Commentary on the mapping process:
From BM to CIDOC-CRM

This document is an informal "log" of comments documenting the translation process from the British Museum model to CIDOC-CRM. It provides notes and reasons for decisions made, and may serve as a guide for performing a similar process for other datasets.

We started from the SQL-like schema provided from the existing system, considering one table at a time, forming "paths" in the CIDOC-CRM model.

In parallel to this, the fields which initiate these paths were found in the sample.xml dump, informing the construction of the config.xml file. Note that this is the final `config.xml`, which has evolved over the course of the mapping process. You may wish to start by looking at config-earlier-version.xml. The configuration is used with the `makeRDF` tool to generate the RDF models.

The following resources are created from the sample data —

- BCB12909 — Lindow man
- COC248969
- GAA79859
- JCF20544
- MCC3832
- MCC6489 — a multi-part clock
- PPA243492
- RFC32005
- WCO94017
- YCA62958 — the Rosetta Stone
- the-british-museum

A number of extensions to the CRM have been created, see bm-extensions.ttl.

An online version of the CRM ontology (and extensions) has been made available at http://crm.rkbexplorer.com/.

While working through this document and/or making a mapping, it can be useful to consult this resource. Adding /id/X12 to the end of http://crm.rkbexplorer.com/ will make a best effort guess at the real URI eg http://crm.rkbexplorer.com/id/P3 will redirect to a choice of properties (`P3F.has_note` and `P3F.parts_description`), and http://crm.rkbexplorer.com/id/`P79F.beginning` will redirect straight to the correct property, http://crm.rkbexplorer.com/id/`P79F.beginning_is_qualified_by`.

We now briefly summarise the tables which have been mapped. The extracts of config are included as a guide only, they are not complete, and can appear spread about throughout the configuration. You will have to look in the configuration files to find the final and complete version.

Some items early on in this description may have been changed later on, however they remain in their original form in this document. Again, see configs/config.xml which is the authoritative final representation.

Thesauri

Converted thesauri data and asserted. This was really a very straightforward activity, and can best be seen by just looking at the file, configs/thesaurus-config.xml.

Example resource — x83343 or x115902

Codex_Object

1. This is the core table representing each object. They are of type `E22.Man-Made_Object`, and identified by `Merlin_PRN`.

```
<mapping match="{/rmxml/mus_cat/bm_object}">
  <resource>
    <identifier prefix="&id;object-" value="{bm_prn}" />
    <type value="&crm;E22.Man-Made_Object" />
```

2. The `Object_id` is the alternative Codex ID. We record this as both an alternative URI and as a literal value for convenience when querying.

```
<triple predicate="bm:codex-id" value="{bm_codex_object_id/_}" />
<triple predicate="owl:sameAs" object="&id;codex-{bm_codex_object_id/_}" />
```

3. `Department_Code` maps to `P50F.has_current_keeper`. In the original files, these codes were specified in `extra-data.xml`, but later were subsumed in other files.

```
<triple predicate="crm:P50F.has_current_keeper" prefix="&id;department/" object="{bm_owning_department}" />
```

4. `BM_Object_Copyright_Ind` contains copyright info. We extend `P3F.has_note` with `P3F.has_copyright`. There is no copyright info in `sample.xml`.

```
<triple predicate="crm:P3F.has_copyright" value="{bm_copyright_ind}" />
```

5. `Object_Description` is another extension of `P3F`, `P3F.has_description`, however these appear within object parts in the `sample.xml`.
6. `Object_Dimension_List`. We represent each dimension as a `bnode` which has both `P2F.has_type` (width, height, etc) and a `P91F.has_unit` (cm, kg).

Because the actual value could be a range, we subclass `P90F.has_value` to give `PX.min_value` and `PX.max_value`. All dimenions have a `min_value`. If the value is a range, then `max_value` is also defined.

The measurements and units were defined in `extra-data.xml`, and were processed by the config file `extra-config.xml`.

```
<!-- dimension(s) -->
<mapping match="{mus_alias_dimension/_}">
  <bnode predicate="crm:P43F.has_dimension" type="crm:E54.Dimension">
    <triple predicate="crm:P2F.has_type" object="&id;measurement/{mus_dimension}" />
    <triple predicate="crm:P91F.has_unit" object="&id;units/{mus_dimension_measurement_unit}" />
    <triple predicate="bmx:min_value" value="{mus_dimension_value}" />
    <triple predicate="bmx:max_value" value="{bm_dimension_value_end}" />
    <switch>
      <case match="{bm_dimension_value_end[.!='']}">
        <!-- min and max value -->
        <triple predicate="rdfs:label" value="{mus_dimension} = {mus_dimension_value} - {bm_dimension_value_end} {
      </case>
      <default>
        <!-- just min value -->
        <triple predicate="rdfs:label" value="{mus_dimension} = {mus_dimension_value} {mus_dimension_measurement_u
      </default>
    </switch>
  </bnode>
</mapping>
```

7. `Acquisition` takes a little care, as in the CIDOC-CRM it depends on whether an object is being loaned, or has been acquired permanently — these are two different types of event: E8 and `E10`

We look for a `bm_acq_name_ass` of value `L` to indicate a loan

```
<switch>
  <case match="{bm_object_part/_/bm_alias_xml_acq_name/_/bm_acq_name_ass[.='L']}">
    <!-- acquisition: the object has been loaned, or some other such arrangement -->
    <triple predicate="crm:P30B.custody_transferred_through" object="&id;object-{bm_prn}-acquisition" />
    <resource>
      <identifier value="&id;object-{bm_prn}-acquisition" />
      <type value="&crm;E10.Transfer_of_Custody" />
      <triple predicate="crm:P29F.custody_received_by" object="&id;the-british-museum" />
      <triple predicate="crm:P30F.transferred_custody_of" object="&id;object-{bm_prn}" />
      <triple predicate="crm:P3F.has_note" value="{bm_object_part/_/mus_acquisition_note}" />
     </resource>
```

```
    </case>

    <default>
      <!-- acquisition: the object is now owned by the BM -->
      <triple predicate="crm:P24B.changed_ownership_through" object="&id;object-{bm_prn}-acquisition" />
      <resource>
        <identifier value="&id;object-{bm_prn}-acquisition" />
        <type value="&crm;E8.Acquisition" />
        <triple predicate="crm:P29F.custody_received_by" object="&id;the-british-museum" />
        <triple predicate="crm:P24F.transferred_title_of" object="&id;object-{bm_prn}" />
        <triple predicate="crm:P3F.has_note" value="{bm_object_part/_/mus_acquisition_note}" />
      </resource>
    </default>
  </switch>
```

8. We also insert the date of acquisition into each of the above blocks

```
  <mapping match="{bm_alias_acq_year/_}">
    <date text="{bm_acq_year_text}" earliest="{bm_acq_year_earliest}" latest="{bm_acq_year_latest}" comment="{bm_acq
  </mapping>
```

9. And in the case that the object is not loaned, record that the object is owned by the BM

```
  <triple predicate="crm:P52F.has_current_owner" object="&id;the-british-museum" />
```

10. `No_Of_Objects` — maps to `P57F.has_number_of_parts`

```
  <triple predicate="crm:P57F.has_number_of_parts" value="{mus_number_of_objects}" />
```

Object_layer

This table describes parts of an object. Each object has at least one part (most only have one). A lot of the information is attached to the part rather than the object. This fits the model well, and is consistent with multi-part objects, but may be slightly counter-intuitive as you might expect these details to be attached to the object for a single part object rather than it having one part.

1. `Object_part_id` indicates the part of an object. Each part is a `E24.Physical_Man-Made_Thing`, identified by a concatenation of the parent `Object_id` and `Object_part_id`. We reference from the object to the part with `P46F` and from the part back to the object with `P46B`.

```
  <mapping match="{bm_object_part/_}">
    <triple predicate="crm:P46F.is_composed_of" prefix="&id;object-" object="{../../bm_prn}-part-{mus_obj_parts}" />
    <resource>
      <identifier prefix="&id;object-" value="{../../bm_prn}-part-{mus_obj_parts}" />
      <type value="&crm;E24.Physical_Man-Made_Thing" />
      <triple predicate="crm:P46B.forms_part_of" object="&id;object-{../../bm_prn}" />
```

2. We have a variety of comments/notes, represented as `P3F.has_note` or subclasses thereof. Insert into relevant object, acquisition or findspot details.

```
  <triple predicate="bm:PX.physical_description" value="{mus_physical_description}" />

  <triple predicate="crm:P3F.has_note" value="{bm_object_part/_/mus_acquisition_note}" />
```

Object_Production_Date

We assume there is only one production event for a given part. A clock for example that has multiple parts can have different data in each part, of course. If there are multiple possible dates for a given part then that single event will have more than one time span.

```
    <mapping match="{mus_alias_object_production_date/_}">
      <triple predicate="crm:P108B.was_produced_by" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../
      <resource>
        <identifier prefix="&id;object-" value="{../../../../bm_prn}-part-{../../mus_obj_parts}-production" />
        <type value="&crm;E12.Production" />
        <triple predicate="crm:P108F.has_produced" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../m
        <date text    ="{mus_object_production_date_text}"
              earliest="{mus_object_production_earliest}"
              latest  ="{mus_object_production_latest}"
```

```
            comment ="{bm_object_production_com}" />
        </resource>
      </mapping>
```

Object_Technique

We could have a technique without date, or vice-versa, so this looks a little repetitive (indeed the definition of the production event is duplicated, but when asserted into a triplestore this will not be a problem).

1. We associate the thesaurus term using `P32F.used_general_technique`

```
<mapping match="{mus_alias_technique/_}">
  <triple predicate="crm:P108B.was_produced_by" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../mus_
  <resource>
    <identifier prefix="&id;object-" value="{../../../../bm_prn}-part-{../../mus_obj_parts}-production" />
    <type value="&crm;E12.Production" />
    <triple predicate="crm:P108F.has_produced" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../mus_o
    <triple predicate="crm:P32F.used_general_technique" prefix="&id;thesauri/" object="{mus_technique_th_i}" />
    <triple predicate="bm:PX.has_technique_note" value="{bm_technique_note}" />
  </resource>
</mapping>
```

Object_Production_Place

This represents a place which is in some way related to the production of the object. Various values of `mus_object_production_place_association` indicate different things, and we create sub-properties of `P7F.took_place_at`. Some are sub-properties of others.

These relationships are in the form *object production* event which *took place at* the specified *place*. You must be happy replacing any of the following sub-properties with *took place at*, otherwise they should not be sub-properties of `P7F.took_place_at`.

```
<mapping match="{mus_alias_object_production_place/_}">
  <triple predicate="crm:P108B.was_produced_by" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../
  <resource>
    <identifier prefix="&id;object-" value="{../../../../bm_prn}-part-{../../mus_obj_parts}-production" />
    <type value="&crm;E12.Production" />
    <triple predicate="crm:P108F.has_produced" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../m
    <switch>
      <case match="{mus_object_production_place_association[.='A']}">
        <triple predicate="bm:PX.attributed_in" prefix="&id;thesari/" object="{mus_object_production_place_th_
        <triple predicate="crm:P3F.has_note" value="Attributed at {mus_object_production_place}" />
      </case>
      <case match="{mus_object_production_place_association[.='CF']}">
        <triple predicate="bm:PX.claimed_to_be_from" prefix="&id;thesauri/" object="{mus_object_production_pla
        <triple predicate="crm:P3F.has_note" value="Claimed to be from {mus_object_production_place}" />
      </case>
      <case match="{mus_object_production_place_association[.='D']}">
        <triple predicate="bm:PX.designed_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i
        <triple predicate="crm:P3F.has_note" value="Designed in {mus_object_production_place}" />
      </case>
      <case match="{mus_object_production_place_association[.='DE']}">
        <triple predicate="bm:PX.decorated_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_
        <triple predicate="crm:P3F.has_note" value="Decorated in {mus_object_production_place}" />
      </case>
      <case match="{mus_object_production_place_association[.='E']}">
        <triple predicate="bm:PX.engraved_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i
        <triple predicate="crm:P3F.has_note" value="Engraved in {mus_object_production_place}" />
      </case>
      <case match="{mus_object_production_place_association[.='F']}">
        <triple predicate="bm:PX.factory_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}
        <triple predicate="crm:P3F.has_note" value="Factory in {mus_object_production_place}" />
      </case>
      <case match="{mus_object_production_place_association[.='I']}">
        <triple predicate="bm:PX.issued_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}"
        <triple predicate="crm:P3F.has_note" value="Issued in {mus_object_production_place}" />
      </case>
```

```
        <case match="{mus_object_production_place_association[.='L']}">
          <triple predicate="bm:PX.lustred_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}
          <triple predicate="crm:P3F.has_note" value="Lustred in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='M']}">
          <triple predicate="bm:PX.made_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}" /
          <triple predicate="crm:P3F.has_note" value="Made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MB']}">
          <triple predicate="bm:PX.bell_made_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_
          <triple predicate="crm:P3F.has_note" value="Bell made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MC']}">
          <triple predicate="bm:PX.case_made_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_
          <triple predicate="crm:P3F.has_note" value="Case made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MD']}">
          <triple predicate="bm:PX.dial_made_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_
          <triple predicate="crm:P3F.has_note" value="Dial made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='ME']}">
          <triple predicate="bm:PX.ebauche_made_in" prefix="&id;thesauri/" object="{mus_object_production_place_
          <triple predicate="crm:P3F.has_note" value="Ebauche made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MI']}">
          <triple predicate="bm:PX.minted_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}"
          <triple predicate="crm:P3F.has_note" value="Minted in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MM']}">
          <triple predicate="bm:PX.movement_made_in" prefix="&id;thesauri/" object="{mus_object_production_place
          <triple predicate="crm:P3F.has_note" value="Movement made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MP']}">
          <triple predicate="bm:PX.watch_pendant_made_in" prefix="&id;thesauri/" object="{mus_object_production_
          <triple predicate="crm:P3F.has_note" value="Watch pendant made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='MQ']}">
          <triple predicate="bm:PX.dust_cap_made_in" prefix="&id;thesauri/" object="{mus_object_production_place
          <triple predicate="crm:P3F.has_note" value="Dust-cap made in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='O']}">
          <triple predicate="bm:PX.overpainted_in" prefix="&id;thesauri/" object="{mus_object_production_place_t
          <triple predicate="crm:P3F.has_note" value="Overpainted in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='P']}">
          <triple predicate="bm:PX.painted_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}
          <triple predicate="crm:P3F.has_note" value="Painted in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='PH']}">
          <triple predicate="bm:PX.photographed_in" prefix="&id;thesauri/" object="{mus_object_production_place_
          <triple predicate="crm:P3F.has_note" value="Photographed in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='R']}">
          <triple predicate="bm:PX.printed_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i}
          <triple predicate="crm:P3F.has_note" value="Printed in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='RT']}">
          <triple predicate="bm:PX.retailed_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i
          <triple predicate="crm:P3F.has_note" value="Retailed in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='W']}">
          <triple predicate="bm:PX.workshop_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_i
          <triple predicate="crm:P3F.has_note" value="Workshop in {mus_object_production_place}" />
        </case>
        <case match="{mus_object_production_place_association[.='Z']}">
          <triple predicate="bm:PX.published_in" prefix="&id;thesauri/" object="{mus_object_production_place_th_
          <triple predicate="crm:P3F.has_note" value="Published in {mus_object_production_place}" />
        </case>
      </resource>
    </mapping>
```

The following codes also imply a `E11.Modification` — `DE`, `E`, `L`, `O`, `P` so we add the additional

```
<type value="&crm;E11.Modification" />
```

Finally, we have two more types of note

```
<triple predicate="bm:PX.production_place_note" value="{mus_object_production_place_note}" />
<triple predicate="bm:PX.production_place_site" value="{bm_object_production_place_site}" />
```

Object_Producer_Name

The association of a company/organisation/person with the production of an item. They are not all listed here, see configs(configs/config.xml).

Some important points to note — most of the production person associations are sub-properties of `P14F.carried_out_by` so you must be happy to replace any of the predicates with *production event* was *carried out by* the *person or organisation*.

Certain things definitely don't fit with that, eg `AL`, `AT`, `CF`, `CM`, `X` which essentially indicate manner/style of, so these are not sub-properties.

Code `IR` is a special case, the CRM has means to specifiy that the object carries an inscription, and then we can say the inscription has a production event and that was carried out by the given person. We do not actually have any further information on the inscription, but it could be useful if people were looking for objects that carry inscriptions as expressed natively in CRM rather than custom predicates.

```
<mapping match="{bm_alias_xml_object_production_person/_/mus_object_production_person_association[.='IR']}">
  <triple predicate="crm:P128F.carries" object="&id;object-{../../../../../bm_prn}-part-{../../../mus_obj_part
  <triple predicate="crm:P3F.has_note" value="Carries an inscription which was created by {../mus_object_produ
  <resource>
    <identifier value="&id;object-{../../../../../bm_prn}-part-{../../../mus_obj_parts}-inscription" />
    <type value="&crm;E34.Inscription" />
    <triple predicate="crm:P3F.has_note" value="Created by {../mus_object_production_person}" />
    <triple predicate="crm:P94B.was_created_by" object="&id;object-{../../../../../bm_prn}-part-{../../../mus_
    <resource>
      <identifier value="&id;object-{../../../../../bm_prn}-part-{../../../mus_obj_parts}-inscription-producti
      <type value="&crm;E12.Production" />
      <triple predicate="crm:P94B.has_produced" object="&id;object-{../../../../../bm_prn}-part-{../../../mus_
      <triple predicate="crm:P14F.carried_out_by" prefix="&id;thesauri/" object="{../mus_authority-bm_auth_bio
    </resource>
  </resource>
</mapping>
```

The codes `AA`, `AF`, `AI`, `AM`, `C` and `N` indicate that the production was carried out by an anonymous person, although the association code specifies something about them (often in relation to another person or group). We model this by saying the production event was carried out by an anonymous actor, which then has properties as prescribed. For example:—

```
<case match="{mus_object_production_person_association[.='N']}">
  <bnode predicate="crm:P14F.carried_out_by" type="crm:E39.Actor">
    <!-- anything we know about this anonymous person/organisation -->
    <triple predicate="bmx:PX.pupil_of" prefix="&id;thesauri/" object="{mus_authority-bm_auth_biog_number}" />
  </bnode>
  <triple predicate="crm:P3F.has_note" value="Pupil of {mus_object_production_person}" />
</case>
```

Object Ethnic Group

Hopefully this is fairly straight forward, as we only have three codes. `AW` is a "fuzzy" relationship again, so is outside CRM. `IR` maps to `P62F.depicts`, but note `M` (made by) is applied to the production event, whereas the others are triples attached directly to the relevant object part.

```
<mapping match="{bm_alias_eth_name/_}">
  <switch>
    <case match="{bm_eth_name_ass[.='M']}">
      <!-- the object/part was made by the given ethnic group -->
      <resource>
```

```
        <identifier prefix="&id;object-" value="{../../../../bm_prn}-part-{../../mus_obj_parts}-production" />
        <type value="&crm;E12.Production" />
        <triple predicate="bmx:PX.made_by_ethnic_group" object="{../bm_eth_name_th_i}" />
        <triple predicate="crm:P3F.has_note" value="Made by ethnic group {../bm_eth_name}" />
      </resource>
    </case>
    <case match="{bm_eth_name_ass[.='AW']}">
      <!-- object/part is associated with the given ethnic group -->
      <triple predicate="bmx:PX.is_related_to" object="{../bm_eth_name_th_i}" />
      <triple predicate="crm:P3F.has_note" value="Is related to {../bm_eth_name}" />
    </case>
    <case match="{bm_eth_name_ass[.='IR']}">
      <!-- object/part is a representation of the given ethnic group -->
      <triple predicate="crm:P62F.depicts" object="{../bm_eth_name_th_i}" />
      <triple predicate="crm:P3F.has_note" value="Depicts {../bm_eth_name}" />
    </case>
  </switch>
</mapping>
```

Object Associated Place

Strange ones include NI (so it is a good job we modelled that inscription earlier), NS, OF, RP and UT which all involve other largely unknown objects or events (unfortunately these mean the use of more b-nodes, which we were trying to avoid!).

```
<mapping match="{bm_alias_as_place/_}">
  <switch>
    <case match="{bm_as_place_ass[.='AW']}">
      <triple predicate="bmx:PX.is_related_to" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
      <triple predicate="crm:P3F.has_note" value="Is related to place {bm_as_place}" />
    </case>
    <case match="{bm_as_place_ass[.='EE']}">
      <triple predicate="bmx:PX.is_emblem_of" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
      <triple predicate="crm:P3F.has_note" value="Is emblem of {bm_as_place}" />
    </case>
    <case match="{bm_as_place_ass[.='IT']}">
      <triple predicate="crm:P62F.depicts" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
      <triple predicate="crm:P3F.has_note" value="Is topographic representation of {bm_as_place}" />
    </case>
    <case match="{bm_as_place_ass[.='MF']}">
      <triple predicate="bmx:PX.made_for" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
      <triple predicate="crm:P3F.has_note" value="Made for {bm_as_place}" />
    </case>
    <case match="{bm_as_place_ass[.='NI']}">
      <!-- object carries an inscription, which names this place -->
      <triple predicate="crm:P128F.carries" object="&id;object-{../../../../bm_prn}-part-{../../mus_obj_parts}
      <resource>
        <identifier value="&id;object-{../../../../bm_prn}-part-{../../mus_obj_parts}-inscription" />
        <type value="crm:E34.Inscription" />
        <triple predicate="crm:P67F.refers_to" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
        <triple predicate="crm:P3F.has_note" value="Names place {bm_as_place}" />
      </resource>
      <triple predicate="crm:P3F.has_note" value="Carries an inscription which names place {bm_as_place}" />
    </case>
    <case match="{bm_as_place_ass[.='NS']}">
      <!-- the production used some raw material which came from the given place -->
      <triple predicate="crm:P108B.was_produced_by" prefix="&id;object-" object="{../../../../bm_prn}-part-{..
      <resource>
        <identifier prefix="&id;object-" value="{../../../../bm_prn}-part-{../../mus_obj_parts}-production" />
        <type value="&crm;E12.Production" />
        <bnode predicate="crm:P16F.used_specific_object" type="crm:E19.Physical_Object">
          <triple predicate="crm:P53F.has_former_or_current_location" prefix="&id;thesauri/" object="{bm_as_pl
          <triple predicate="crm:P3F.has_note" value="Originally from {bm_as_place}" />
        </bnode>
        <triple predicate="crm:P3F.has_note" value="Produced using material from {bm_as_place}" />
      </resource>
    </case>
    <case match="{bm_as_place_ass[.='OF']}">
      <!-- this is a copy, the original is from... -->
      <triple predicate="crm:P3F.has_note" value="Original from {bm_as_place}" />
      <bnode predicate="bmx:PX.is_copy_of" type="crm:E24.Physical_Man-Made_Thing">
```

```
            <triple predicate="bmx:PX.has_copy" prefix="&id;object-" object="{../../../../bm_prn}" />
            <triple predicate="crm:P53F.has_former_or_current_location" prefix="&id;thesauri/" object="{bm_as_plac
            <triple predicate="crm:P3F.has_note" value="Originally from {bm_as_place}" />
          </bnode>
        </case>
        <case match="{bm_as_place_ass[.='RP']}">
          <!-- has undergone repair in... -->
          <triple predicate="crm:P3F.has_note" value="Repaired in {bm_as_place}" />
          <bnode predicate="bmx:PX.repair" type="bmx:EX.Repair">
            <triple predicate="crm:P7F.took_place_at" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
          </bnode>
        </case>
        <case match="{bm_as_place_ass[.='UN']}">
          <!-- unspecified?! -->
          <triple predicate="bmx:PX.is_related_to" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
          <triple predicate="crm:P3F.has_note" value="Is related to {bm_as_place}" />
        </case>
        <case match="{bm_as_place_ass[.='UT']}">
          <!-- some activity took place at <place> and used this specific object -->
          <triple predicate="crm:P3F.has_note" value="Was used in an activity which took place at {bm_as_place}" /
          <resource>
            <type value="&crm;E7.Activity" />
            <triple predicate="crm:P7F.took_place_at" prefix="&id;thesauri/" object="{bm_as_place_th_i}" />
            <triple predicate="crm:P16F.used_specific_object" prefix="&id;object-" object="{../../../../bm_prn}" /
          </resource>
        </case>
      </switch>
    </mapping>
```

Object Associated Name

These are mostly straightforward, and inscriptions make a return. We introduce a new concept of `EX.Repair`, a subclass of the `E11.Modification` activity.

See config.xml

Object Associated Event

Events do not have a formal identifier in the thesauri. It appears events either have a very specific name, eg “The uprising of the Borg, 1702”, or something very generic, such as “Easter”. In either case it seems sensible to associate all events of the same name together (the very specific or a general vague concept), so we have decided initially to generate a URI for events based on `md5(event_name)`. Usually this is a bad idea, but think that in this case the benefits of joining what are likely to be similar resources outweigh the dangers of getting it wrong. (Peter also says events are not often used).

Object Acquisition Association

This table has varying implications on the type of acquisition, and/or actors involved in the acquisition of an object. We mostly distinguish between transfer of custody (ie object now under the care of) and/or transfer of title (ie object now under the ownership of).

Some sub-properites of `P22F` and `P28F` to represent different types of acquisition. In addition the acquisition events are typed with `P2F.has_type` where appropriate.

See config.xml

Find-spots

The CRM does not have any explicit notion of discovery (excavation, recovered from wreck, etc), merely acquisition. We introduce the subclass `EX.Discovery` to represent the activity in which an object was discovered.

Predicate PX.was_discovered_by associates a EX.Discovery event with an object, which in turn can have P7F.took_place_at a given place. Subproperties of P3F.has_note are used to capture specific details of the discovery event.

mus_field_collection_place_association with codes E or F type the EX.Discovery event as either excavation or find.

Any mus_alias_stratigraphic_unit comments are also attached to the EX.Discovery

Associated Title

This is rather vague. A bm_as_title_ass code of TI implies that the title is associated with an inscription on an object, otherwise the title is to be associated with the general object part.

Alias Title, Series Title

These are textual titles, associated with the relevant object part using subclasses of P3F.has_note.

Condition

There is only ever one condition comment for each object part, PX.condition_note

Collections

In the BM sense a Collection differs from the strongly typed notion of an E78.Collection in the CRM (which has associated implications as to their curation) — it is a more loosely defined group of objects. We introduce EX.Collection (which is kind of confusing?!), PX.is_part_of_collection and PX.contains_item.

The name of the collection is a controlled list, so forms the basis of the identifier.

Denomination

There does not seem to be anything in the CRM to do with money/coins/currency? (Confirmed.)

New predicates PX.currency and PX.denomination to represent the type and value of a monetary item. Custom functions within the makeRDF tool applied using modifier="" to split single bm_denomination field into the two separate components. Currency is a controlled list, denomination is numeric value.

Object location

We only want to expose the location of an object if it is currently on display in a public gallery, ie bm_loc_avail == G. A new E53.Place is created on the fly based on the bm_loc value, to which the object is associated with P53F.has_former_or_current_location

Alternative identifiers

There are a number of alternative identifiers used with the BM, `bm_calc_reg_no_expr`, `bm_calc_big_no_expr`, `bm_calc_gr_catno`, `mus_alias_other_number`. These are represented using sub-properties of `P3F.has_note`.

Representation of dates and times

Dates in the XML dump are currently output in a variety of formats. The `<date>` directive in the configuration mapping takes these various inputs and produces [-]YYYYMMDD format.

We ran into a little issue in 4store in representing dates which did not work before 1901 (technically they were overflowing number of seconds before epoch). In current data we have represented dates as an `xsd:integer` in the format `[-]YYYYMMDD`.

4store has now been patched so that `xsd:dateTime` now works correctly, which is perhaps less ambigious than an integer. Note however that `xsd:dateTime` requires all fields to be populated, ie `[-]YYYY-MM-DDT00:00:00+00:00`

Bibliography

Joshan has produced bibliography-config.xml. This mostly uses the bibliontology to represent properties of books and journal publications.

Identifiers in the form `&id;/bibliography/xxxxxxxx`

Biography

People and institutions are represented in the Biography.

Joshan/Tim are producing a configuration.

For the moment the simplest of configs has been produced just creating an entity with `rdfs:label`.

Identifiers are in the form `&id;/person-institution/xxxxxxxx`.

Note that lots of predicates link to the biographical records.

Alias Escapement

One of the thesauri is for types of escapement. If we have a `bm_alias_escapement` entry then we infer that the object has a part which is an escapement.

```
<mapping match="{bm_object_part/_/bm_alias_escapement/_}">
  <triple predicate="crm:P46F.is_composed_of" prefix="&id;object-" object="{../../../../bm_prn}-part-escapemen
  <resource>
    <identifier prefix="&id;object-" value="{../../../../bm_prn}-part-escapement" />
    <type value="&crm;E24.Physical_Man-Made_Thing" />
    <triple predicate="crm:P46B.forms_part_of" prefix="&id;object-" object="{../../../../bm_prn}" />
    <triple predicate="crm:P2F.has_type" object="&id;thesauri/{bm_escapement_th_i}" />
    <triple predicate="bmx:PX.escapement_comment" value="{bm_escapement_com}" />
  </resource>
</mapping>
```

Authority

The `bm_alias_authority` represents one or more “authorities” under which an object was made (eg kings, rulers, etc). We attach to the production event custom predicates which are sub-properties of

PX.produced_under_authority_of which itself is a sub-property of P11F.had_participant.

There is also a regnal_date which at the moment is concatenated into a single predicate string.

Inscriptions

Inscriptions form an important part of objects, with many details recorded. An object P128F.carries an Inscription. We have seen these before, and attributed properties to it (including production event for an inscription).

We now have a number of other details, specifically about the content of the inscription. The inscription may be in different parts, so we say that *all* inscriptions are composed of at least one part, in a similar way to objects and object parts.

Inscription type is a controlled list, ideally we would have the definitive list of these inscription types.

The various properties are then applied to the appropriate inscription part.

```
<mapping match="{mus_alias_inscription/_}">
  <triple predicate="crm:P128F.carries" object="&id;object-{../../../../bm_prn}-part-{../../mus_obj_parts}-ins
  <resource>
    <type value="crm:E34.Inscription" />
    <identifier value="&id;object-{../../../../bm_prn}-part-{../../mus_obj_parts}-inscription" />
    <triple predicate="crm:P46F.is_composed_of" prefix="&id;object-" object="{../../../../bm_prn}-part-{../../
  </resource>
  <resource>
    <type value="crm:E34.Inscription" />
    <identifier value="&id;object-{../../../../bm_prn}-part-{../../mus_obj_parts}-inscription-{mus_inscription
    <triple predicate="crm:P2F.has_type" prefix="&id;inscription-type/" object="{mus_inscription_type}" />
    <triple predicate="bmx:PX.inscription_content" value="{mus_inscription_content}" />
    <triple predicate="bmx:PX.inscription_language" value="{mus_inscription_language}" />
    <triple predicate="bmx:PX.inscription_script" value="{mus_inscription_script}" />
    <triple predicate="bmx:PX.inscription_translation" value="{mus_inscription_translation}" />
    <triple predicate="bmx:PX.inscription_transliteration" value="{mus_inscription_transliteration}" />
    <triple predicate="bmx:PX.inscription_note" value="{bm_inscription_note}" />
  </resource>
</mapping>
```

Other aliases — state, school, subject, ware, series

We have a basket of other aliases.

- State represents the geo-political entity under which an object was produced, so is attached to the production event with PX.produced_in_state. Controlled list.
- School is a style or type of work, object parts are attributed as being of P2F.has_type
- Subject are entries into the thesauri, object parts associated as having PX.subject
- Ware identifies the type of object (pottery). Entries into the thesauri with PX.ware, subclass of P2F.has_type
- Series is a free-text entry, PX.type_series identifies objects as being of a particular type or series run of prints

Exhibitions / Exhibition labels

bm_exhib_hist is a free text field describing details if an object has been exhibited.

mus_alias_label are details of the little label that went next to an object describing it when on exhibition. This also specifies the name of the exhibition, which is a controlled list.

```
<mapping match="{mus_alias_label/_}">
  <triple predicate="bmx:PX.object_exhibition_label" value="{mus_label_text}" />
  <triple predicate="bmx:PX.object_exhibition_label_comment" value="{mus_label_com}" />
  <triple predicate="bmx:PX.appeared_in_exhibition" prefix="&id;exhibition/" object="{bm_label_exhib_name}" mo
</mapping>
```

Again, ideally we would have a definitive list of these exhibitions, so that they can be generated once in a separate config, rather than defined in-line at each occurrence in the main config.

Review

At this point we reviewed the configuration with Martin Doerr. Two significant changes have been made, the first dealing with parts of objects, and the second naming conventions for attributes and comments. In addition, the URI scheme has been changed to favour concatenation with forward slashes rather than hyphens, eg http://rs.rkbexplorer.com/id/object/ABC12345/production

Object Parts

Firstly, after much debate we have concluded that preserving the existing modelling relationship as described earlier whereby each object always consists of at least one part is largely nonsense and should not be preserved. While arguments were put forward earlier for retaining this minimum one part per object scheme, it has now been decided that only objects which are genuinely composed of multiple parts will be shown as having parts.

This is achieved in the configuration mapping by replacing all 1 indicating the “master” part with an empty value, and changing URIs in the configuration file to be in the form /id/object/{bm_prn}/{mus_obj_parts}/production.

The empty mus_obj_parts vanishes during xpath substitution (double slashes are replaced with single slashes automatically) and properties are correctly attributed to either the object as a whole (formerly part 1), or to a real-world part as described within the input data.

As a result, the production of object ABC123 which has no real-world part is output as /id/object/ABC123/production whereas a clock which has a separately described case may have a production event such as /id/object/XYZ987/case/production

Most of the configuration has not had to change (beyond replacing hyphens for slashes); however in a small number of cases `if` directives have been wrapped around certain statements (eg `P46F.is_composed_of`) such that they are only output if there is genuinely a real part.

Association of comments

Many fields within the BM model permit a comment to be made, often prescribing an uncertainty or other comment associated with the attribution of a value. (they’re often just a question mark!)

For example, a Chinese vase may have a production event associated with two different dynasties, representing uncertainty or differing opinions. Comments are often used in this case to indicate the more likely, eg by the curator adding the notes “probable” and “possibly” as appropriate.