Don Quixote

Records

Selected to illustrate the following:

- Different expressions of the same work
  - translations or other derivatives
  - where the main entry person (creator) is considered to be the original author (in library terminology)
- Other derivations
  - main entry person is not different from the original author
- Publications with augmentations/supplements
  - In addition to the main text there are illustrations, forewords and other additions that are of interest to find
- Works that have parts
  - What we now refer to as the work Don Quixote was originally two different works published separate in time

Records are mainly from National Library of Spain and Bodleian Library (UK) and can be found in Europeana. All records are in MARC and retrieved from the original institutions using Z39.50. Most records include a 856-field with a link to a digital resource with the exception of a few (which are alternative MARC records for comparable resources in Europeana).

FRBRoo modelling

The records have been modelled in three rounds. The first round of modelling was done manually to explore different use of classes and properties. Afterwards, we have used a rule-based interpretation of the MARC records to produce rdf and diagrams.

The two different modelling mainly differ in the following way:

**Derivation-based**
Translations are modelled as a self-contained expression that is the realisation of a "translation-work" and derivative relationship between the translation-work and the "original" work.

**Realisation-based**
Translations are modelled without the use of separate works for each translation. All translation-expressions are realisations of the same work. Derivations are still used for abridgements/adaptations (see record 016754131)

---

1 Using PLANTUML: http://plantuml.sourceforge.net
Other modelling decisions

• **Using F1 Work (used in both modelling approaches)**
  We did not come across any issue that required the use of specialisations of F1 Work.

• **Using P148 to relate the part of the work to the whole (used in both)**
  We consider this to be the preferred solution for modelling works with parts. The alternative is to use Complex Work and *has member*, but this is potentially ambiguous because this property can be used for alternatives to other members of the work as well as components of the overall concept.

• **F28 Expression Creation (used in both)**
  Persons are related to works and expressions by the use of F28 Expression Creation Event. This is consistently used in both models for all works.

• **F27 Work Conception (only used in realisation-based models)**
  In the models we needed a way to distinguish between the “original” creator of the “first/representative” expression and creators of subsequent derivations. The first is what we typically consider to be the author and the latter are persons with secondary responsibility (translators etc). This requirement can be solved by using “in the role of” for P14 which means that we need to enhance of use of FRBRoo with an additional type system for P14. As an alternative to this solution we have explored the use of F27 because it potentially is a much easier (and FRBRer-like) way of distinguishing the person we consider as the primary creator from persons that otherwise only are considered to be contributors (in DC-terminology). The current definition, however, of this class implies that it is not intended for this purpose.

• **“Original expressions” (only used in derivation-based models)**
  In the Derivation-based models we have created entities for the “original expressions” in all records. This means that even if the record describes a translation (with an expression created by the translator) we have created an entity for an expression that was created by Cervantes. Original expressions are not included in the realisation-based models. The main reason for this difference was to explore the effect of this instance when mapping to EDM.

• **Manifestation Product Type and Publication Expression (used in both)**
  All records are modelled using Manifestation Product Type and a Publication Expression that incorporates one or more Self-Contained Expression. We have attempted to identify all additional content from the use of relator codes (but this of course easily creates errors).

• **Always a Self-Contained Expression (both models)**
  To have a consistent use of entities across all records we are always using a Publication Expression that incorporates at least one Self-Contained Expression.

---

2 imho this should be avoided for two good reasons. Properties of properties are difficult to implement in rdf and other systems (although it can be solved with subtyping). Secondly, I think it is bad of the FRBRoo model requires that we have to implement and use additional type-systems to express very common information.
Example FRBRoo models

The following two figures show some of the design choices in our alternative models (see [http://november.idi.ntnu.no/frbrized/rest/db/edm/index.html](http://november.idi.ntnu.no/frbrized/rest/db/edm/index.html) for complete set of records).

**Derivation-based**

![Diagram of Derivation-based FRBRoo models]
Mapping to EDM

Initial mapping rules
An initial solution for mapping our FRBRoo models to a corresponding EDM representation is shown in the following figure (this is the mapping used for the realisation-based models. I will make another for the derivation-based models, but they are comparable). The figure only shows the main entities and properties. Trivial properties (such as extent, publisher, date etc.) are probably rather straightforward to implement a mapping for when the core structure is defined.

Mapping single records is potentially a deceiving process. A solution that applies to one record may cause errors if applied to other records. Mapping from FRBRoo to EDM has to be verified by:

- Applying the mapping to a representative set of records that includes relevant variations
- Additionally, the mapping must be verified by evaluating the result from merging/combining records. Solutions that work well for each record in a set, does not necessarily give the desired result when merging entities from multiple records.

Mapping rules explored
We have explored mapping using FRBRoo records in rdf as input for an XSLT-based mapping procedure. The result of the process is EDM in rdf which then has been transformed to graphs using plantUML.

This is a preliminary description of ideas and challenges:

- **edm:ProvidedCHO for F3 and F24**
  Because of the 1:1 relationship between F3 Manifestation Product Type and F24 Publication Expression, both entities can be mapped to a single edm:ProvidedCHO.
• **edm:incorporates and edm:InformationResource**
  Publication Expression that incorporates multiple Self-Contained Expression are mapped using an InformationResource for the Self-Contained Expression. 
  *PS! Ideally we wanted to avoid using an Information Resource when there is only a single Self-Contained Expression, but it turns out that this creates inconsistencies between records where this expression is the only content and records where this expression is one of several. This is an essential issue that needs some more work.*

• **edm:Agent**
  All persons are treated as first class objects and mapped to edm:Agent

• **dc:creator and dc:contributor**
  Persons with primary responsibility for the resource should be expressed as dc:creator (authors eg.) and persons with secondary responsibility should be expressed using dc:contributor.
  For both frbroo-models it turns out to be challenging to determine if person is a creator or contributor. Although it theoretically is possible to reason over the graph-structure, it is not something that we should rely on. The use of Work Conception (as used in the realisation-models) seems to be a relevant and easy solution.

• **mapping F1 Work and F22 Self-Contained Expression to edm:InformationResource instances**
  The use of edm:InformationResource for both F1 Work and F22 Self-Contained Expressions seems to be unproblematic. The graphs should be able to answer queries related to persons names, titles etc regardless of the actual type of instances. The most crucial is that the graph collocates InformationResource and ProvidedCHO instances using relevant properties.

• **Reducing the number of InformationResource objects**
  A straight-forward transformation of FRBRoo models to a corresponding structure based on InformationResource will in some records appear to create redundant objects. This is typically found in the realisation-based models (e.g. see realisation-based edm-model for bima0000007171).
  The reduction of entities was easy to implement for the derivation-based models. In this case we could consistently merge all works and expressions into a single InformationResource. This kind of merging is more difficult to implement consistently for realisation-based models. In some cases we want to have different InformationResources for eg. translation and the work it realises, but in other cases (mainly for the expressions created by Cervantes that realises his own work) it is sufficient to have a single InformationResource. This need to be explored a bit more.....
EDM examples

The following figures show the resulting edm records. The example shows that it is possible to create the same edm-models from both the derivation- and realisation-based models (with the exception of the use of dc:creator for the translator in the derivation-based). The example mappings also includes some "worst-case" mappings.
Example of redundant objects

The following example illustrates the redundant InformationResource problem when mapping realisation-based FRBRoo.

PS! This may be solved by merging InformationResource instances where one is the derivation of the other and they have the same contributor/creator…. sounds easy but it is a bit tricky to implement….