

2006 E-MELD Workshop on Digital Language Documentation

Wayne State University - Eastern Michigan University

Tools and Standards:
The State of the Art

June 20-22, in conjunction with the 2006 LSA Summer Meeting



**MODELING CONTESTED CATEGORIZATION
IN LINGUISTIC DATABASES**

By

Jeff Good

Max Planck Institute for Evolutionary Anthropology, Leipzig
and
The Rosetta Project, San Francisco

and

Calvin Hendryx-Parker
Six Feet Up, Inc.

Paper presented at

2006 E-MELD Workshop on Digital Language Documentation
Lansing, Michigan
July 20–22, 2006

Please cite this paper as:

Good, Jeff and Calvin Hendryx-Parker. (2006), Modeling Contested Categorization in Linguistic Databases, *in* 'Proceedings of the EMELD '06 Workshop on Digital Language Documentation: Tools and Standards: The State of the Art'. Lansing, Michigan. July 20–22, 2006.

1 Introduction

A fundamental problem in the design of linguistic databases is that much of the information which needs to be encoded is of a contested nature. That is, it involves data for which there is no general consensus on how best to interpret it. Perhaps the best example of this is the grouping of languages into genealogical units. Numerous proposals abound, but only a relatively limited subset of these proposals are not, in some way, contested. For example, while current wisdom accepts the existence of numerous low-level genealogical units (e.g., Germanic language or Algonquian language) and a number of high-level units (e.g., Indo-European or Niger-Congo), there is no general consensus on the grouping of the larger families together nor, in general, is there consensus on the arrangement of the subgroupings in those families. There are proposals, of course, some of which have more support than others, but there is no consensus.¹

In the ideal world (from a technical perspective), none of the information we may want to put into a database would be contested in any way. However, it is often undesirable to exclude such information from a database for both practical and analytical reasons. On the practical side, certain kinds of contested content may be very useful to the user of a database. Again, borrowing from the domain of genealogical relationships, navigating through information about the 7000+ languages of the world can be greatly facilitated by putting languages into genealogical groupings which allow the user to traverse up and down a tree to discover information about different languages. Since, at present, there is no non-contested family tree for all the world's languages, this sort of feature can only be implemented by using at least some contested information.

On the analytical side, if we want a database to be useful not only to impart “textbook” information but also to play a role in working on “cutting-edge” research problems, it will often be necessary for the database to include contested content so that different hypotheses embodied by that content can be compared to see how well each explains the data in question. For example, if there are two competing subgroupings of a given family, it could be quite useful to encode both in a single database. A user could then see how each subgrouping fares against different data sets and, perhaps, use the database to establish that one subgrouping is very likely to be correct—thus, shifting its status from contested to non-contested.

However, trying to make use of contested and non-contested data in the same database raises a number of technical issues. One of these is ensuring that the overall structure of the database does not crucially depend on contested content because of the likelihood that such content will be quickly rendered obsolete. Another is that, while it is clearly desirable to allow the user to compare contested hypotheses about the same data, it is not clear how best to encode contradictory information possibly embodied by those hypotheses within a single database structure, especially since the contradictory portions of the hypotheses may only to apply to a quite specific part of the database's overall content.

The present paper outlines one solution to this problem, as conceptualized and implemented with the Rosetta Project's All Language Database.² As will be seen, a solution first devised to

¹ Of course, it should go without saying that a “consensus view” should not be equated with the “correct view”. It simply reflects what one community believes to be correct based on the evidence available to it at any given point in time.

² <http://rosettaproject.org/>

deal with problems relating to a need for multiple, cross-cutting, and possibly contradictory, classifications of languages, dialects, and families, has been usefully extended to many other parts of the database which, while not contested from a linguistic point of view, tend to be “unstable” for various reasons and, therefore, in need of a model comparable to that for contested data. Furthermore, Rosetta’s technical requirements and desirable to create an interoperable, long-lived language database both converged on the same solution, and the present implementation of this system conforms in important respects to some of the best practice recommendations for content interoperability as envisioned by the Semantic Web framework.³

While much of the content of the Rosetta database is distinct in some ways from the content of the resources produced by documentary and descriptive linguistics, we believe that the general model for dealing with contested data described in this paper will be applicable to a wide range of language resources and linguistic data types. Therefore, even though the problem space covered in this paper does not coincide with the prototypical concerns of the ordinary working linguist, we believe that it may still contain a number of relevant lessons for designing tools and standards relevant to their work.

The structure of this paper is as follows. Section 2 gives a descriptive overview of the kinds of information found in the Rosetta database, breaking it down into contested and non-contested information. Section 3 describes our current implementation of the Rosetta database, with a focus on how the distinction between contested and non-contested information is maintained. Section 4 offers a brief conclusion.

2 Rosetta’s data structures

2.1 Introduction

The Rosetta Project has the goal of assembling a database containing all information available on the languages, dialects, and language families of the world. Such an ambitious goal will probably never be fully reached, of course. Nevertheless, having such a goal in mind has forced us to seriously consider what kind of database structure can best facilitate such an undertaking. Of particular concern has been devising a model which has enough structure to allow users to navigate through the information we have collected and perform meaningful searches on the data, on the one hand, while avoiding making our data model overly dependent on “unstable” content, on the other hand.

This data that is most problematic for Rosetta, in this regard, is that involving genealogical relationships between languages and language families. Important aspects the site’s functionality depends on having a complete genealogical classification of all the world’s languages in the database. In particular, such information allows (i) users to navigate up and down a “family” tree in order to move between languages and families in a user-friendly and intuitive way and (ii) allows the site to aggregate information both up the family tree and down the family tree so that users looking at information about a particular language, dialect, or family can easily discover how much information is available on linguistic units closely related to that language, dialect, or family.

However, as is well-known to linguists, there is no universally-accepted genealogical classification of the world’s languages. In order to provide such functionality, we must, therefore, choose a classification that is not universally accepted. Over time, we hope (in part through use of Rosetta’s

³ <http://www.w3.org/2001/sw/>

database) more and more parts of the classification will become universally accepted, and our database can be updated accordingly. This raises an immediate technical problem: How can we effectively use one classificatory system today but be able to easily replace it with another system tomorrow?

While the problems of genealogical classification are particularly salient in this regard, the same basic issues are found in a number of other types of information in the database. For example, in addition to genealogical navigation, the site supports geographic classification. While geographic relations are less contested than genealogical relationships, at least in linguistic circles, they are not necessarily stable—especially as regards politically-defined geographic units. At least for the foreseeable future, it is unwise to assume, in the context of database design, that the current political partitions of the world will remain constant. We, therefore, need to encode at least some aspects of the geographic content of the database in ways which afford similar flexibility to what is required for genealogical relationships.

Another problematic area of database content, worth mentioning in this context, is the basic classification of linguistic entities like languages, dialects, or families into such basic types. It is a truism in linguistics that the line between a language and a dialect cannot be firmly drawn. And drawing such lines is not only relevant to determining whether or not something is a language, it is also central in determining what is a family. If the varieties of Chinese, for example, are considered dialects, then Chinese would be a language. If they're considered languages, then Chinese would be a family. Therefore, in addition to being flexible about genealogical and geographic relationships, the database cannot even crucially rely on the idea that some named linguistic unit is a “language” or a “family”—since these sorts of classifications, too, may change over time.

In the rest of this section, we will discuss the conceptual model we have developed for the content of the Rosetta database to help deal with these issues. In section 3, we will, then, be in a position to discuss how we have been able to implement this conceptual model. In the discussion to follow, for the sake of concreteness, we will focus on problems relating to genealogical classification—though most of the points to be raised will be relevant for the other problems just mentioned, and they will sometimes also be referred to.

Rosetta's conceptual model rests on making a fundamental distinction between nodes and relationships. The former are understood to be relatively stable, non-contested parts of the database, and the latter are used to encode contested information.

2.2 Nodes

In the Rosetta database architecture, nodes correspond to stable, non-contested content. At present, there are two primary types of nodes, which are given in (1).

- (1) a. **Lingual Nodes:** Database elements corresponding to the linguistic entities, language, dialect, family, etc.
- b. **Geographic Nodes:** Database elements corresponding to geographic entities (both purely geographic and political)

Critically, for Rosetta, for an entity to qualify as a lingual node or a geographic node, it is not important whether or not its existence as a “real” entity is contested or not. All that is required is for someone to have referred to that entity. So, for example, any entity ever given a *Ethnologue*

code—in any version of the Ethnologue—could be treated as a node in the database. A node is, therefore, the analog of a *proposal* for a linguistic entity, not an analog of a linguistic entity itself. Whether or not an entity has been proposed will, generally, be non-contentious (one need only find a relevant reference). What is contentious is how that node relates to the real world.

At present, the Rosetta database, only contains nodes corresponding to Ethnologue 14 languages and families and, in some cases, nodes for dialects of entities classified as languages in Ethnologue 14. This has allowed us to avoid devising a solution to a problem that immediately arises from the idea that a node corresponds to a proposal and not to a real entity: How do we deal with proposed nodes which correspond to the “same” real-world entity, but might not be exactly the same thing. For example, is Greenberg’s Niger-Congo the same thing as Ethnologue 14’s Niger-Congo? There’s no easy answer to this question—both are referring to roughly the same proposed language family, but they will differ in details. While Rosetta has not yet had to implement a solution to this problem, we believe it can be accommodated in our current system and will briefly come back to this issue in section 3.5.

A Rosetta node has three properties, given in (2).

- (2) a. **Identifier:** Every node has a unique identifier.
- b. **Metadata:** Every node is associated with a limited amount of metadata. This includes, for example, a human-readable name for the node, information about what kind of node it is (e.g., lingual vs. geographic), and an indication of what resources the node is associated with (e.g., digital resources or references to books, articles, etc.)
- c. **Resources:** Each node can be associated with digital resources or references to books, articles, etc.

Figure 1 schematizes three Rosetta lingual nodes, one for English, one for Russian, and one for Slavic. Each node is marked with an identifier and is associated with a metadata record and a set of resources. In addition, the metadata record points to the resources the node is associated with. Strictly speaking, this information is not required for the system to be operative. However, modeling the structure this way has been proven convenient, in ways which become clearer in section 3.

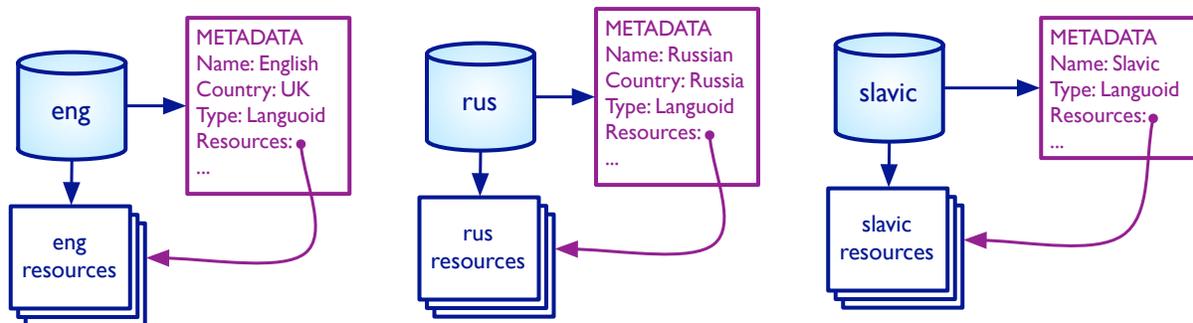


Figure 1: Nodes in the Rosetta database

As indicated in figure 1, a node can be understood as a kind of container, in which resources can be placed. The metadata associated with gives some basic information about what kind of container the node is and what it contains.

Critically, all nodes are conceptualized as belonging flat structure, separate from any information indicating how the nodes may be grouped genealogically or geographically—or in almost any other way. The modeling of such information is the topic of the next section.

2.3 Relationships

Content which is considered to be non-stable and potentially contested in the Rosetta database is modeled as relationships among nodes and between nodes and other data types. Such relationships can be schematized as in figure 2. Three possible kinds of relationships are given in the figure, a relationship where one node is the genealogical mother of another node, a relationship where one node is genealogical daughter of another node, and a relationship where a lingual node is classified as a language (as opposed to a family or a dialect).⁴

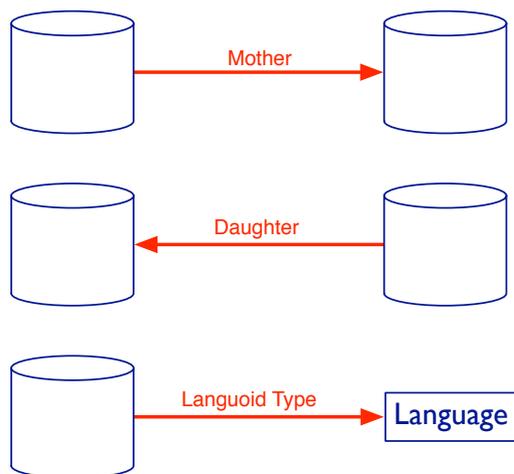


Figure 2: Relationships in the Rosetta database

As implied in figure 2, there are two broad classes of relationships in the Rosetta database, *nodal* relationships and *classificational* relationships. These are defined in 3.

- (3) a. **Nodal relationship:** A relationship holding between two nodes in the database. Prototypical cases includes mother/daughter relationships between nodes and relationships holding between lingual nodes and geographic nodes (e.g., that a language *X* is spoken in country *Y*).
- b. **Classificational relationship:** A relationship stating that a node is of some type or has some attribute. Prototypical cases include specifying whether or not a lingual node is

⁴The figure in (2) makes use of the term *languoid* which Rosetta uses, at present, as a cover term for any type of lingual entity: language, dialect, family, language area, etc. It is roughly similar to the term *taxon* from biological taxonomy, except it is agnostic as to whether the relevant linguistic grouping is considered to be genealogical or areal (or based on some other possible criteria for grouping languages). Another word which has been suggested (by David Gil) for this concept is *lange* [læŋ]. Further suggestions/advice on how to name this concept are actively solicited.

classified as a language, dialect, or family or specifying that a given node corresponds to a languoid as used by a group other than Rosetta.

Clearly, some of the relationships in the Rosetta database will be more contested than others. For example, many of the genealogical relationships are likely to be highly contested, while some of the classificational relationships are likely to be relatively non-contested. For example, many of the relationships in the database amount to statements like: “Rosetta lingual node *X* corresponds to Ethnologue entity with code *YYY*.” The reason for treating this kind of information as a relationship, rather than incorporating it directly into nodal metadata, is not that, in and of itself, it is contested or unstable. Rather, its utility is unstable. Another organization’s language classification may be quite useful at a particular point in time but less useful in the future. Today Ethnologue 14 and Ethnologue 15 codes may be in wide use, but tomorrow another system may become standard.

In order to assure that the Rosetta’s database structure will be valuable far into the future, we need to ensure that we limit our dependency on such external coding schemes as much as possible. However, since we clearly need to relate out lingual nodes to those propose by others, we do this via relationships and not via nodal metadata.

2.4 Combining nodes and relationships

To this point, the innovative nature of the Rosetta database—at least for linguistic work—may not be completely clear. Clearly, other databases have had node-like elements and relationship-like statements. The online version of the Ethnologue, for example, makes use of such a database, for example. What is different about Rosetta’s database structure (at least, we think it is) is that nodes and relationships are completely partitioned from each other and treated as independent data sets. Their only point of interconnection is the fact that relationship statements can refer to nodal identifiers.

To understand the nature of this partition of the data sets, it is useful to compare figure 3 and figure 4. Figure 3 schematizes a set of lingual nodes as an unstructured set of containers, holding information about lingual entities. All the nodes in figure 3, except one, refer to languages or families. The one exception is the node *Earth* which is used in the database as node corresponding to the set of all lingual entities spoken on Earth.

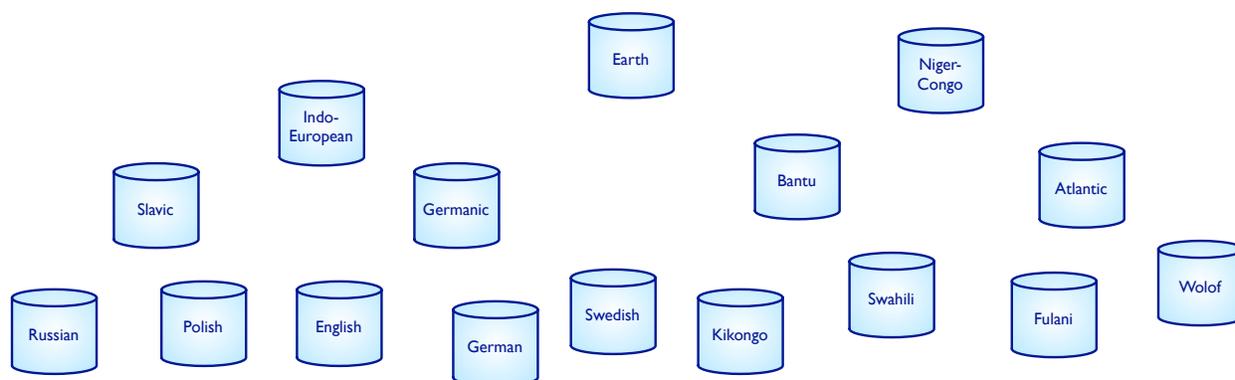


Figure 3: Subset of unstructured nodes from the Rosetta database

Figure 4 schematizes a set of (genealogical) mother-daughter relationships holding among the

lingual nodes depicted in figure 3. The “empty” containers in figure 3 are intended to schematize the fact that the relationships are not stated directly with respect to the nodes but, rather, only refer to the nodes.

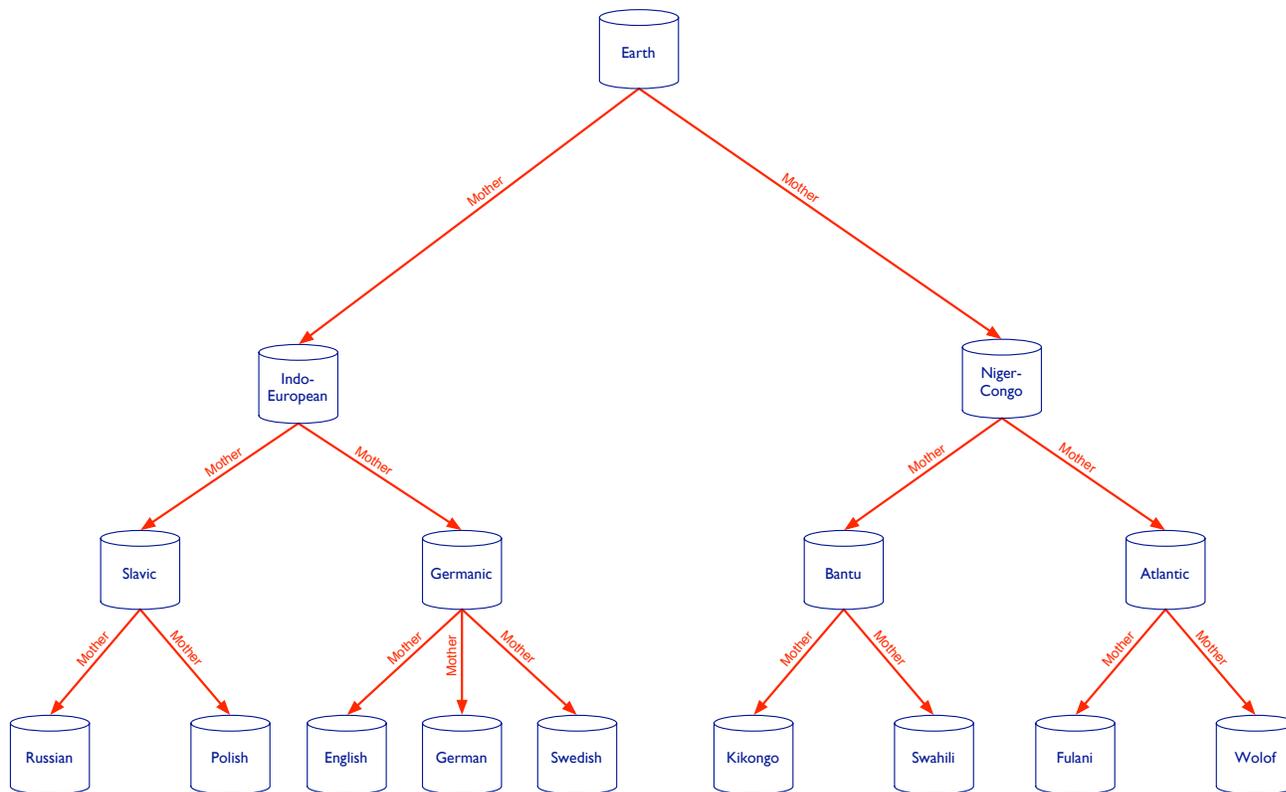


Figure 4: Subset of unconnected relationships from the Rosetta database

In order to make use of information included as part of lingual nodes and information encoded as relationships, the nodal data structures and the relationships must be merged. Such a merger is schematized in figure 5, where “full” containers replace the empty, placeholder containers given in figure 4. Crucially, this data merger is done in response to particular user requests for information from the database and is created in “real-time”. That is, the data is never stored in merged form but, rather, is only merged temporarily when such merger will facilitate specific tasks. The temporary nature of this process of database merger is analogous to the creation of a working format or display format for a lexicon or a set of texts.

The partition of the database into a set of nodes and a set of relationships requires Rosetta to merge the two data sets in order to achieve many basic types of desirable functionality (e.g., using a family tree to discover resources associated with related languages). The partition, therefore, puts extra burdens on Rosetta’s database tools since these do not merely need to retrieve information from the database, but they need to both retrieve and merge information. However, we believe these burdens are more than outweighed by the benefits of the system which gives our database the two crucial features of flexibility and *longevity* as summarized in (4).

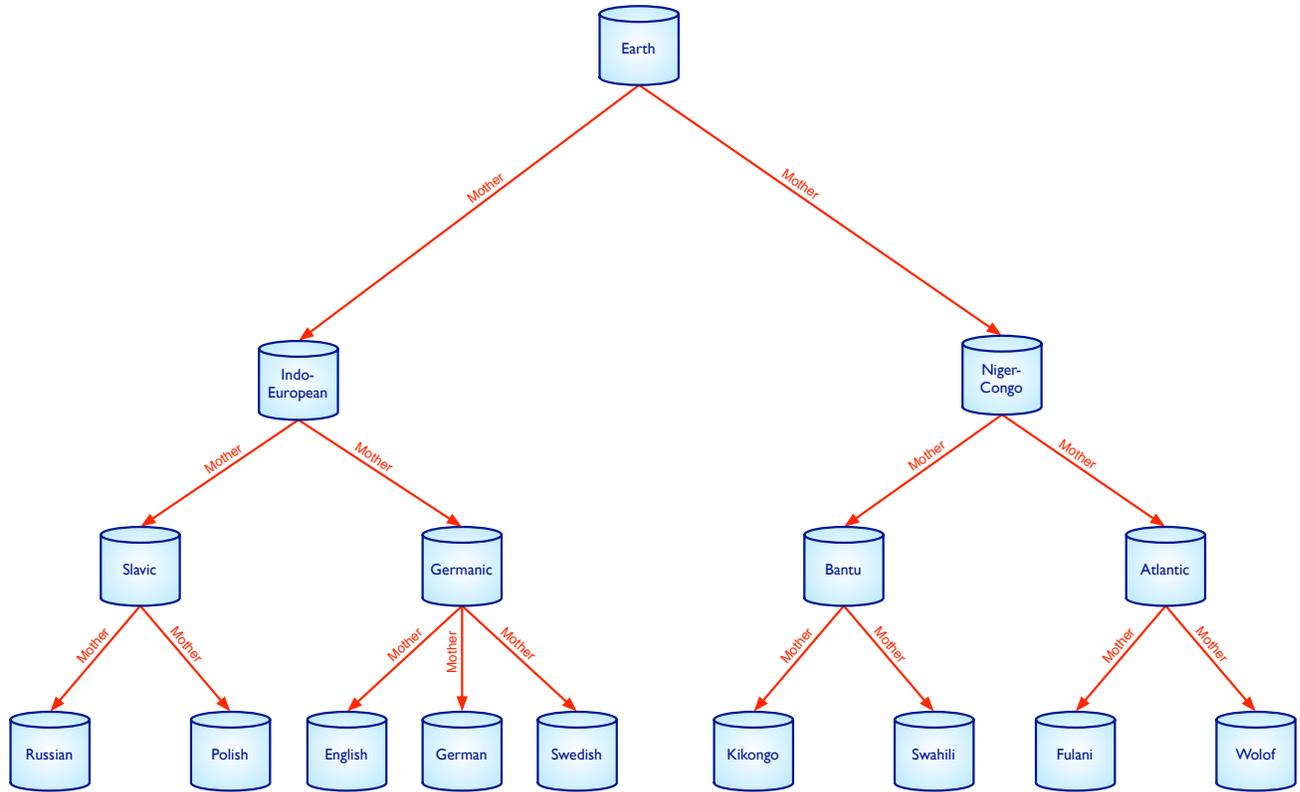


Figure 5: Merger of nodes and relationships into a unified structure

- (4) a. **Flexibility:** The database’s structure does not refer, in any crucial way, to any classificational structure for lingual nodes. Therefore, different sets of nodal relationships can be merged with the nodes, as appropriate (assuming those relationships have been coded in the database).
- b. **Longevity:** Nodal content, which is expected to be relatively stable, is not dependent on relationship content in any crucial way, thereby ensuring that changing relationship content will not adversely affect the ability of the database to encode and store nodal content.

To give a concrete example of the benefits of the flexibility of the Rosetta database structure, consider figure 6 which gives a different organization of all the language nodes seen in (3). In figure 6, the nodes are organized according to geographic relationships—specifically what continent they are found in.⁵

Geographic organization is only one other possible way of organizing lingual nodes. The database architecture, described here, could only for the use of any number of arbitrary nodal organizational structures, assuming the relevant relationships were stored in the database. One could imagine, for example, organizing nodes based on linguists who work on them. One could

⁵ Geographic entities, of course, can refer both to areas defined primarily based on linguistic criteria and areas defined based on other criteria, e.g., topographic ones. The current form of the Rosetta database, in principle, would allow for either a “linguistic” geographic classification or a geographic classification of any other desired type.

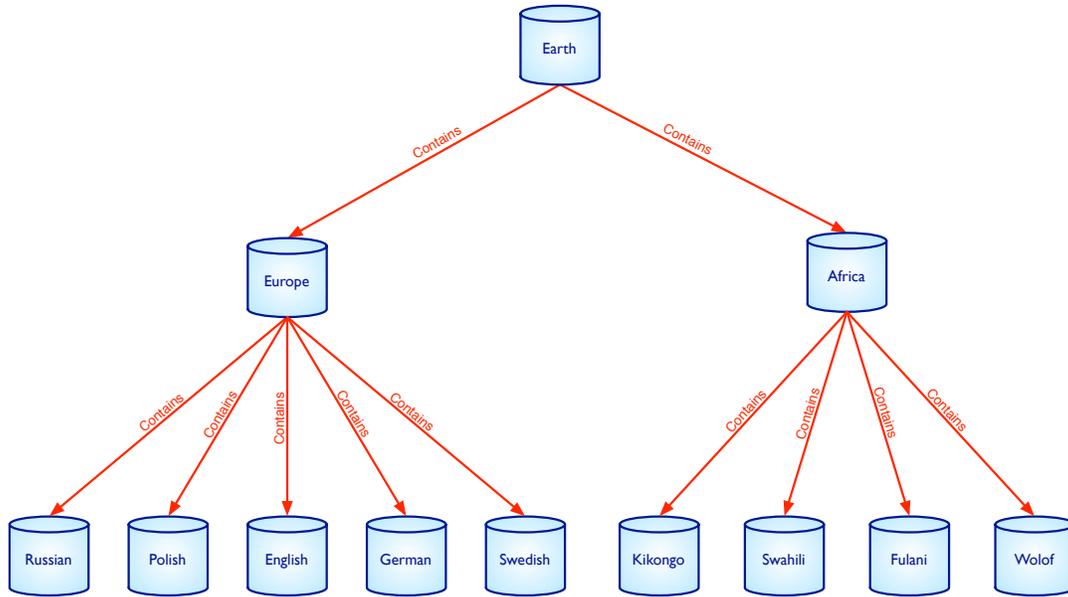


Figure 6: A geographically-based merger of nodes and relationships

also imagine overlaying two separate organizational systems on top of each other to see how they compare. Achieving such functionality could require the development of techniques and tools beyond what is currently provided, but, crucially, unlike a database which coded relationships among nodes directly into a nodal storage system, the Rosetta database itself provides no barrier to the development of such services.

We will return to the issue of database longevity in section 3.4 where we describe a straightforward pathway through the content associated with nodes and content associated with relationships can be transformed into archival formats without compromising the overall content of the system.

In the next section, we will give an overview of how the database system just described has been implemented.

3 Implementation

3.1 Introduction

In this section, we will discuss how the database system described in section 2 has been implemented. While not all aspects of the implementation will be relevant to other projects, our management of relationships among nodes, we believe may be applicable to a wide variety of problems involving the organization of linguistic data.

3.2 Nodes as objects

The Rosetta Archive and web site are built on the open-source Plone content management system⁶, which is itself built on the open-source Zope application server⁷. Zope is written in Python⁸, an

⁶ <http://plone.org>

⁷ <http://zope.org>

⁸ <http://python.org/>

open-source programming language which has been adopted by a number of other linguistic tool building projects, most notable the Annotation Graph Toolkit⁹. As a result of the fact that Rosetta is already making use of the Zope application server, we have also chosen to use the Zope Object Database (ZODB)¹⁰ for some of our nodal storage needs.

As implied by its name, a ZODB is an object-oriented database. Such a database can have largely similar functionality to a relational database. However, rather than store its data in the form of tables, it stores it in the form of objects. These objects can be assigned properties, as needed, which correspond to relational database fields. We could thus model the ZODB representation of a Rosetta lingual node as in (5).

```
(5) NODE OBJECT
    Properties
      ID: rus
    Metadata:
      Name: Russian
      Type: Lingual
      Resources: RussianText1, RussianText2, RussianGrammar1
```

The representation in (5) gives the ID and properties of a lingual node for Russian. The properties include a human-readable name for the node, an indication that this is a lingual node, and a list of references to resources associated with node.

Depending on the nature of the resources, they may be included directly in the ZODB, becoming, in effect, another property of the node. Or, if they are of a type which is impractical for storage in the ZODB for various technical reasons (e.g., large sets of scanned image files), the node object will contain information necessary for successful external access of those resources, typically in the form of a URL pointing to a Rosetta server where those materials are stored.

The ZODB's object-oriented database model, thus, gives us a natural way to implement our model of Rosetta nodes, since it, too, conceptualized them as a type of object. However, Rosetta nodes themselves are relatively simple data structures, and it would not be particularly difficult to also encode the information found in them in a relational database. The only truly crucial feature a node must have in order to interface with a database of relationships, in the present model, is a unique identifier.

Section 3.4 will discuss how the content of Rosetta nodes can be expressed in an archival format.

3.3 Relationships as RDF triples

Unlike the nodal portion of the database, the primary storage format for the relationship portion of the database is not tied to any specific platform but is, rather, expressed using the formal mechanisms of Resource Description Format (RDF)¹¹, which is one of the component technologies currently underlying the Semantic Web¹². One of the principal products of the E-MELD initiative, the GOLD ontology, is grounded in the Semantic Web, and the use of RDF in order to express

⁹ <http://agtk.sourceforge.net/>

¹⁰ <http://www.zope.org/Wikis/ZODB/>

¹¹ <http://www.w3.org/RDF/>

¹² <http://www.w3.org/2001/sw/>

information about linguistic data has already been raised in the context of E-MELD. Therefore, while it is not clear that many other projects would be able to usefully adopt our implementation for storing the content of nodes, our implementation for the storage of relationships is likely to be of wide interest within the E-MELD community.

Simons' (2005) contribution to E-MELD¹³ contains a useful introduction to RDF in the context of research on the digital expression of linguistic data. I will not give a complete description of RDF here and refer the reader to Simons (2005) for further discussion.¹⁴

The core principal of RDF is that a semantic representation of the content of a set of data can be expressed in the form of a special class of statements referred to as *triples*. A triple consists of a *subject*, a *predicate*, and an *object*. And the triple expresses that the relationship referred to by the predicate holds between the subject and the object. Triples can be expressed in a number of ways. The most intuitive is in the form of a directed graph. The simplest such possible graph, expressing a single triple, is given in figure 7. This example gives an actual triple from the Rosetta database.

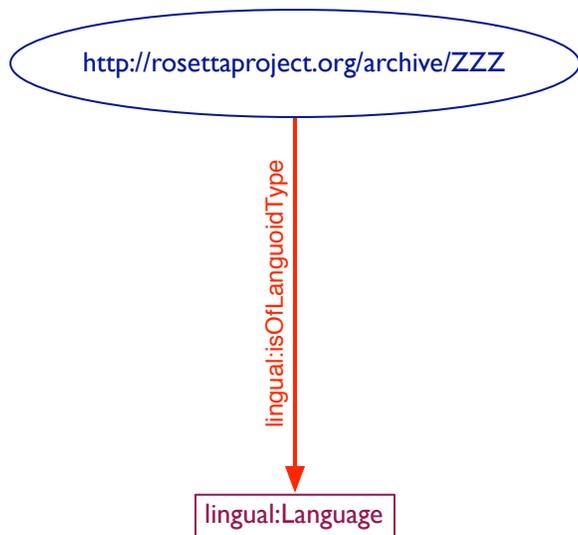


Figure 7: An RDF triple from the Rosetta relationship database

The graph in figure 7 expresses that an entity with URI <http://rosettaproject.org/archive/ZZZ> is a language. The entity referred to is the language which has Ethnologue 14 code ZZZ (Dimli, an Iranian language).¹⁵ This URI serves as the unique ID which allows nodal content and relationships to be merged along the lines of what is discussed in section 2.4. The predicate in the triple is represented as an arrow pointing from the subject to the object. The prefix “lingual:” marking the predicate and the object refers to the XML namespace¹⁶ in which the terms “isOfLanguoid-

¹³ <http://emeld.org/workshop/2005/papers/simons-paper.pdf>

¹⁴ For a non-linguistic, technical introduction to RDF, see also: <http://www.w3.org/TR/rdf-primer/>.

¹⁵ While Rosetta uses Ethnologue 14 codes in many of the URI's of its archives. It should not be assumed that Rosetta will maintain any isomorphism between codes indicated in its URI's and any Ethnologue coding scheme. Language codes associated with lingual entities are coded by the use of RDF triples explicitly linking a code to the triple as in figure 8.

¹⁶ This is namespace maintained by Rosetta. The URI for the namespace is: <http://rosettaproject.org/lingual/>.

Type” and “Language” are defined.¹⁷ The term *linguoid* is a cover term employed by the Rosetta RDF system for a dialect, language, family, or other kind of linguistic grouping. This triple, therefore, expresses the (rather uninteresting) claim that in English would be rendered as “Dimli is a language”.¹⁸

Much of the power of RDF comes from the fact that triples can be joined to form larger networks of triples. Figure 8 exemplifies by adding a number of other triples to the graph in (7).

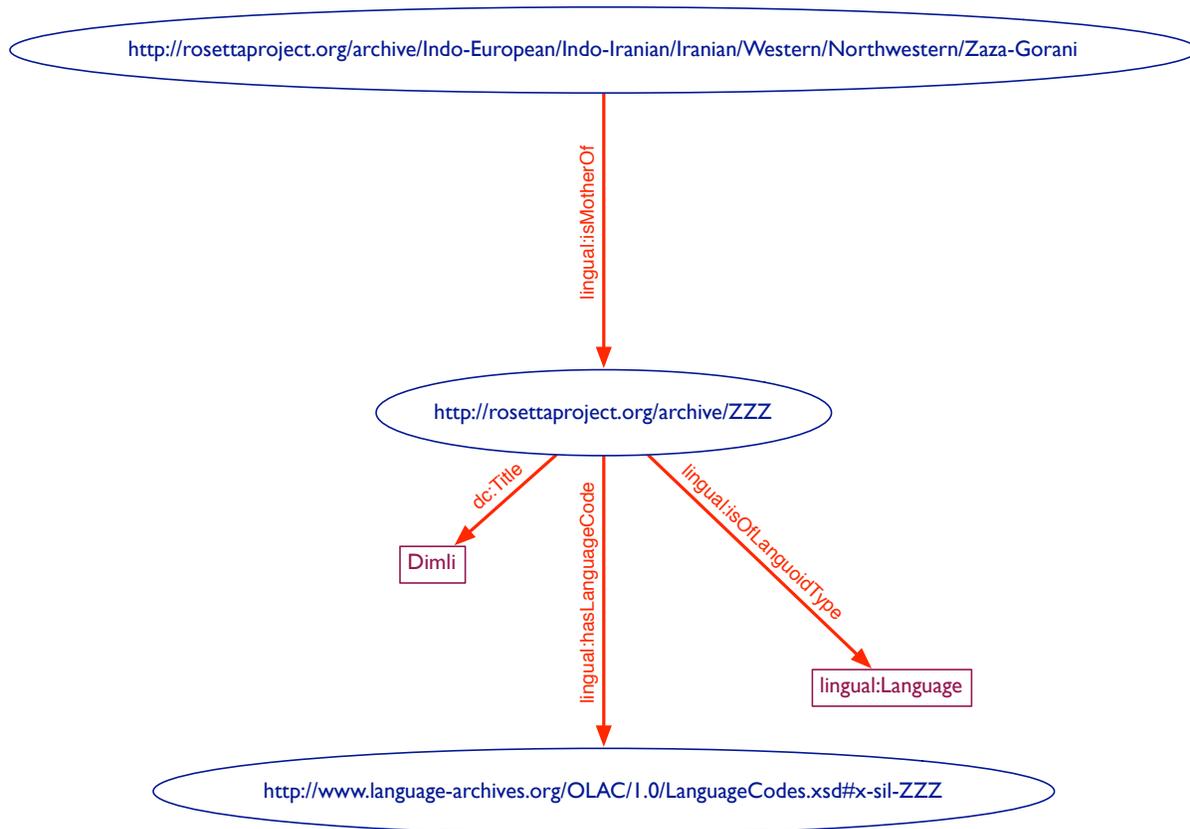


Figure 8: A small network of RDF triples from the Rosetta relationship database

The set of triples in figure 8 can be expressed in English as follows: A lingual node in the Rosetta database which is commonly known as Dimli is a language with OLAC language code x-sil-ZZZ, and it is the daughter of the Zaza-Gorani language group. Note that one of the predicates for the triples in figure 8 does not make use of the lingual namespace but, rather, makes use of the “dc” (i.e., Dublin Core¹⁹) namespace. (Rosetta uses the Dublin Core Title element as a predicate for expressing a relationship between a lingual node and a human-readable name for the node.)

As a standard of the Worldwide Web Consortium (W3C)²⁰, RDF is associated with a specification for how to express RDF triples in XML, along with other less verbose machine-readable

¹⁷ For an introduction to XML namespaces, see: <http://emeld.org/school/classroom/xml/howto.html>.

¹⁸ Following W3C convention, URI’s in the RDF graphs in this paper will be represented by ovals and string literals by rectangles.

¹⁹ <http://dublincore.org/>

²⁰ <http://www.w3.org/>

formats.²¹ The primary format for Rosetta's database of relationships is this RDF XML format. At present, the database contains statements of the relationships holding among languages and language families given in Ethnologue 14 and statements of relationships between language areas and countries and continents.

As XML is a poor format for fast processing, the Rosetta web site makes use of a backend database system for handling user requests. At present, the backend is a ZODB database, of the sort described in section 3.2. However, the relationships are stored separately from the nodes themselves and another database system (e.g., a SQL database) could also serve as the high-speed backend with minimal changes to the site. Interaction with the RDF graph is mediated by Python library, custom built for the Rosetta database on top of a general RDF library written in Python.²²

While the driving force behind the use of RDF in the Rosetta site was to construct a database model where unstable and contested relationships could be expressed independently from more stable nodal content, we have found RDF to also be a useful mechanism to express other kinds of content on the site. For example, an important class of content on the site are references to books and articles on various languages. Given that the Zope application server incorporates Dublin Core directly into its metadata architecture and Dublin Core also has an RDF specification²³, we found Dublin Core RDF, augmented with RDF specifications of the OLAC²⁴ extensions to Dublin Core, to be a sensible choice for the storage of references in the database. We probably would not have taken this route if we had not otherwise needed to develop methods for integrating RDF with Zope/Plone for other reasons. However, once those mechanisms were in place, we found RDF databases could be used for a range of purposes. In the near future, as we develop user and community tools on the Rosetta site, we plan to use friend-of-a-friend RDF²⁵ for expressing information about individuals and how they relate to each other.

The RDF model for the Rosetta database is described in two RDF Schemas, which are still partially under development. One of these schemas, the lingual schema, specifies properties of languoids and relationships among languoids and between languoids and geographic entities.²⁶ The other schema, the lingbib schema, which is much simpler, is used for references to language resources (books, articles, etc.). It extends the Dublin Core RDF Schema with the OLAC's Dublin Core metadata extensions.²⁷ In addition, we are experimenting with linking the geographic categories specified in the lingual schema to those found in the SWEET Earth Realm ontology²⁸.

3.4 Comments on archiving and interoperability

Having described the implementation of the Rosetta database, it would be useful to comment briefly here on issues relating to archiving and interoperability.

All portions of the database stored primarily in RDF already make use of an archive-ready, interoperable format—this is a tremendously valuable feature of RDF databases. As discussed above, for reasons relating to site performance, the Rosetta web site uses a ZODB working format

²¹ A specification for RDF XML can be found at: <http://www.w3.org/TR/rdf-syntax-grammar/>.

²² This library can be found at <http://rdflib.net/>.

²³ <http://www.dublincore.org/documents/dcmes-xml/>

²⁴ <http://www.dublincore.org/documents/dcmes-xml/>

²⁵ <http://www.foaf-project.org/>

²⁶ The current version can be found at: <http://emeld.org/workshop/2006/papers/goodparker-rdf/lingual.rdfs>.

²⁷ The current version can be found at: emeld.org/workshop/2006/papers/goodparker-rdf/lingbib.rdfs.

²⁸ <http://sweet.jpl.nasa.gov/ontology/earthrealm.owl>

of the relationships database. RDF triples can be added and removed to this working format. However, this working format still represents the content of the database as RDF, even though it doesn't use RDF XML. The process of going from the ZODB working format to RDF XML is trivial since such functionality is built-in the Python RDF library we use. Therefore, the route from working format to archival format presents no noteworthy problems.

This leaves open the issue of how the contents of Rosetta nodes can be archived and made interoperable. Some of the Rosetta nodal content consists of images and audio files. Such resources present the same archival and interoperability issues as they would in any digital archive. Since these are not problems specific to the Rosetta database, I set them aside here. However, the other content found on nodes, in particular the metadata associated with them, does present database-specific archival problems. Fortunately, the RDF architecture we have built to maintain the RDF databases on the site can also be applied to this problem. While the primary form of the nodal content database is not stored in RDF, the RDF tools used by Rosetta can straightforwardly translate nodal objects in the ZODB into RDF, once a human user develops a translation from the object's properties and to an RDF representation of those properties. Thus, nodal data can be straightforwardly transformed into an archival and interoperable format, not presenting any particular problems with respect to longevity and interoperability.

3.5 Outstanding issues

There are two outstanding issues with respect to the RDF model for relationships described here, which we have not yet implemented a solutions for—because this has not been necessary—but which we expect to also be straightforwardly handled within an RDF database once this becomes necessary. We describe each in turn.

The first issue centers around the fact that, for many reasons, it may often be desirable to not simply encode a triple in the RDF database but to also annotate the triple itself. For example, we might want to indicate who was responsible for the creation of the triple. More substantively, we might want to say what the justification is for putting a triple in the database in the first place. Again, borrowing from the realm of genealogical relationships, it might be useful to say what the evidence is for a particular genealogical grouping. This situation is schematized in figure 9. Where the “justification” for claiming a genealogical relationship between Latin and several Romance languages is schematized as being the content of resources whose subject is the relationship between those languages and Latin.

Annotating triples for information about their justification, as schematized in figure 9 could be extraordinarily useful for research since, in principle, it could allow a user to easily explore competing hypotheses with respect to some data set. In the context of Rosetta, it would allow, for example, us to include contradictory proposals regarding the genealogical classification within the same database, annotating each proposed relationship for what sort of evidence was used to justify it and where that evidence can be located. Users could then choose to extract proposed relationships based on different kinds of evidence or of different provenance in order to compare them—and, perhaps, ultimately, to propose worthwhile refinements to existing claims.

The data in the Rosetta relationship database has been, to this point, limited enough in nature to allow us to skirt the issue of annotating triples. However, fortunately, when we reach a point when it will be important to annotate triples, we will be able to exploit an existing RDF mechanism designed explicitly for this kind of problem. This mechanism is referred to as *reification*, and it

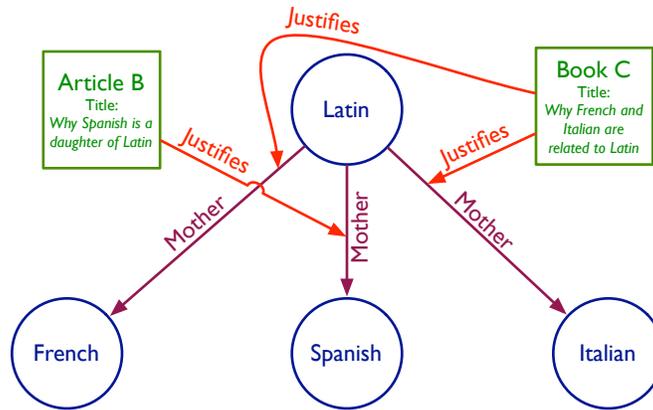


Figure 9: Annotating a triple

effectively allows triples to be treated as subjects or objects in the graph which can, then, be components of other triples. Abstracting from certain technical details, reification can be schematized as in figure 10. The example in 10 is based on “Latin is the mother of Spanish” triple schematized in figure 9.

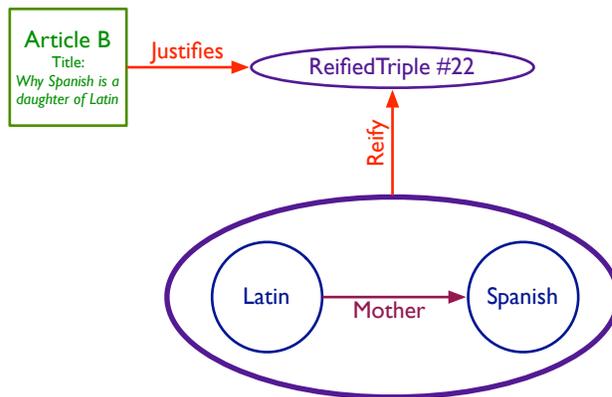


Figure 10: Reification of a triple

As schematized in figure 10, reification allows a triple to be treated as an entity in the graph, as opposed to a statement. This entity can then be annotated by being either the subject or object of another statement.

The second outstanding problem in the Rosetta database, which was alluded to above in section 2.2 is dealing with the issue of whether or not two linguistic entities generally thought of as the “same” are really the same. For example, is Greenberg’s Niger-Congo the same thing as the contemporary notion of Niger-Congo? Conventional wisdom of the classification of languages into Niger-Congo has changed considerable since Greenberg’s proposal. However, most linguists understand contemporary Niger-Congo to be essentially the same as Greenberg’s Niger-Congo, but simply revised in ways which are not substantial enough to treat the two as distinct entities.

What concerns us here is not the problem of whether or not Niger-Congo—or any comparable “entity”—should be treated as one node or two nodes in the database. We take this to be an issue

that can only be resolved by debate among linguists, and it is not the responsibility of the database to make a determination either way. However, we *would* like a mechanism which would allow any reasonable analysis of such cases to be encoded, including the possibility that it might be desirable to encode competing such analyses in the database. In the case of Niger-Congo, this could mean including *three* nodes in the database: One for Greenberg’s Niger-Congo, one for contemporary Niger-Congo, and one for “general” Niger-Congo where Greenberg’s Niger-Congo and contemporary Niger-Congo are assumed to be one and the same entity

The structure of the Rosetta database, as described here, of course, would allow us to enter three such nodes into the database. The problem is we want to make sure that, when this might happen, the database also contains information saying that these nodes could, under certain circumstances, could sensibly be understood as all referring to a single entity. That is, we want to be able encode a situation like the one in figure 11, where three distinct nodes in the database are considered to be the same from the point of view of the (public) Rosetta web site.

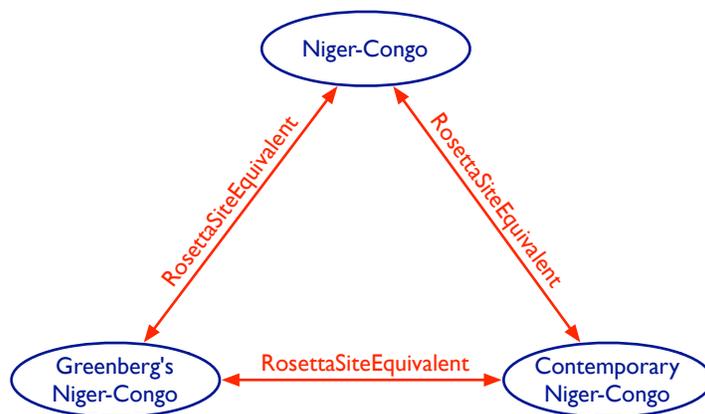


Figure 11: Equivalence among distinct nodes

Figure 11 schematizes the three Niger-Congo nodes as being equivalent in one particular context (the Rosetta Site). Thus, elsewhere, they could still be considered distinct. It should be clear that the sort of relationships schematized in figure 11 could be expressed via RDF triples, in the same manner as, for example, genealogical relationships can be. Therefore, while we have not yet had to implement a solution for this problem of not being certain when two invocations of ostensible the same linguistic entity really do refer to the “same” thing, we believe that such a case could be encoded in the RDF database by making use of predicates specifically devised to encode different kinds of equivalence among nodes which are neither clearly the same or clearly distinct.²⁹

Of course, implementing such multiple-node structures in this way will adds a layer of complexity to the process of retrieving all relevant relationships from the RDF database. Nevertheless, for certain kinds of data, this complexity may be a small price to pay for being to able to accurately the encode the lack of real-world consensus on whether or not what appears to be a single linguistic entity really is a single linguistic entity.

²⁹ The inspiration for dealing with this problem through a mechanism of contextual equivalence comes from the use of *canonical equivalence* and *contextual equivalence* within the Unicode standard.

4 Conclusion

In this paper, we have outlined a database structure which we believe is well-suited to distinguishing between stable, non-contested content and unstable, contested content in ways which ensure that non-contested content is not encoded in ways which make it unduly dependent on contested content while still allowing non-contested content and contested to be merged together where desirable. The conceptual model for this structure was to view non-contested content as encoded in stable nodes, on which very little organizational structure is imposed, and to view contested content as encoded as relationships holding among nodes. This conceptual model has been implemented by using an object-oriented database for most nodal content and an RDF database for encoding relationships. However, once RDF tools are put into place, we found it can make sense for some stable content to also be placed within an RDF database because of the ease with which RDF content can be put into archival forms and made interoperable.

The problem space that was the focus of this paper—genealogical relationships among languages—is not of direct concern to most documentary and descriptive projects. However, we believe that the model, and some aspects of the implementation, proposed here may be applicable to a large number of problems involving contested content. Consider, for example, the annotation of primary text data for grammatical information. A sequence from the text could be conceptualized as a database node—since it is stable and non-contested. Furthermore, a large number of labels for grammatical categories exist, whose role in annotation is also not contested.³⁰ These could also be conceived of as database nodes. However, the association of a particular sequence from a text to a grammatical label is likely to be contested (perhaps even within the mind of a single annotator). Such associations, therefore, could be stored as relationships between text sequences and grammatical labels. Thus, the unstable analysis of a text could be separated from the stable text and a stable set of category labels.

Of course, much of what we are discussing in this scenario is central to the vision on which the GOLD Ontology community³¹ is based. Here, we are not offering a whole new set of ideals to aim for. Rather, we have outlined a concrete, implemented model on which, ultimately, could play a role in attaining the ultimate aims of initiatives like E-MELD.

³⁰ Of course, this is not to say other aspects of those categories are not contested.

³¹ <http://www.linguistics-ontology.org/>