fhiso

CFPS 77
(Call for Papers Submission number 77)

A unified formalism for genealogical statements

Submitted by: Smith, Richard

Type: A comment on a submitted paper

Comment on: 4

Created: 2013-06-05

URL: Most recent version: http://fhiso.org/files/cfp/cfps77.pdf
This version: http://fhiso.org/files/cfp/cfps77_v1-0.pdf

Description: This paper proposes a way of unifying the various types of node in the formalism for modelling genealogical research proposed in CFPS 4.

Keywords: data model, architecture, statement, graph, RDF

Abstract

This paper considers and endorses the formalism proposed in CFPS 4 for modelling genealogical research in terms of several distinct types of node. A means of unifying these distinct nodes is proposed here, resulting in a single type of statement for all data, lowering the threshold for further extensions. A mechanism is added by which the 'thing' nodes of CFPS 4 can be declared to have a particular type, for example a place or an individual.

1 Introduction

Modelling Research, not Conclusions (CFPS 4) proposes a formalism for describing genealogical data which this paper supports [1]. In that paper, nouns such as people, places and events are represented by a *thing node*. By itself, a thing node contains no information, but it is something that can be subject to connection, properties and matches. It may prove convenient to allow some form of identifier to be associated with thing nodes in order that they can be referenced.

In CFPS 4, a *connection* is described as a 'node that has a type label, two references to thing nodes, and a single citation'. Whilst supporting that definition, this paper proposes an alternative way of conceptualising this relationship. A collection of things and connections as a *graph* — that is a collection of vertices joined by edges. The things are the vertices and the connections are the edges.

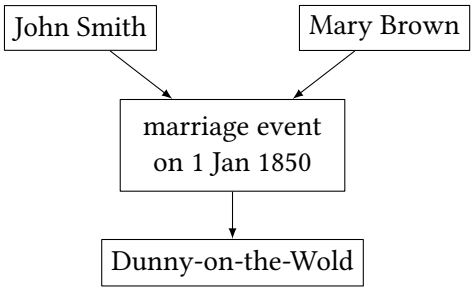

Figure 1: Graph representing three connections between four thing nodes.

Figure 1 gives an example of the thing nodes and connections in the sentence "John Smith and Mary Brown participated in a marriage at Dunny-on-the-Wold on 1 Jan 1850". Although figure 1 does not label the edges in the graph, the top two edges could be labelled "participated in" and the bottom one "happened at". These are the type labels mentioned in CFPS 4. Ignoring the citation (which will be discussed later), a connection can therefore be represented by a (first-node, type-label, second-node) triple.

A *property*, in CFPS 5, is something attached to a node carrying some sort of data. That paper suggests that the data can either be key–value pairs or free-form text. This paper suggests that that flexibility is not needed because an "comment" key-name can be used to associate free-form text with a node. Thus, it is suggested here that all properties can be expressed with a (node, key-name, value) triple. It is not specified in CFPS 5 whether properties can only be applied to thing nodes, or whether connection nodes (or even other property nodes) might be the subject of properties. However it seems likely that many of the important use cases will involve the attachment of properties to things.

2 Unifying properties and connections

At one level, connections and properties are distinct entities. But at another level, the only difference is that the former connects a node to another node, and the latter connects a node to a literal value. The type label on a connection and a key name on a property are not conceptually very different: both simply provide a way of classifying different types of connection or property. Presumably in the terminology of CFPS 20, the set of type labels and key names form a partially controlled vocabulary — one that can be extended by third parties [2].

Borrowing terminology from RDF [3], this paper proposes that connections and properties should be considered to be two forms of a common entity: the *statement*, which is represented by a (subject, predicate, object) triple. The *subject* is a node: for properties, it is the node the properties are attached to; for connections it is the first of the two nodes being connected. The *predicate* is a type label or key name. And the *object* is either another node (in the case of connections) or a literal value (in the case of properties).

In CFPS 4, another type of node is a *match*, which is used to say that one node represents the same thing as another node. For example, that the Jane Smith in one source is the same person as the Jane Smith in another source. It is easy to fold this into the general statement mechanism by introducing a "same as" predicate, much as free-form properties were converted into the key–value properties (and thence into statements) by introducing a "comment" predicate.

The picture in figure 1 was misleading as it placed data (in particular, names) in the thing nodes, but, as mentioned above, CFPS 4's thing nodes contain no data themselves. In fact, this data should have been moved into properties of the thing node. This is shown properly in table 1.

In the table, the thing nodes are represented with an ID which for clarity here has been prefixed with a '#'. (This paper does not propose that thing nodes must have an ID attached to them, or what syntactic form they should have. Their use

Subject	*Predicate*	*Object*
`#I1`	*type*	*person*
`#I1`	*name*	John Smith
`#I1`	*participated in*	`#E1`
`#I2`	*type*	*person*
`#I2`	*name*	Mary Brown
`#I2`	*participated in*	`#E1`
`#E1`	*type*	*marriage event*
`#E1`	*happened at*	`#P1`
`#E1`	*date*	`1850-01-01`
`#P1`	*type*	*place*
`#P1`	*name*	Dunny-on-the-Wold

Table 1: Eleven statements representing the information in figure 1

here is purely expository.) Three of the statements are connections — the three where the object is an ID representing a thing node. A further four statement — those with a "name" or "date" predicate — are properties. The values of the three "name" properties are arbitrary strings, whereas the "date" property's value has been encoded using the Gregorian calendar specified in CFPS 17 [4]. The four "type" statements are discussed in §3.

3 Classes and subclasses

The proposal in CFPS 4 says that thing nodes came in different types, 'be it a person, place, personal event, historical event', and presumably it intended there to be some way of specifying which of these a thing is. This paper introduces the term *class* to refer to the different types of thing node — for example, "place" and "event" are classes. The class could just be an attribute somehow associated with the thing node, but as CFPS 4's property nodes, which provide a mechanism for associating attributes with thing nodes, have been folded into the statement mechanism, this paper suggests that a statement is also used to associate a class with a thing node. This paper uses a "type" predicate for this purpose, and example of it are found in table 1.

The statements in table 1 contained an example of a "type" One of the statements in table 1 gave `E1` the class "marriage event". This goes further than just saying `E1` is an event. It tells the application that the event was a marriage rather than, say, a baptism. The "marriage event" class is a subclass of "event", and in general the set of classes form a hierarchy, some members of which are given in figure 2.

This hierarchy should be a partially controlled vocabulary (per CFPS 20), and third

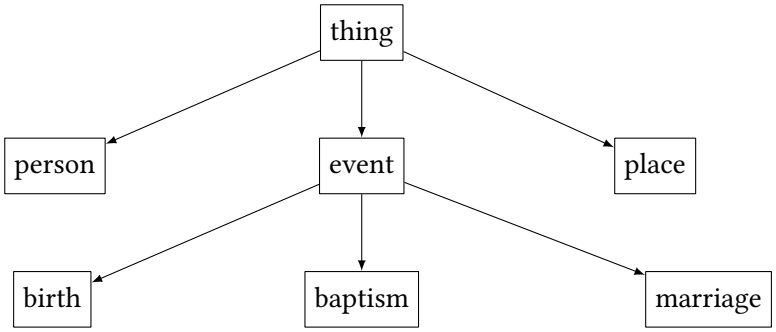

Figure 2: Part of the class hierarchy

parties would be free to define their own event classes, beyond those that the FHISO standardise. It may be desirable to introduce subclasses of other classes: for example, "country", "county" and "parish" as subclasses of "place", but that is beyond the scope of this paper.

The decision to make classes into statements is consistent with the CFPS 4 where it says a thing node 'express the assertion …that some noun exists'. Therefore the type statement isn't just a note of the thing node's class, but also a assertion on its existence. The result is that there is a one-to-one correspondence between nodes (in CFPS 4's terminology) and the statements in this paper. Even though thing nodes more accurately correspond to the subjects of this paper's statement, each thing node has precisely one type statement; and CFPS 4's connection, property and match nodes all correspond to statements in this paper.

4 Advantages to unification

This paper has proposed a unified statement formalism for four of node types from CFPS 4. Certainly this might be an æesthetic improvement, but so far no other justification has been made for it. There are, however, some significant benefits to this unification.

First, the unification lowers the threshold of adding new sorts of statement. As an example, it is likely that CFPS 4's nodes will need to be augmented with a mechanism for adding new terms in the various partially controlled vocabularies. Suppose there is no standard event for a bar mitzvah, an application serialising a document referring to bar mitzvah events might want to include a statement saying that "bar mitzvah event" is a subclass of "event". That definition statement does not fit within any of the sorts of nodes defines in CFPS 4, but the experience of RDF Schema demonstrates that it can accommodate in a general (subject, predicate, object) statement framework [5]. This paper does not propose that such def-

initions should necessarily be standardised, nor does it specify how they should be represented: that may be the subject of a future paper.

Second, because statements will presumably be serialised in a standard way, applications will be able to store unknown sorts of statement and include them in the output, even if the application cannot its derive any meaning from the unknown statemnt.

Third, if statements are unified, then it becomes relatively much easier to introduce a new form of statement where the subject is another statement. This paper does not propose a specific mechanism for that — it may be the subject of a future paper — but statements about statements would have a lot of use in genealogy. Attaching a source to a statement is a good example. It is the whole statement — the combination of the subject, predicate and object, rather than any specific one of them — that is being sourced. Other uses include noting the likelihood that a statement is correct, and associating other metadata with a statement.

5 Verbosity

The sentence "John Smith and Mary Brown participated in a marriage at Dunny-on-the-Wold on 1 Jan 1850" is represented by eleven nodes or statements. This may seem uneconomical, though in fact it is similiar to number of tags in the corresponding GEDCOM, as shown in figure 3.

```
0 @I1@ INDI                    0 @F1@ FAM
1 NAME John Smith              1 HUSB @I1@
1 FAMS @F1@                    1 WIFE @I2@
0 @I2@ INDI                    1 MARR
1 NAME Mary Brown              2 DATE 1 JAN 1850
1 FAMS @F1@                    2 PLAC Dunny-on-the-Wold
```

Figure 3: The information in table 1 expressed in GEDCOM

It is not within the scope of this paper to propose a serialisation for a collection of statements, but it seems reasonable to suppose that a general serialisation mechanism could be defined to allow the serialistion of arbitrary statements, and that it need not be any more verbose than GEDCOM.

6 Relationship to RDF

Although this paper has tried to harmonise its terminology for statements with that used in the various RDF standards, this paper specifically does not propose the use of RDF. Such a proposal would be premature here because RDF has certain of limitations, and while it is entirely compatible with the types of statement discussed in this paper, it is not certain that this will remain the case if new types of statement are introduced to cope with the specific problems.

Nevertheless, this paper suggests that it is desireable to harmonise, so far as is practicable, the terminology used to describe genealogical data with the terminology of RDF. Even if the FHISO decides against an RDF-based approach, if the development of semantic web continues along its present course, it seems likely that a vendor may wish to allow the export genealogical data as RDF.

It is worth noting that the abstract RDF data model (which is what is being discussed here) is quite distinct to the RDF/XML serialisation. The latter has a number of disadvantages, and several alternative approaches to RDF serialisation exist. Were the FHISO eventually to adopt an RDF-compatible data model, it could do so without adopting RDF/XML as its serialisation format, and, indeed, could avoid XML entirely if desired.

References

[1] Luther Tychonievich, 2013, *Modeling Research, not Conclusions* (CFPS 4), `http://fhiso.org/files/cfp/cfps4.pdf`

[2] Tony Proctor, 2013, *Proposal for Handling Partially Controlled Vocabularies* (CFPS 20), `http://fhiso.org/files/cfp/cfps20.pdf`

[3] World Wide Web Consortium, 2004, *RDF Primer*, `http://www.w3.org/TR/rdf-primer/`

[4] Tony Proctor, 2013, *Proposal to Accommodate Gregorian Dates using a Modified ISO 8601* (CFPS 17), `http://fhiso.org/files/cfp/cfps17.pdf`

[5] World Wide Web Consortium, 2004, *RDF Vocabulary Description Language 1.0: RDF Schema*, `http://www.w3.org/TR/rdf-schema/`