Automatically Linking Dictionaries of Gallo-Romance Languages Using Etymological Information

Pascale Renders[1], Gérard Dethier[2], Esther Baiwir[3]

[1]FNRS/University of Liège
[2]University of Liège
[3]FNRS/University of Liège
pascale.renders@ulg.ac.be, g.dethier@alumni.ulg.ac.be, ebaiwir@ulg.ac.be

Abstract

How could we link together digital dictionaries which have no common lexical units, but deal with the same linguistic area? And how could we do that automatically, in order to ensure that all future updates of these dictionaries are taken into account in the linking process?

This contribution exposes the solutions that we propose in the field of French and Gallo-Romance historical lexicography. The digitalisation currently in progress of a work of scientific reference, i.e. the Französisches Etymologisches Wörterbuch (FEW), gives us a mean to link together other dictionaries, such as the Dictionnaire Etymologique de l'Ancien Français (DEAF), the Dictionnaire du Moyen Français (DMF), the Anglo-Norman Dictionary (AND), or the Atlas Linguistique de la Wallonie (ALW), through the use of the references of these dictionaries to the FEW. Concrete examples of linking lexical data are discussed in this context.

We also describe a simple peer-to-peer protocol allowing e-dictionaries to be automatically linked in a distributed way using the references of their articles. An implementation based on a simple REST API is suggested to let teams maintaining different e-dictionaries keep their own technologies and data schema.

Keywords: Linked lexical data; Gallo-Romance lexicography; FEW; Exploitation of language resources

1. Introduction

Etymology is an information that is not systematically available in all dictionaries. However, it might be used to link together digital dictionaries which have no common lexical units, but deal with the same linguistic area. In the field of French and Gallo-Romance lexicography, the digitalisation currently in progress of a reference dictionary, the *Französisches Etymologisches Wörterbuch* (FEW), gives us the opportunity to automatically link dictionaries such as the Dictionnaire Etymologique de l'Ancien Français (DEAF), the Dictionnaire du Moyen Français (DMF), the Anglo-Norman Dictionary (AND) or the Atlas Linguistique de la Wallonie (ALW).

The questions that will be addressed are (1) how can we link these resources and what is to be linked exactly; (2) how can this be done automatically? This contribution gives some examples of lexical units that could be linked in French and Gallo-Romance lexicography, exposes the linking process we imagine in theory and explains the way in which this could be implemented in practice.

2. A Case Study: Gallo-Romance Lexicography

The FEW has the particularity to gather lexical units of French, Gascon, Occitan, Francoprovençal and their dialects, according to their common ancestry (etymon). Each FEW article provides, under an etymon lemma, the history of one lexical family. Lexical units whose etymology is not known are gathered in the volumes 21–23, with an onomasiologic classification.

As a thesaurus and a reference for the etymology of all lexical units in the area under consideration, the FEW works as a "lieu de synthèse" in this linguistic area, see (Buchi and Renders, 2013). Consequently, the FEW is systematically cited in many historical dictionaries of these languages and dialects. This provides a wonderful opportunity to link dictionaries together by putting the FEW at the center of a lexicographic network, through the use of etymological information.

The linking process has another purpose. The dictionaries mentioned above not only mention, but regularly update, the FEW, for instance by providing a new etymology to FEW units from volumes 21–23. Unfortunately, providing an updated version of the FEW integrating these contributions is not possible in practice, because of the complex structures of the FEW. Linking the FEW with all the lexicographic resources available would provide users and lexicographers with a facilitated and easy access to these updates. In this context, it is necessary to implement an automatic linking process, in order to ensure that all future updates of these dictionaries are actually taken into account.

Gallo-Romance dictionaries that could be involved are, for example, the DEAF, the AND, the ALW, the TLF, and all the resources provided by the ATILF (TLF-Etym etc.). Some of the historical or etymological dictionaries of Romance languages, such as the DERom, could also be added to this network. These dictionaries mention for each lexical unit a "FEW reference" i.e. the exact location in the dictionary (volume, page and column) where this lexical unit can be found. For example, ALW 17 provides a new etymology for 21 lexical units that are described as from "uncertain origin" in FEW. For each of them, the ALW mentions the exact location where it appears in the FEW and provides the new location where it should be moved according to its new etymology. The wallonish verbs "zam'ter", "cham'ter" were, for instance, marked "from uncertain origin" in the FEW and therefore put in the volume 21 (FEW 21, 342a). However, ALW 17, 206a defines "examen" (FEW 3, 258a) as their common etymon. Updating the FEW means that these lexical units should be moved from FEW 21, 342a to FEW 3, 258a under the "examen" lemma. The same applies for the wallonish term fournakeye (f.) "ribambelle" (ALW 17, 73a and 75a), which should be added to FEW 3, 907b.

3. Linking E-Dictionaries

This section describes an automated method of linking e-dictionaries. The method is first described from a theoretical point of view. Then a suggestion of implementation is proposed.

3.1 Definitions

From a Computer Science point of view, a dictionary is a set of entries (k, v) where k is a key and v a value. An additional property that is commonly accepted is the unicity of the keys in a given dictionary i.e. in the set of all entries, it is not possible to find two entries (k_1, v_1) and (k_2, v_2) with $k_1 = k_2$.

Let v_1 be the article of a dictionary d_1 having the key k_1 and v_2 be the article of another dictionary d_2 having the key k_2. If the article v_2 references the article v_1, the reference can be represented by the tuple (d_2, k_2, d_1, k_1). A reference can also be noted $v_2 \rightarrow v_1$ or $(d_2, k_2) \rightarrow (d_1, k_1)$.

Although the above definition is straightforward, the keys and articles for a particular dictionary are not always easily defined. In the case of the FEW, the FEW reference can be used as the key (e.g. FEW 3, 258a). As previously stated, the FEW reference is a location (the column of a particular page in a given volume). In some cases (when several articles have the same location), the location has still to be augmented with the etymon to uniquely identify one article. This is also true for ALW references which also represent locations where a particular notice can be found.

Let D be the set of all the dictionaries complying to the rules described above (set of entries with unique keys), K_i the set of all the keys of a dictionary d_i and R the set of all references $(d_i, k_{i,j}, d_k, k_{l,m})$ where $d_i, d_k \in D$, $k_{i,j} \in K_i$ and $k_{l,m} \in K_l$. In a perfect world, when reading the article v of dictionary d_i with key $k_{i,j}$, we would have access to all references $(d_j, k_{j,l}, d_i, k_{i,j})$ with $d_j \in D$ and $k_{j,l} \in K_j$ and therefore all the information available on the article: its content but also links to other articles (and their content) referencing it. If one of these articles suggests an update, the reader would be aware of it and always have access to the latest "version" of an article.

The above model can be applied to the task of linking dictionaries of Gallo-Romance languages exposed in the previous section. Indeed, if the FEW, the ALW, etc. can be considered as part of D, then we can model references between articles of these dictionaries using above framework. For instance, let d_{FEW} be the FEW and d_{ALW} be the ALW. The example of update of the FEW by the ALW from previous section actually implies two distinct references:

1. ALW 17, 206a $\rightarrow$ FEW 21, 342a (removing the lexical unit)
2. ALW 17, 206a $\rightarrow$ FEW 3, 258a (adding the lexical unit)

with ALW 17, 206a $\in K_{ALW}$, FEW 21, 342a $\in K_{FEW}$ and FEW 3, 258a $\in K_{FEW}$.

We can define an e-dictionary as a system able to provide the content of an article v given its key k. We suppose that an e-dictionary represents a single dictionary of D. In the following, we will note e_i the e-dictionary system hosting a dictionary $d_i \in D$. In order to link several e-dictionaries together, we only need a way to implement R. In this case, someone reading an article through an e-dictionary would have access to the content of the article and to all the articles referencing it or referenced by it only by querying the e-dictionary and the system hosting R.

Implementing that kind of system is not trivial. The most obvious solution is a centralised platform maintained by an independent organisation. However, building this kind of organisation and platform is neither simple nor efficient: it requires substantial funding in order to maintain R, a huge set that continuously evolves. Also, it is not scalable nor secure from a technical point of view as it represents both a potential bottleneck and a single point of failure.

An alternative is to let the e-dictionaries build a distributed representation of R in a collaborative way. Indeed, each e-dictionary does not need to be aware of the whole R set. Let $R_{i,j}$ be the subset of R containing all references implying keys from either $d_i \in D$ or $d_j \in D$. An e-dictionary representing d_i only needs to be aware of $S_i = \bigcup_{j \in E} R_{i,j}$ where E is the set of dictionaries to which d_i refers (i.e. the dictionaries to which d_i's articles refer).

Next section describes the protocol that enables e-dictionaries to build $R_{i,j}$ in a collaborative way. Some technological choices are also suggested to build a practical solution.

3.2 The Linking Protocol

In this section, we will describe a simple protocol allowing e-dictionary systems to build their S_i set in a distributed way. Concrete technologies are suggested to actually implement the protocol.

3.2.1. Theory

Let d_i be a dictionary represented by an e-dictionary system e_i. In order to build S_i, e_i will send and receive messages representing the creation of references. When a reference from article v with key k_v of d_i is made to an article w with key k_w of d_j, e_i sends a message notifying e_j of the new reference (d_i, k_v, d_j, k_w) being created, in addition to storing the new reference in its own representation of $R_{i,j}$ (and therefore S_i). When e_j receives the message sent by e_i, it updates its representation of $R_{i,j}$ (and therefore S_j). When $R_{i,j}$'s representation is updated on both e_i and e_j, both e-dictionaries are aware of the reference being made from article v to w and are therefore able to expose this reference to their users.

With this protocol, creating a reference in a e-dictionary enriches automatically the set of references in all other relevant e-dictionaries. This incremental approach also allows the continuous improvement of the existing set of references with a minimum effort as the maintenance of the global references set is automated.

It is to be noted that letting e-dictionaries build their set of references actually leads to the emergence of a network of e-dictionaries connected by their references.

The protocol described here implies a peer-to-peer architecture where e-dictionaries are the peers. This is good news as peer-to-peer architectures are well known for their good scalability and ro-

bustness. We did not address the security and robustness problems that may arise. Although these must be tackled in a real world implementation, they are beyond the scope of this paper.

3.2.2. Implementation

As already stated, most e-dictionaries are developed by different teams from different organisations. The technologies used by these teams to actually implement the e-dictionaries might strongly differ (PHP, Java, Node.js, etc.). Our suggestion is for all these e-dictionaries have their own internal representation and technology stack, but for them to expose a common yet minimal API allowing the exchange of messages as exposed at the beginning of this section. In this way, the coupling between different projects and teams is minimised and allows more flexibility, robustness and scalability from the technical point of view, as well as from the point of view of project management.

A modern approach is to implement the API using web-oriented technologies, and our suggestion would be to implement a simple REST API based on HTTP request and using JSON-encoded data[1]. The advantage of this approach is that this kind of interface can be implemented using a wide range of technologies, thus imposing almost no constraints to the teams developing the different e-dictionaries.

Each e-dictionary must be hosted under a different hostname which can therefore be used to uniquely identify the e-dictionary system itself. Let `my-edict.org` be the hostname of e-dictionary `my-edict`. Below REST resources should be exposed in order to let the e-dictionary receive messages coming from external systems and let other e-dictionaries access the content of hosted articles.

In the following, we will use cURL[2] syntax to express HTTP requests in a formal and precise way. Each section starts with a summary of the HTTP request composed of the HTTP method (GET, POST, etc.) and the URL pattern (parameters are prefixed with a colon) e.g. `GET http://www.google.com/:service/` where `service` is a parameter.

Creating References

`POST http://my-edict.org/api/reference`

Posting (i.e. doing an HTTP POST request with) the following data to this resource should lead to the addition of a reference in `my-edict`:

```
{
    "source_dict": "http://other-edict.org",
    "source_id": "a-key-in-other-edict",
```

[1] JavaScript Object Notation, see http://json.org/ for a full specification of this data-interchange format.
[2] Curl is a command line tool and library for transferring data with URL syntax, see http://curl.haxx.se for more details.

```
    "dest_dict": "http://my-edict.org",
    "dest_id": "a-key-in-my-edict",
}
```

The following cURL command (or some equivalent implementation) should be executed by e-dictionary `other-edict` when represented reference is actually created:

```
curl "http://my-edict.org/api/reference" -X POST -H "Content-Type:
application/json" -d @data.json
```

where `data.json` is a file containing above data.

On reception of this kind of message, `my-edict` should ensure that:

1. `dest_dict` does contain the identifier of `my-edict`,
2. `dest_id` is the identifier of an existing article hosted by `my-edict`.

If above conditions are true, the reference can be inserted in `my-edict`'s database. In this way, when a user wants to read the article of `my-edict` identified by `a-key-in-my-edict`, `my-edict` will be able to expose the incoming reference from article of `other-edict` identified by `a-key-in-other-edict`.

Accessing Articles

```
GET http://my-edict.org/api/articles/:article-id
```

Getting (i.e. doing an HTTP GET request on) this resource should return the following data describing the article identified by `:article-id` (which is a placeholder for a real ID) in `my-edict`, for instance:

```
{
    "article-id": "a-key-in-my-edict",
    "url": "http://my-edict.org/a-key-in-my-edict"
}
```

where `article-id` is the unique identifier of the article in `my-edict` and `url` is the URL at which the article can be accessed. It is to be noted that the URL scheme used to let users access articles is totally up to the team implementing the e-dictionary.

The following cURL command (or some equivalent implementation) should be executed when accessing an article:

```
curl "http://my-edict.org/api/articles/a-key-in-my-edict"
```

Listing References of an Article

```
GET http://my-edict.org/api/articles/:article-id/references
```

Getting (i.e. doing an HTTP GET request on) this resource should return the list of references associated to the article identified by `article-id` (which is a placeholder for a real ID) in `my-edict`, for instance:

```
[
    {
        "source_dict": "http://other-edict.org",
        "source_article_id": "a-key-in-other-edict",
        "dest_dict": "http://my-edict.org",
        "dest_id": "a-key-in-my-edict",
    },
    {
        "source_dict": "http://my-edict.org",
        "source_article_id": "a-key-in-my-edict",
        "dest_dict": "http://other-edict2.org",
        "dest_id": "a-key-in-other-edict2",
    }
]
```

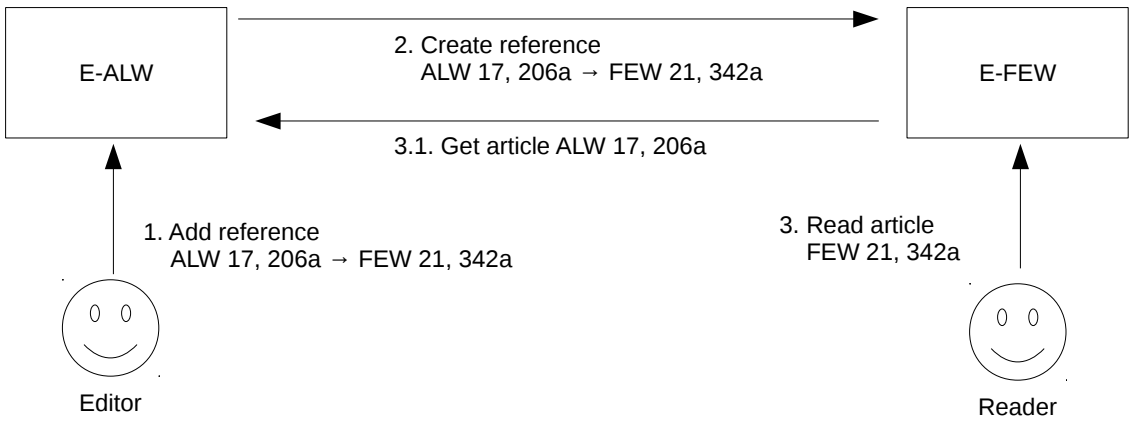
The references of an article include both incoming and outgoing references.

The following cURL command (or some equivalent implementation) should be executed when accessing the list of references of an article:

```
curl "http://my-edict.org/api/articles/a-key-in-my-edict/references"
```

3.2.3. Example

The following figure illustrates the interactions between users and e-dictionaries and the requests these interactions imply. The scenario described here uses the example given in section 3.1: an editor creates the reference ALW 17, 206a → FEW 21, 342a and, after that, a reader of the FEW displays the article FEW 21, 342a and has access at the same time to the update made by the article ALW 17, 206a.



1. An editor of the ALW adds the reference ALW 17, 206a → FEW 21, 342a by a means that is dependent on the way the e-ALW is implemented e.g. using a web interface.
2. The e-ALW notifies the e-FEW that a new reference has been created using the request described in section "Creating References".
3. A reader of the FEW accesses the article FEW 21, 342a and, in a transparent way, the e-FEW builds a consolidated view of the article by retrieving also the article ALW 17, 206a (step 3.1) using the request described in section "Accessing Articles".

The request described in section "Listing References of an Article" is not used in above scenario. However, it might make sense in more elaborated scenarios where a user wants to explore a graph of references that might span several e-dictionnaries.

4. Conclusion

This paper discussed the question of linking together digital dictionaries which deal with the same linguistic area, some of these dictionaries giving additional or updated information about lexical units from other dictionaries. The update of the FEW through the references made by the ALW is given as a case study and highlights the need for linking.

We exposed a simple peer-to-peer protocol allowing several e-dictionaries to connect and maintain together the set of references involving the articles they host without the need for a central organisation or system, preventing a potential bottleneck and a single point of failure. We also suggested an implementation of this protocol implying a small REST API that should be exposed by all e-dictionaries willing to be connected. This approach allows the teams responsible for the maintenance of the various e-dictionaries to keep their own technologies and representation for their data.

The described protocol allows us to link lexical units on the basis of any criteria. In the particular case of Gallo-Romance lexicography, the etymological information and the systematic mention of the FEW allow a quick linking process. At the same time, this linking process enables the update of the FEW by giving direct access to updates made by other dictionaries.

5. References

Buchi, E. & Renders, P. (2013). 41. Gallo-Romance I: Historical and etymological lexicography. In Gouws, R. H., Heid, U., Schweickard, W. & Wiegand, H. E., editors, *Dictionaries. An International Encyclopedia of Lexicography. Supplementary Volume: Recent Developments with Focus on Electronic and Computational Lexicography*, Handbooks of Linguistics and Communication Science (HSK) 5/4, De Gruyter Mouton, Berlin, pp. 653–662.