic convergence. For example, *roles:compilerWork* has the unconstrained parallel property *roles:compiler*, and *roles:editorOfCompilationExpression* has the unconstrained parallel property *roles:editorOfCompilation* – but the definitions show that both unconstrained properties are the same, so one is redundant and can be removed or made equivalent to the other.

Recommendation: Identify and remove redundant unconstrained properties representing RDA relationship designators.

Property labels

It is bad practice to use the same label for more than one property within an element set, and the OMR prevents this from occurring. However, the FRBRized and unconstrained versions of RDA properties are currently in the same element set, and so require differentiated labels. This has been achieved by adding the FRBR entity names used as domains and ranges to the vocabulary term or element name from the Toolkit. For example "compiler (Work)" is the current label for the constrained property and "compiler" for the unconstrained property. Work is the FRBR entity that is the domain of the constrained property. The range of the property will become Agent if the recommendation above is accepted, but it is not necessary to add this to the label because the context of the *roles* element set allows the assumption that "compiler" is a type of agent. Technical note: The "role as type of agent" semantic is represented in RDF as a class, as discussed below in the Classes section.

The need for label differentiation can be removed, in the main, by placing the unconstrained properties in a separate namespace. There are exceptions when the attribute or designator applies to more than one FRBR entity; for example "Contact information" is an attribute of Manifestation and Item and "Composer" is a designator for Work and Expression.

Recommendation: Move the unconstrained versions of properties in all RDA element sets to separate namespaces.

The entity names can be dropped from property labels without changing the URIs. It is not bad practice to do this before the status of the label becomes "Published". Labels for properties for attributes and designators assigned to more than one entity can remain as they are.

Recommendation: Simplify the property labels in all RDA element sets, except for those requiring entity differentiation.

Inverse properties

An inverse property swaps the domain and range, and the pairing of a property and its inverse ensures that linked data triples are connected bi-directionally. Technical note: the inverse property cannot be used in pattern b) where a data triple using the original property has a literal as its object.

Discussion: No inverse properties for RDA designators have been added to the namespace. It is sufficient for JSC or an external community to just create a URI for an inverse property and declare it as inverse to the existing property; semantic reasoner software can invert the domain and range and generate corresponding data triples automatically, while human users can mentally invert the definition. However, it would improve uptake of the namespace representation of the designators if JSC created and maintained the inverse properties as a trusted organization, and to minimise delay. It would also improve uptake if reliance on external software such as a semantic reasoner was minimised.

Recommendation: Develop and register inverse properties for RDA relationship designators, including domains and ranges.

It is therefore useful to create a verbal phrase as the label associated with the property URI, for example "has compiler (Work)" rather than the current "compiler (Work)", and add the inverse property labelled "is compiler of (Work)" to the namespace. This is the approach taken by IFLA namespaces.

Note that it is possible to relate a property to other labels. For example, it is feasible to related a designator property to the SKOS preferred term given in the parallel value vocabulary, as discussed in the Value vocabularies section, so that the value vocabulary term can be used in displays and other outputs. That is, if there is a parallel value vocabulary term, it is possible to have a verbal label for the property, but use the non-verbal term for display of the property. For example, "has compiler" as the general label for the property, and "compiler" as the "record display label" taken from the value vocabulary. This is similar to how the designator is displayed in the Toolkit: the display/value vocabulary label is used, but the verbal phrase reflects the underlying semantic of a relationship between two entities.

It is also useful to clearly indicate the directionality of the property within its definition; that is, which type of entity is the subject/domain, and which type of entity or literal is the object/range.

Advantages:

- Disambiguation of property, class, and concept labels (see the Value vocabulary representation section below).
- Reduction of entity dependence on property directionality.

Disadvantages:

- De-coupling[6] of namespace content from its source documentation (the RDA Toolkit).
- Additional cost of separate maintenance, although costs can be reduced by use of syntactical patterns, etc. for labelling entities and relationships.

Recommendation: Replace the labels of properties for RDA relationship designators with verbal phrases, and ensure that definitions of properties indicate directionality between domain and range entities.

Recommendation: Replace the labels of properties for all RDA elements with verbal phrases, and ensure that definitions of properties indicate directionality between domain and range entities.

Classes

An RDF class defines a set of individual entities with shared characteristics.

As discussed above, classes can be used to add semantics to a property, as domains and ranges which can infer the class membership of an individual entity and therefore the expected characteristics of the entity.

Discussion: The hypothetical relationship designator class *roleclasses:CompilerWork* would have the definition "A person, family, or corporate body responsible for creating a new work (e.g., a bibliography, a directory) through the act of compilation, e.g., selecting, arranging, aggregating, and editing data, information, etc". This implies the existence of a super-class for "a person, family, or corporate body" (the class "Agent" noted above), a related class labelled "work", and a property representing the "act of compilation". The last two are already in the RDA namespace, as *FRBRentitiesRDA:Work* and *roles:compilerWork* respectively.

The super-class "Agent" defines additional characteristics, such as the common attributes of a person, family, or corporate body.

The super-class is also already in the RDA namespace, as *FRBRentitiesRDA:Agent*. It is an extension to the FRBR-for-RDA element set which has no equivalent in FRBRer, but is equivalent to the FRBRoo/CRM class "Actor". The super-class is implied by the standardized phrasing of "person, family, or corporate body", etc. in the RDA Toolkit.

See the Domains and ranges section above about the impact of not using this class for *roles* designator ranges.

Advantages:

- Improved semantic representation of the compound entity.
- Overall reduction in element set entities by avoiding the need to have three sets of properties and/or classes for treating persons, families, and corporate bodies separately in relationships.

[6] De-coupling means the wording of the Toolkit and the namespace is different, and that irregularities in English grammar prevent simple transformations between them. As a result, complete automatic synchronization of changes to terms, derived labels, etc. is not possible, so manual intervention and some duplication of effort is required.

- Reduction of cost of element set maintenance.

Disadvantages:

- De-coupling of namespace content from its source documentation (the RDA Toolkit).
- Additional cost of separate maintenance.

As noted above, *roleclasses:CompilerWork* can be used as the range of the property *roles:compilerWork*. The data triple <*individualURI1*> <*roles:compilerWork*> <*individualURI2*> would then generate a new data triple <*individualURI2*> <*is-a*> <*roleclasses:CompilerWork*>; that is, the agent in question is a compiler of a work.

This has utility when it is necessary to assign distinctive characteristics to different roles occupied by the same agent. For example, compilers need to find, identify, and select the components of the compiled work; they will interact with other agents over copyright, design, and publishing matters in a different context than authors do; etc. This approach, however, leads to specification rather than generalization.

Representing the relationship designators purely as classes, and not as properties, requires a new, general property, say *roleclasses:hasRole*, to relate the designator class to the agent; an example data triple is <*agentURI*> <*roleclasses:hasRole*> <*roleclasses:CompilerWork*>. But this does not indicate the individual work involved. This approach focuses on the FRBR Group 2 entities, as the entities of primary interest, rather than the Group 1 entities.

Recommendation: Do not represent the RDA designators as RDF classes. The utility of doing so currently lies outside of the main focus of RDA.

Value vocabulary representation

A value vocabulary "represents a controlled list of allowed values for an element. Examples include: thesauri, code lists, term lists, classification schemes, subject heading lists, taxonomies, authority files, digital gazetteers, concept schemes, and other types of knowledge organization systems".

Discussion: The hypothetical relationship designator term/concept *role:Compiler%20(Work)* would have a *skos:prefLabel* property with the value "Compiler (Work)" and a *skos:definition* property using the same RDA Toolkit definition as for the element set class approach.

Such a term/concept has utility as the object of a data triple; for example <*agentURI*> <*roleclasses:hasRole*> <*role:Compiler%20(Work)*>. This triple would link to a value vocabulary triple giving the preferred label for display. Technical note: there is no utility in using the term/concept as the object of a designator-specific property, because the specific property already carries the semantics.

The range of the general "has role" property can be specified to be a concept/term from the *role* value vocabulary using an Application profile, but no further inferencing is possible without additional ontological data.

This approach is therefore useful for the construction of authorized access points for agents and resources, providing values for elements such as 9.16 Profession or Occupation and 6.6 Other Distinguishing Characteristic of the Work.

There has been significant discussion of related issues on RDA-L recently[7][8].

Advantages:

- Source of controlled terms for agent and resource attributes.
- Improves consistency of authorized access points.

Disadvantages:

- Scope of attribute is wider than scope of relationship designator.

Recommendation: Represent RDA designators as RDF concepts in one or more value vocabularies in addition and in parallel to the current representation as RDF properties. This allows the designators to be used as controlled terms in the construction of authorized access points, and as the content of some other RDA elements.

Representing hierarchical relationships between relationships designators

RDA relationship designators are grouped at several levels of hierarchy. For example “compiler” is a sub-value of the implied general designator “creator”, and “editor of compilation” is a sub-value of “contributor”. Other examples are “actor” < “performer” < “contributor”, and “dramatization of (work)” < “adaptation of (work)” < “derivative of (work)”.

It is easy to represent such hierarchies in RDF.

The current designator element set properties are related in the namespace using the *rdfs:subPropertyOf* property.

If designators were represented as classes, they could be related using the *rdfs:subClassOf* property.

If designators are represented as value vocabularies, as recommended, they can be related using the *skos:broader* property (and its inverse, *skos:narrower*).

Top-level elements

The implied uppermost element of each hierarchy (Agent for Appendix I, FRBR Group 1 for Appendix J, and Agent for appendix K) varies depending on the RDF representation.

For the element set property representation, the top-level element is just “relationship between entities” which would be the RDF property “has relationship”. It is not necessary to add this to the namespace.

[7] http://www.mail-archive.com/search?l=rda-l%40listserv.lac-bac.gc.ca&q=relationship+designator
[8] http://www.mail-archive.com/search?l=rda-l%40listserv.lac-bac.gc.ca&q=relationship+designators

For value vocabulary representation, the top-level element is the concept/term for the entity, for example "actor" < "performer" < "contributor" < "agent" or "dramatization of (work)" < "adaptation of (work)" < "derivative of (work)" < "work". The latter is more clearly expressed as "dramatization" < "adaptation" < "derivative" < "work". Note that this is easily confused with the element set class representation, where the same hierarchy would involve classes for "dramatized work", "adapted work", and "derived work". This is not recommended, as discussed in the Classes section above, and it is not necessary to add such top-level terms to the proposed value vocabularies.

Recommendations

1. Change the qnames in the OMR to be consistent and include RDA branding.
2. Continue to represent RDA relationship designators as RDF properties.
3. Add the existing "Agent" super-class for the range of the RDA designator properties in the *roles* element set.
4. Identify and remove redundant unconstrained properties representing RDA relationship designators.
5. Move the unconstrained versions of properties in all RDA element sets to separate namespaces.
6. Simplify the property labels in all RDA element sets, except for those requiring entity differentiation.
7. Develop and register inverse properties for RDA relationship designators, including domains and ranges.
8. Replace the labels of properties for RDA relationship designators with verbal phrases, and ensure that definitions of properties indicate directionality between domain and range entities.
9. Replace the labels of properties for all RDA elements with verbal phrases, and ensure that definitions of properties indicate directionality between domain and range entities.
10. Do not represent the RDA designators as RDF classes.
11. Represent RDA designators as RDF concepts in one or more value vocabularies in addition and in parallel to the current representation as RDF properties.

Gordon Dunsire
30 September 2012