fhiso

CFPS 36
(Call for Papers Submission number 36)

Proposal for Accommodating Custom Properties

Submitted by: Proctor, Tony

Type: Technical proposal

Created: 2013-04-03

Last updated: 2013-04-20

URL: Most recent version: http://fhiso.org/files/cfp/cfps36.pdf
This version: http://fhiso.org/files/cfp/cfps36_v1-1.pdf

Description: Proposal for accommodating custom properties for persons

Keywords: Properties, Customisation, Person

Contents

1. Abstract

This proposal suggests a way of accommodating custom properties for persons. Properties may be single-valued or multi-valued, and may have associated units. Reference to a Dublin Core semantic type can be attached if relevant.

2. Proposal

In this proposal, the term *property* refers to a discrete item of data extracted from a source of evidence. For instance, in the case of a census page, it could be a name, occupation, role, etc.

While a Data Model may make provision for a predefined common set of named properties, it cannot make that a limiting set, or try and enumerate all valid possibilities for researchers worldwide. We therefore need to accommodate custom properties, beyond that predefined set.

In order to support custom properties we have some basic core criteria:

- Name clashes must be avoided (between different contributions).
- No central registration scheme required (too onerous).
- Data must remain portable between all compliant products.

This is easily handled using a URI-based namespace, as incorporated into STEMMA. A description of this approach was submitted separately under the title Partially Controlled Vocabulary.

All named properties should only be defined once in any given dataset. This ensures that the associated namespace, data-type, and any other attributes are not duplicated.

As with other areas, there must be a well-defined separation of direct evidence and conclusions or inferences derived from it. For property values, this means the original value – as written or recorded, and with any Uncertain Character Format (UCF) – must be separate from the interpreted value

We therefore have a number of requirements of the data representation too:

- Need a conclusion value, or link, that's distinct from the recorded value.
- Must be able to represent basic data-types such as Text, Integer, Number (decimal), Date, and Boolean.
- Need a Measure data-type that has associated units.
- Need to be able to record a multi-valued property using a list of values for a particular property.
- If relevant then we need to be able to associate a Dublin Core tag with the property definition.
- Need higher-level data-types for person and place references.

2.1 Basic Data-types

Basic data-types must include integers (whole numbers), decimal numbers, text, dates, and Booleans. These must be expressed in a locale-independent fashion. Following established precedents, this means always using a period as a decimal point and representing Booleans as 1 (=true) or 0 (=False).

Dates should, in principle, use a standard encoding but we only have ISO 8601 which is not adequate for all cases. See my separate proposals on Calendars, Modified ISO8601, and Dual Dates. A date string encoded according to those proposals could be used to represent a date value from any calendar. However, for representing dual dates, or date imprecision, a larger date-entity structure may be required.

2.2 Measure

A special type of decimal value is needed where the quantity is a measure of something. The classic case is someone's age. Most cases will measure it in years, but for infants it may be in months, weeks, or even days.

When defining a named 'measure' property, a Units attribute could provide a list of valid unit terms, separated by commas, with the default being the first one. For instance:

```
<PropertyDef Name='Age'  Type='Measure' Units='y,m,w,d'/>

<Property Name='Age' Value='6.5' Units='w'>Six and a half
weeks</Property>
```

Note that the Value attribute represents an interpreted value, and hence a conclusion.

2.3 Multiple Values

Some properties may require a list of values. The classic case is a Role property since a person may have more than one applicable role at an event. An example is presented under the Illustration section.

```
<PropertyDef Name='name'  [Type='type']  ItemList='boolean'/>
```

If ItemList is true (i.e. '1') then the value may be an array of items, each of which is represented using an <Item/> element. Any interpreted value (including Units) is then associated with each item in the property list.

```
<Property Name='name'>
        <Item [Units='units'] [Value='value'] ... > originalvalue </Item>
        ...
</Property>
```

2.4 Dublin Core

The definition of a named property should provide an optional association with a semantic type as follows:

```
<PropertyDef Name='name'  [Type='type']  [DCType='dc-type']/>
```

In this STEMMA-like syntax, the semantic type is indicated by the DCType attribute which uses the Dublin Core vocabulary, e.g. DCType='DC.Title' or DCType='DC.Publisher.CorporateName.Address'.

Note that this is done via a separate attribute. It would be wholly wrong to assume a connection based on the property name alone.

2.5 Conclusion Link

In all cases of a property value, there is the recorded form (the evidence) and the interpreted form (the conclusion). It is tempting, but still wrong, to assume these are one and the same, and that no separation is required.

The recorded characters may be uncertain (involving UCF sequences), or a reference may be vague or ambiguous, or the interpretation may rely on data from elsewhere. The aim of using extracted property values is to make those interpretations and to use them to build a

conclusion model – otherwise, they might as well be left in the original source. Hence, a distinction between the original and the interpreted forms is essential.

```
<Property Name='DateOfEnrolment' Value='2013-01-01'> Last Tuesday
</Property>
```

In this example, the encoded value of a conclusion date is provided (1st January 2013) but the real written form contains insufficient details, by itself, to make that connection. If, for instance, the reference was in a letter dated 5th January 2013 then a reference note could be attached to the property to indicate how such a conclusion value was obtained.

This separation has a direct analogy with mark-up that identifies values in situ, in their respective sources. It further connects the concept of properties with extracted items of evidence from a source.

2.6 Person and Place References

This is another case of providing an interpreted value. The original value may be a person or a place reference (i.e. a personal name or a place name) but a separate connection is needed to link the reference to a conclusion person or place. For instance:

```
<PropertyDef Name='PersonalName'  Type='PersonRef'/>

<Property Name='PersonalName' ID='pAnthonyProctor'> Tony
</Property>
```

If the ID link is absent then these become simple named references with no identification. The optional ID attribute therefore provides an additional conclusion link.

3. Not Covered or Not Required

In the case of STEMMA, properties are equally applicable to Places as to Persons. This proposal only describes them in relation to Persons since that's more widely accepted. The balanced application to both Persons and Places would have to be a separate proposal.

No scheme is presented here for identifying the language associated with an original property value.

4. Illustration

This STEMMA example shows a Person that has multiple custom roles in relation to a club social event. STEMMA predefines a multi-valued property called 'Role' under its reserved http://stemma.parallaxview.co domain. This example uses the predefined Role property but defines custom values for it.

The example also defines a completely custom property to accommodate the person's membership details.

```
<Dataset Name='Multi_Role_Example'
xmlns:roles='http://mydomain.com/roles'
xmlns:props='mailto:name@emaildomain.com?subject=properties'>

<ExtendedProperties>
        <PersonProperties>
                <PropertyDef Name='props:MemberID'  Type='Integer'/>
```

```
        </PersonProperties>
</ExtendedProperties>

<Event Key='eClubSocial'>
        <When><Date><Value> 1960-06-09 </Value></Date></When>
        <Type> Social </Type>
</Event>

<Person Key='pGordonBennet'>
        <EventLnk Key='eClubSocial'>
                <Property Name='Roles'>
                        <Item> roles:Photographer </Item>
                        <Item> roles:Host </Item>
                </Property>
                <Property Name='props:MemberID'> 2314 </Property>
        </EventLnk>
</Person>

</Dataset>
```

In other words, Gordon Bennet was both the host and the photographer at the club meeting, and his membership ID was 2314.

5. Use Cases

Support for certain person properties is taken for granted, such as name, age, occupation, and place-of-birth. However, in the general case, we cannot foresee all possible requirements. If we don't have an open mechanism then users and vendors may adopt back-door approaches which will impact portability of the data.

6. References

STEMMA extended properties.
http://www.familyhistorydata.parallaxview.co/home/extensibility/extended-properties.

The registration mechanism used here is an example of the more general framework proposed to FHISO cfps 20, Partially Controlled Vocabulary. http://fhiso.org/files/cfp/cfps20.pdf.

A number of proposals connected with dates and calendars have been submitted to FHISO, and are relevant to this proposal. See Calendars (cfps 13), Modified ISO8601 (cfps 17), and Dual Dates (cfps 14).

STEMMA discussion of extended vocabularies.
http://www.familyhistorydata.parallaxview.co/home/extensibility/extended-vocabularies.