Mapping format for data structures of the CIDOC CRM

Martin Doerr

10/7/01

1. Formalism

The CRM is a formal ontology in the form of a semantic model.
In a structured record,

- No distinction is made between the nature of an attribute value and kind of relation expressed by the attribute between the record as a whole and the attribute value.
- Referred discrete entities are not explicitly identified, but rather referred to by some representative values.
- Information is arranged (sequence, nesting etc.).

A semantic model,

- declares each attribute value as an a priori independent entity in the universe of discourse and connects such values symmetrically by directed links/arcs/properties .
- The entities acquire identity independent from the composition of descriptional elements in some data structure.
- The structure of a formal ontology is supposed to reflect the organization of the universe of discourse as we perceive it, rather than that of the document we create (e.g., if not given graphically, the presentation in a document will have to serialize a multiply connected net in some way). It has an "ontological commitment" to the "underlying conceptualization" of its creators [Guar98].

Let's regard an example of a structured record like:

```
struct painting {
                    integer record_id;
                    string title;
                string kind_of_title;
                struct painter {
                                string artist_name; /* ULAN identifier*/
                            string nationality;
                                string contribution; /* role in the creation process */
                                } artist;
                    time creation_date_begin;
                time creation_date_end;
                string creation_date_comment;
                string creation_place; /* TGN identifier */
                    string last_exhibition;
                 } PAINTING;
```

In the following, CRM entities are referred by the unique id and name , eventually followed by a comment in parenthesis, like:

E7 Activity (of type "exhibition")

If a record with *record_id* = 5034 is to be transferred to the CRM, an instance of E22 has to be created with a unique identifier, e.g. "Object5034". But any other unique for the object combination of values can be used.

CRM links are referred by the unique id of the **applicable** entity in the given context, followed by the unique id of the entity from which the link is **inherited**, the name of the link and the name of the referred entity, like:

E22 (E7) was used for: Activity

We use as name of the link only the part associated with the direction applicable for the given context. E.g. We present the link "E11 Modification. has produced (was produced by)" as

E22 (E11) was produced by: Modification

if I want to read it from the product rather than from the event.

We have therefore chosen a mapping format, where for each PAINTING field we declare the correspondence of the **value** to a CRM entity, like:

last_exhibition = E7 Activity (of type "exhibition")

This means that for a transfer from PAINTING to CRM, the contents of the "*last_exhibition"* field are used to create a respective unique identifier of an E7 entity instance. The contents may also be used to derive additional attributes of the E7 entity instance, if they can be parsed, e.g. "Documenta 1998, Kassel" may be used to find date, title and place. The E7 instance is also assigned the type "exhibition" from a thesaurus.

In the sequence we declare the correspondence of the **relation** expressed by the field to a CRM link, like:

painting -> last_exhibition = E22 (E7) was used for: Activity

This means for a transfer from PAINTING to CRM that a property instance of type "was used for" is created, that connects the instance of E22 "Object5034" representing our example painting to the above created instance of E7 "Documenta1998, Kassel".

For a transfer from CRM to PAINTING, the instance of E7, that has type "exhibition" or a narrower term of it, is connected by "was used for" to the respective E22 entity, and that has the latest date associated, will be used to fill the field *"last_exhibition"* of record 5034.

If a substructure corresponds to a CRM entity, I declare the correspondence of the **relation** expressed by one of its field with respect to the substructure name, rather than the catalog record, like:

painter -> nationality = E21 (E1) has type: Type

If the relation expressed by an PAINTING field corresponds to a path with intermediate entities in the CRM, I have marked it as "(JOIN)", like:

painting -> painter = E22 (E11) was produced by – E12 (E7) carried out by: Actor
(JOIN)

This means, that for a transfer from PAINTING to CRM, an **intermediate** instance of E12 has to be created, with an identifier uniquely defined by the circumstances, like "Object5034-production". In the sequence, it is inserted appropriately between the node "Object5034" and the Actor instance, e.g. "Hans Blumenbaum Jr". This intermediate node may be bare of other attributes. However in this example, missing attributes can be found in the *painting* record, like *creation_date_begin.* Joins may be double or triple, requiring multiple intermediate nodes.

In the inverse transformation, i.e. from CRM to PAINTING, respective attributes of the E12 instance "Object5034-production" would be distributed to the *painting* record, but the identity of the event itself would be lost. The idea of an ontology like the CRM is, to connect properties directly to the entity they are causally related with, and not to the description target (here a painting).

This is the basic mapping mechanism. In the transformation from structured records to CRM, first a set of unique entity identifiers is created. These allow later to merge the compiled knowledge, and to retrieve connections not

visible in single documents. Eventually, the same event may be registered multiply, if not enough evidence for its identity is found. Heuristics may be run to deal with the merging of such cases. Then the entity instances are connected by property instances, and the equivalent CRM instance is ready. This in turn can be transformed into many other forms of records without further ambiguities.

The "has note" link can be used for any information about an entity not formally described by another entity or property in the CRM, as e.g. currently artist biography. It can also be used to capture all comments in the records as e.g. *creation_date_comment* . For fine-grain distinction of the semantic of such comments, the "has note" link has a type attribute.

Complete PAINTING-CRM Mapping:

title = E35 Title
painting -> title = E22 (E19) has title: Title

kind_of_title = E55 Type (Title Type)
painting-> kind_of_title = E22 (E19) has title - E35 (E1) has type: Type (JOIN)

painter corresponds to E21Person, a specialization of E39 Actor.
painting -> painter = E22 (E11) was produced by – E12 (E7) carried out by: Actor (JOIN)
painter-> artist_name can be used to create the unique identifier for the E39 instance. This would be an ideal convention to create E39 instances.
nationality = E55 Type (Person Type)
painter -> nationality = E21 (E1) has type: Type

contribution = E55 Type (Role Type)
painter-> contribution =E22 (E11) was produced by – E12 (E7) carried out by: (in the role of: Type)
This example of a link on a link, i. e. a role specifying the "carried out by" can be modeled in systems without this feature by a relation entity for the "carried out by" link. In RDFS e.g. I would declare a class "carrying out" with property "in the role of".

creation_date_begin = E61 Time Primitive, lower value
creation_date_end = E61 Time Primitive, upper value

painting->creation_date_begin /
painting->creation_date_end =
E22 (E11) was produced by – E12 (E2) has time-span - E52 (E1) at most within: Time Primitive (Double JOIN).

creation_date_comment= E62 String (Text attached to the time-span)
painting->creation_date_comment =
E22 (E11) was produced by – E12 (E2) has time-span - E52 (E1) has note: String (Double JOIN).
creation_place = E53 Place (TGN identifiers are good to create instances of E53)
painting->creation_place= E22 (E11) was produced by – E12 (E4) took place at: Place (JOIN).

2. The Map DTD:

The following template illustrates the logical structure of a mapping from some data structure (schema, DTD, etc.) to the CRM:

```
<mapping>
        <map>
                <domain_map>
                        <comment></comment>
                        <src_domain_condition>
                        </src_domain_condition>
                        <src_domain_entity></src_domain_entity>
                        <eq_crm_domain_entity></eq_crm_domain_entity>
                        <crm_domain_constraint></crm_domain_ constraint>
                </domain_map>
                <link_map>
                        <comment></comment>
                        <src_domain_condition></src_domain_condition>
                        <src_range_condition></src_range_condition>
                        <src_range_map>
                                <src_range_entity></src_range_entity>
                                <eq_crm_range_entity></eq_crm_range_entity>
                                <crm_range_constraint></crm_range_constraint>
                        </src_range_map>
                        <src_path></src_path>
                        <target_path>
                                <crm_link></crm_link>
                                <crm_interm_entity></crm_interm_entity>
                                <crm_domain_ constraint></crm_domain_ constraint>
                        </target_path>
                </link_map>
        </map>
</mapping>
```

E.g. the following statements in the notation used to map the Dublin Core Data Elements:

```
DCT1 physical object      =    E19 Physical Object
DC [E71 Man-Made Stuff]:
        DC.Title        =  E35 Title
        DC->DC.Title = E71 has title: Title

DC.Title.LANG  =  E56 Language
DC->DC.Title.LANG = E71 has title – E35 (E33) has language : Language (JOIN)
        where E33 Linguistic Object.
```

Would be tagged like:

```
<mapping>
        <map>
                <domain_map>
                        <src_domain_condition>
                                If DC.Type  =  DCT1 physical object
                        </src_domain_condition>
                        <src_domain_entity>
                                DC
                        </src_domain_entity>
                        <eq_crm_domain_entity>
```

```
                    E19 Physical Object
                </eq_crm_domain_entity>
            </domain_map>
            <link_map>
                <src_domain_condition>
                    <mapped_entity_of>DC</mapped_entity_of>
                    <op>            SUBCLASS_OF</op>
                    <value>         E71 Man-Made Stuff </value>
                </src_domain_condition>

                <src_range_map>
                    <src_range_entity> DC.Title</src_range_entity>
                    <eq_crm_range_entity>E35 Title</eq_crm_range_entity>
                </src_range_map>
                <src_path>          DC->DC.Title</src_path>
                <target_path>
                    <crm_link>      E71 has title: Title</crm_link>
                </target_path>
            </link_map>
            <link_map>
                <src_range_map>
                    <src_range_entity> DC.Title.LANG</src_range_entity>
                    <eq_crm_range_entity>E56 Language </eq_crm_range_entity>
                </src_range_map>
                <src_path>          DC->DC.Title.LANG</src_path>
                <target_path>
                    <crm_link>      E71 has title: Title</crm_link>
                    <crm_interm_entity>E35 Title</crm_interm_entity>
                    <crm_link>      E35  has language : Language</crm_link>
                </target_path>
            </link_map>
        </map>
</mapping>
```

References

[Guar98] Guarino N. Formal Ontology and Information Systems. In N. Guarino (ed.), "Formal Ontology in Information Systems". Proc. of the 1st International Conference, Trento, Italy, 6-8 June 1998. IOS Press