Towards an XML Representation of Proper Names and their Relationships

Béatrice BOUCHOU
Mickaël TRAN
Denis MAUREL

Université François-Rabelais de Tours
Laboratoire d'Informatique

I present our work about a multilingual lexical resource on proper names, which we want to use now, with XML.

Plan

- Prolex Base: a large multilingual lexical resource on proper names and their relationships
  - Entity / Relationship model
- Schema for XML documents containing Prolex data
  - Design method
  - Current result
- Conclusion and future work

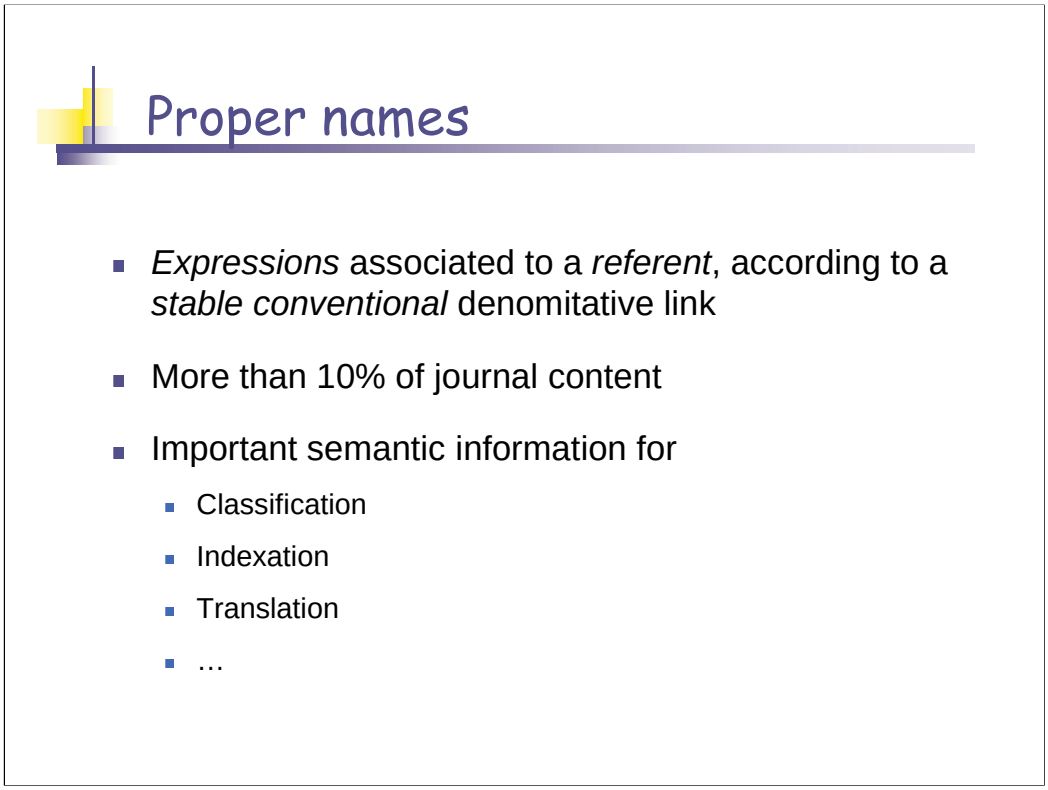
This lexical resource has the particularity to contain not only a set of proper names, but also most of relationships that exist between them.

It leads to a rather complex system, which has been represented using the E/R formalism: I'll begin with an overview of this model.

The need for an XML representation has appeared to import and export data from our relational database. Precisely, we want to be able to export any part of the database in an XML format.

To this aim we have designed an XML schema starting from the conceptual model of the database.

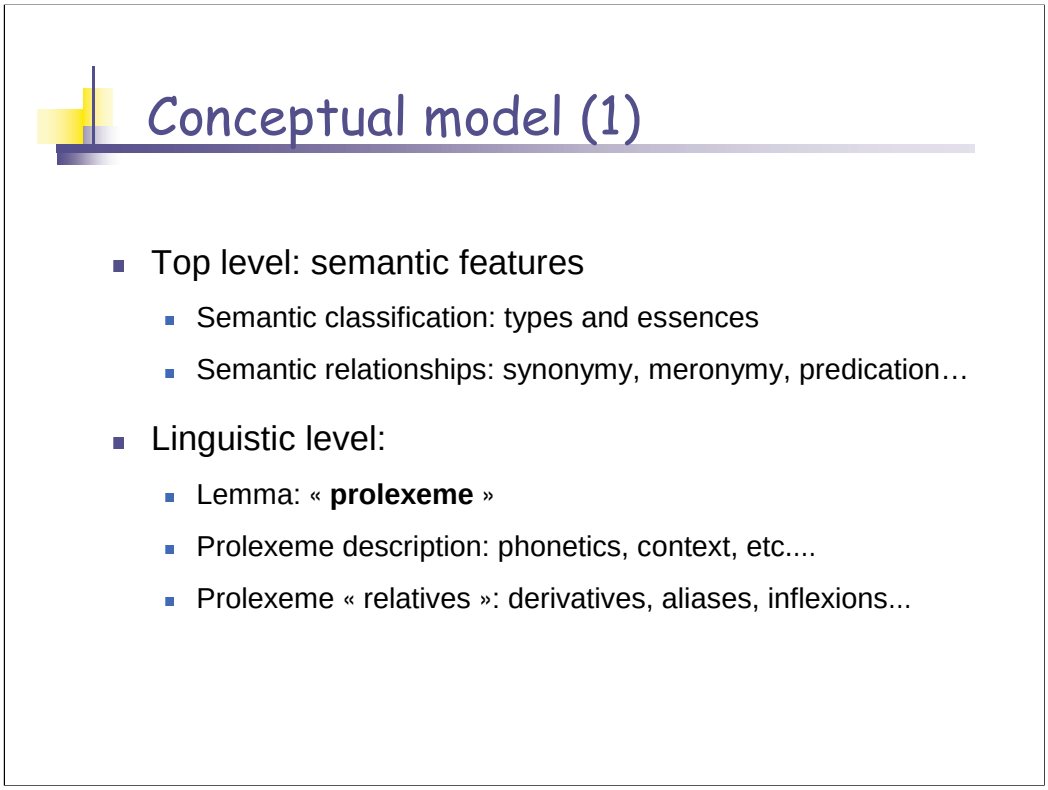
I will present the design process and the resulting schema, then I will conclude and say a word about future work.



First I recall a simple definition of proper names: …

Their interest in natural language processing tasks is important, because they can represent up to 10% of the whole content of a journal.

Information about proper names can be usefull for text processing tasks, such as classification, indexation,translation, etc.



As I was saying, we have built a conceptual model of proper names and their relationships, which is structured in two parts:

The most abstract level, that we call the « top level », contains semantic features of proper names and their relationships.

Semantic classification using types and supertypes is described, together with semantic relationships between proper names such as synonymy , meronymy etc.

Under the top level is the linguistic level, in which the kernel is the proper name's lemma that we call « prolexeme ».

Around this kernel there is a set of prolexeme descriptions, and a set of prolexeme relatives, for instance the derivatives such as Parisian derivated from Paris.

Conceptual Model (2)

- Top level
  - Common to all languages represented
  - Pivot: « abstract » proper name
    - **Shared** by prolexemes of different languages
    - **Distinct** for each point of view:
      - *St Petersbourg* → P1, *Leningrad* → P2, and P1 ≠ P2
- Linguistic level
  - One database for each language, linked via pivots

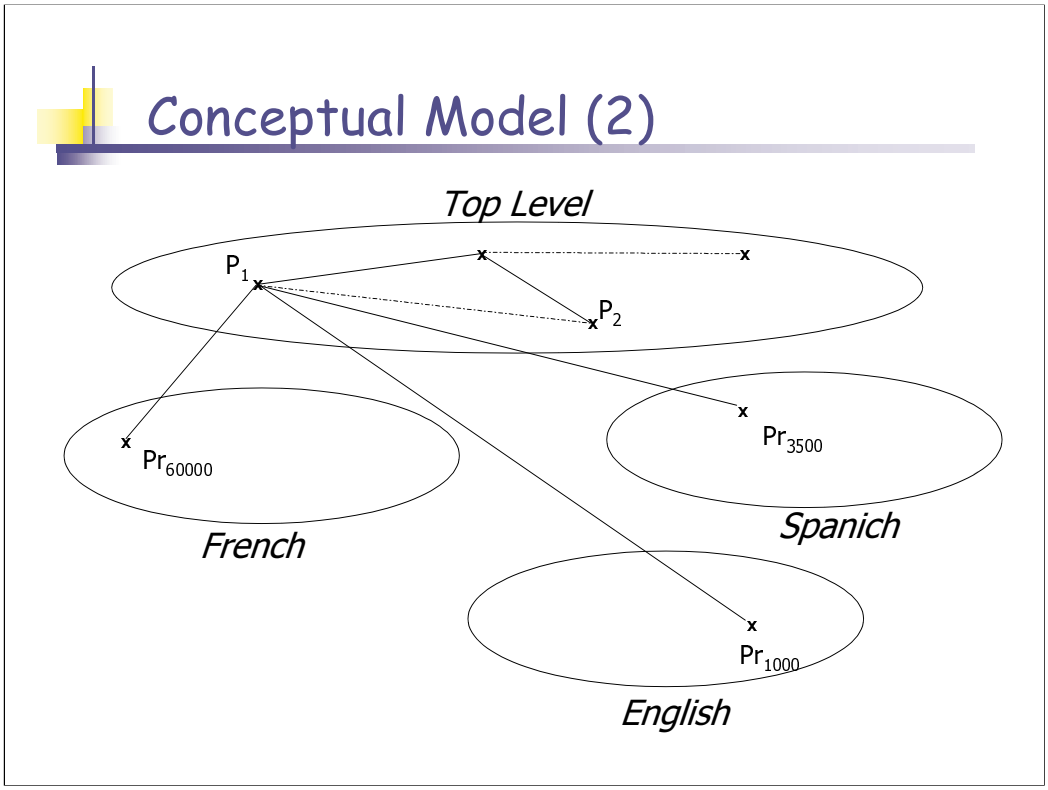
The top level is a description which is shared by all languages represented in the database.

For that reason, in this level, proper names are abstracted by an identifier that we call « pivot », which is similar to the Inter Lingual Index of euroWordNet.

Links between languages are done via this pivot.

Notice also that different points of view on the same referent, such as St Petersbourg and Leningrad for example, have different pivots.

The linguistic level represents a description of one language, then in this level there is one database for each language.



To make thing more clear, I‘ve done this figure:

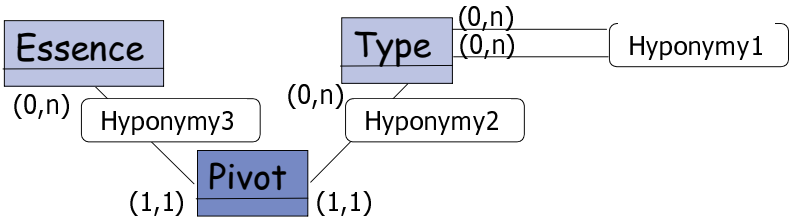
You can see that in the top level we have pivots, with relationships such as the synonymy between Saint Petersbourg and Leningrad.

Pivot P1 is linked to one prolexeme in several languages:

In French it is Saint Pertersbourg, and it is the corresponding proper name in Spanish.

Top level (1)

- Semantic classification of proper names
  - Supertypes: anthroponyms, toponyms, ergonyms and pragmonyms
  - Types: country, town, celebrity, catastrophy
  - Essences: religious, historical, fictional



Here is the part of the conceptual model that concerns the classification.

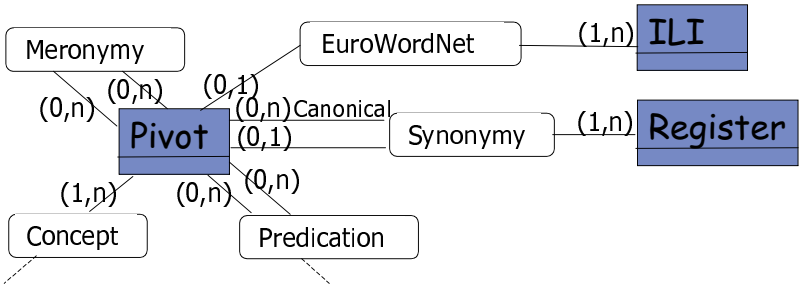
As you can see, proper names are classified using a hierachy of types.

Each pivot is related to one and only one type, which can have supertypes.

Moreover we have added the notion of essence to specify the general register to which the proper name belongs.

Top level (2)

- Semantic relationships between proper names
  - Synonymy: *P(Saint Petersbourg)* ≅(diachronic) *P(Leningrad)*
  - Meronymy: *P(Zinédine Zidane)* ⊂ *P(Real Madrid)*
  - Predication: *P(W. A. Mozart), P(La Flûte enchantée), predicate*
    where predicate is « compositeur »



The second part represents semantic relationships between proper names.

The first is synonymy.

Here I use a functional notation to say « the pivot of Saint Petersbourg is a synonym of the pivot of Leningrad, in a diachronic register.

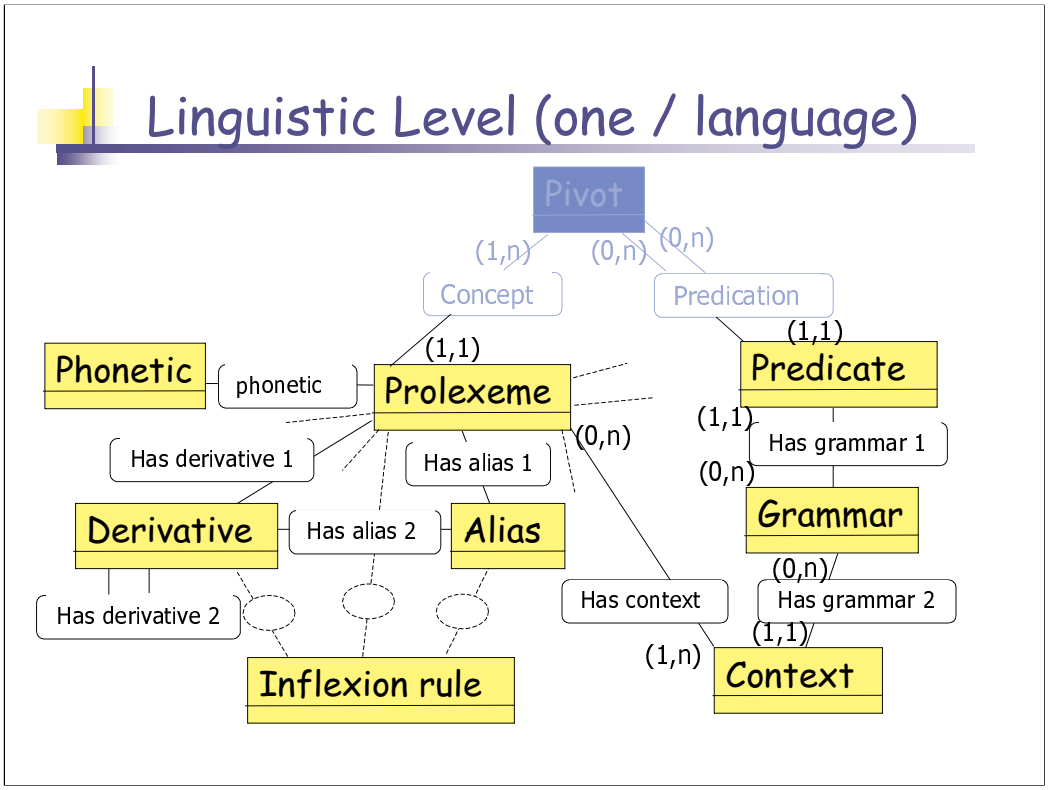
Meronymy is a relationship between a whole and its parts.

Predication represents any relation between two pivots, caracterized by an expression called the predicate, for example Compositeur in French.

The dash lines represent links to the linguistic level.

I've already shown that one pivot is linked to one prolexeme in each language.

Also, the relationship predication links two pivots with one expression in one language.



There are many features that caracterize proper names in languages, I show only some of them here:

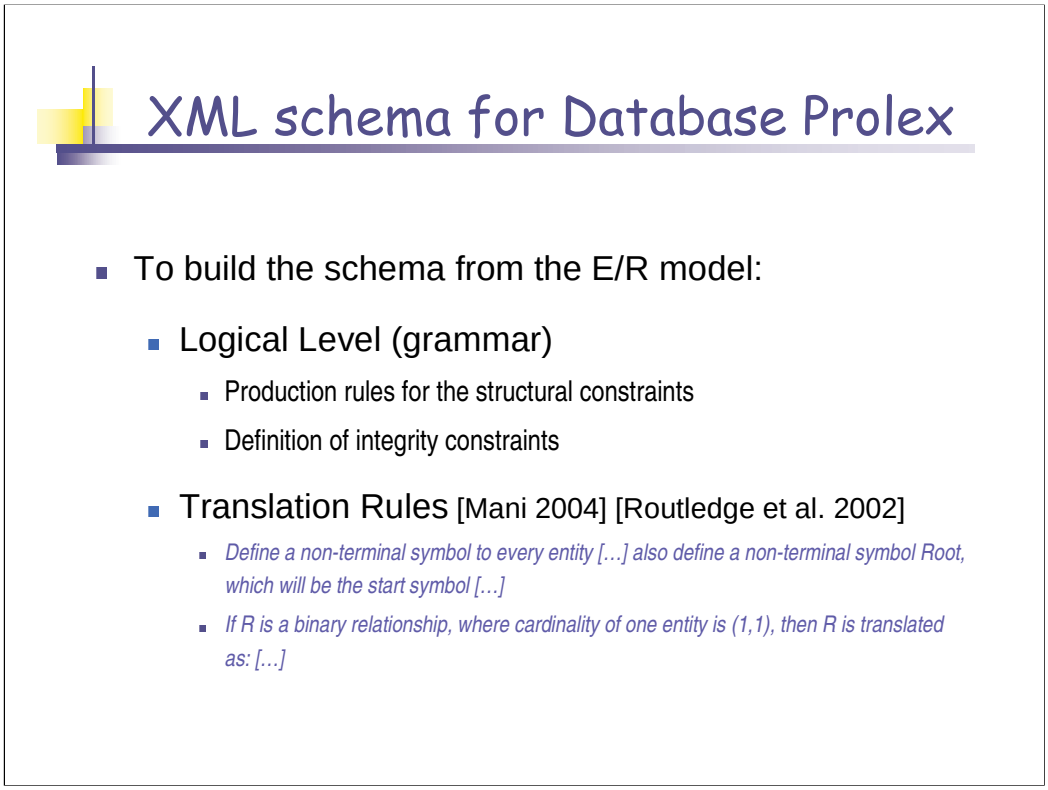
One can see that, except for the predicate, everything is directly related to the prolexeme.

The prolexeme is the kernel is this level.

As I have told, a part of the model represents features such as phonetic, and another part deals with expressions related to the prolexeme, namely derivatives and aliases.

Notice that derivatives can also have aliases. And all of them have an inflexion rule, which specifies the set of their instances.

Notice that the model includes the notion of context, associated to a local grammar, such as « the river Kwaï » for example.



Now, the aim of the presented work was to represent in XML the content of the database Prolex.

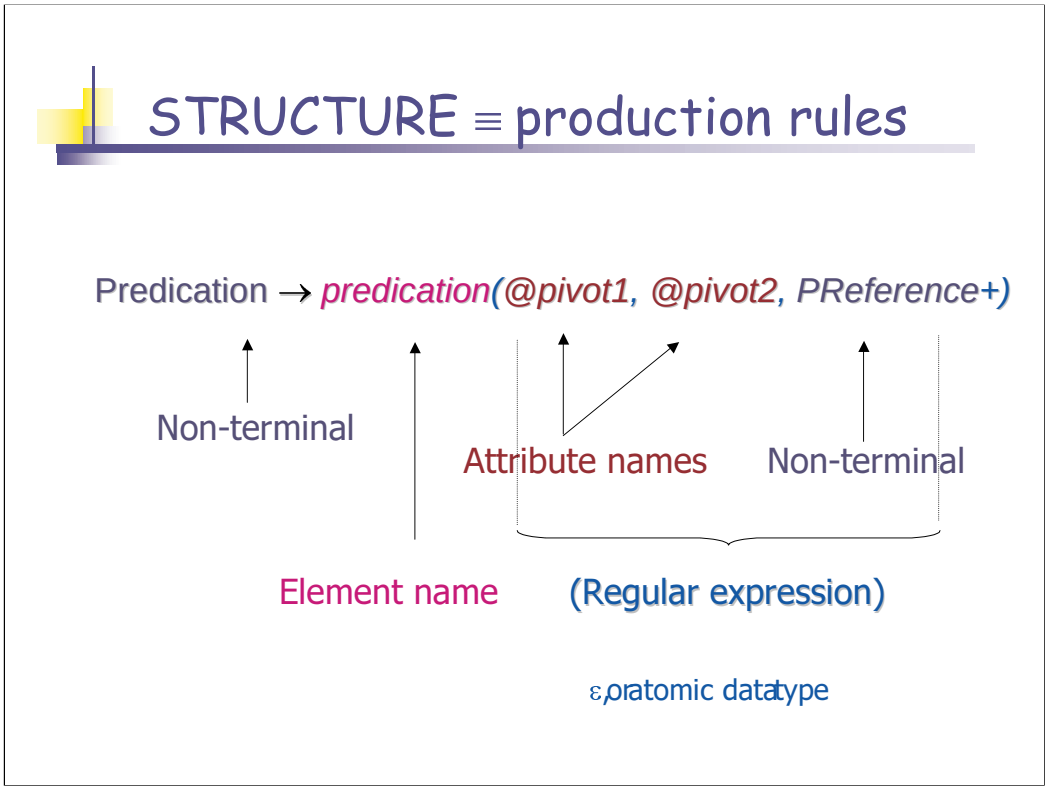
thus, we tried to build the XML schema from the conceptual model of this database.

For that purpose, we have first chosen a logical model to represent the target XML schema,

which is a grammar with production rules, plus integrity constraints,

and then we have applied translation rules recommended by authors (Routledge and Mani).

(References are given in the paper.)



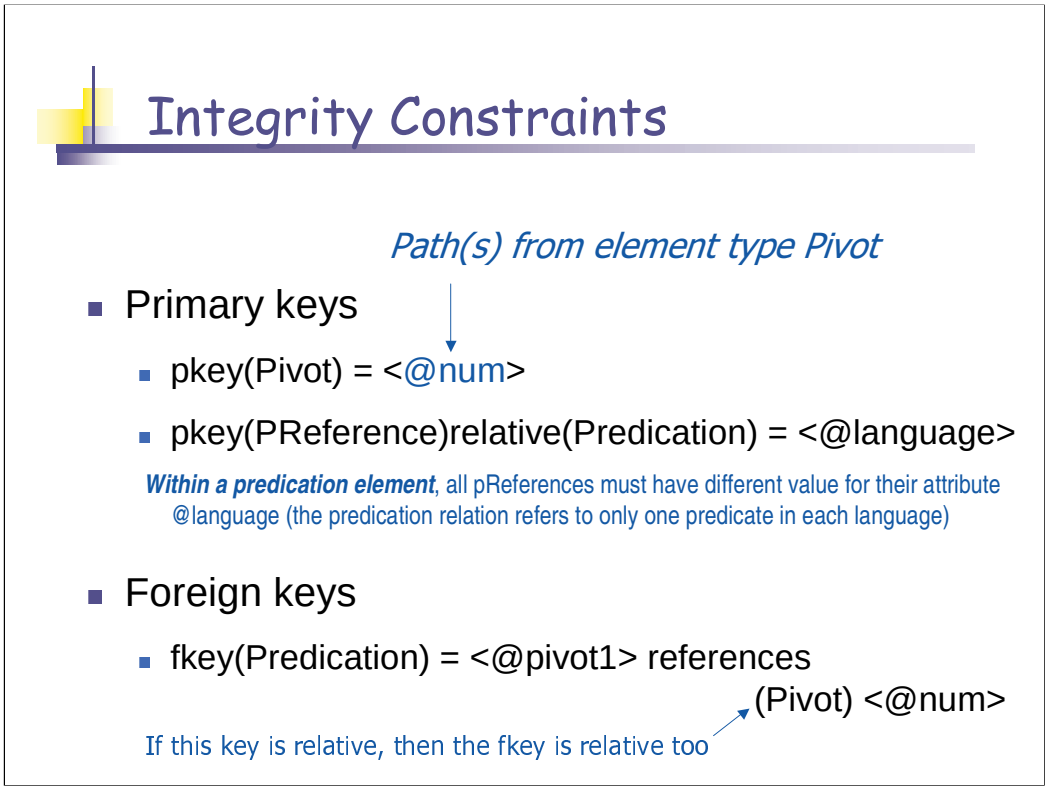
In the logical model, the structural constraints of XML documents are specified using production rules.

More precisely, the grammar is composed of set of non terminal symbols, and terminals which are element names, and attribute names.

Non terminal symbols are called « TYPES ».

These production rules define the content of elements of an XML document.

As usual for XML elements, this content is defined by a regular expression, which can be the empty sequence, and can include atomic data type.



From a database point of view, it is useful to have in the logical model the information about integrity constraints, such as keys and foreign keys.

For that purpose we propose these notations, which uses the types defined by production rules:

We define a primary key for a type, here the type is Pivot by giving the paths from an element pivot to the components of its key. Here the definition means that in the whole document each element pivot is uniquely identified by the value of its attribute num.

We can also define a relative primary key. It is a key that holds only for a part of the document.

In this example, the definition says that within an element Predication each element preference is uniquely identified by the value of its attribute language.

Last, we define foreign keys:

this example of definition means that, in an element predication, the value of the attribute pivot1 must also be the value of the attribute num of an element pivot.

from E/R model to logical model

- Initialization [Mani 2004]
  - Define a non-terminal symbol N_i to every entity E_i […] also define a non-terminal symbol *Root*, which will be the start symbol […]
  - Define an **element name** n_i for every **entity** that has a key constraint or […]. + Linking of non-terminals with element names.
  - Define an **attribute name** to attribute of any entity or relationship […]
  - Define production rules corresponding to every non-terminal symbol as: $N_i \rightarrow n_i(R\_E_i)$, with just the attributes of entity E_i in R_E_i

Now, in order to get the logical model corresponding to the conceptual model, we have followed some rules:

First steps are to translate entities:

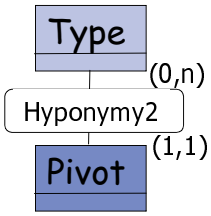
Entities become Types, and for most of them it means that they become XML elements.

Notice that entity and relationship attributes are all translated into XML attributes, which is not the case in all translation propositions.

Entity attributes are placed in the production rules defining corresponding element content.

Translating relationships (1)

- Binary relationship *R(att), att* a list of attributes, with one (_,1) cardinality; let entity E_1 be at association end (_,1):
  - If the cardinality is (1,1) and N_1 does not appear in any R_N_i then $R\_N_2=(R\_N_2, N_1$°) and $R\_N_1=(R\_N_1, att)$;
    °depends on the cardinality of E_2 in *R* (?, * or +)
  - $R\_N_1=(R\_N_1, att, @idN_2)$ and add fkey from $@idN_2$ *to* N_2



Type → type(@name) ; Pivot → pivot(@num)

Type → type(@name, Pivot*) ; Pivot → pivot(@num)

Type → type(@name) ; Pivot → pivot(@num, @idType)
fkey(Pivot)=<@idType> references Type<@name>

The main difficulty of translation from a conceptual model to an XML logical model concerns relationships.

As for the relational model, one can distinguish several cases of relationships, but contrary to the relational model, there are several possible translations for each case.

Here we consider a first case, with one association end having maximal cardinality 1:

In that case it is possible to nest the type at end (1,1) inside the other type, if it is not already nested in another element.

But it is also possible to simply add an attribute in the type at end with maximal cardinality 1 in order to get a reference to the other type.

In this example, we have chosen the second solution

because we were knowing that the pivot was playing a central role in the system and then

we did not want to add an intermediate level to access it.

Translating relationships (2)

- Binary relationship *R(att)* of type N:M: choose E_1 to be the main entity
  - Add a new non-terminal symbol *N* and a new element name *n*
    RPhonetic rPhonetic
  - *R_N=(@N_2ref, att)*
    RPhonetic → rPhonetic(@idPhonetic)
  - fkey(N)=<@N_2*ref* > references N_2 key
    fkey(RPhonetic)=<@idPhonetic> references (Phonetic)<@num>
  - *R_N_1=(R_N_1, N°)*; °is * or +
    Prolexeme → prolexeme(@num, RPhonetic*, ...)

A relationship of type N:M can be translated in many ways.

Most of them suppose to choose a main entity between the two associated entities.

For example this rule describes one possible translation in which the relationship is translated into a new element.

Having chosen a main entity, we can create a new non terminal symbol and a new element name,

whose production rule contains attributes of the relationship,

plus one reference to the second element.

Then, the new non terminal symbol is added to the regular expression defining the main element content.

For this example, we have chosen Prolexeme as the main entity, then we have created a new type Rphonetic to represent the link between a prolexeme and its phonetic descriptions.

Can we build a schema directly from a conceptual model ?

- Yes, to some extent
  - [Routledge et al. 2002] Object Oriented Modeling
    - *Conceptual model = UML classes*
    - *Logical model = UML Stereotypes*
    - *Physical model = a schema in XML Schema*
  - [Mani 2004] Entity / Relationships
    - *Conceptual model = EReX (E / R, plus extensions)*
    - *Logical model = XGrammar*
    - *Physical model = a schema in some schema Language (DTD, RelaxNG, …)*
- But…
  - (Too) many possible ways to represent relationships in XML
  - It is always necessary to reorganise nesting, according to the application

There are many other points to consider, that I can't detail here.

We can say that there are some usefull guidelines in these articles,

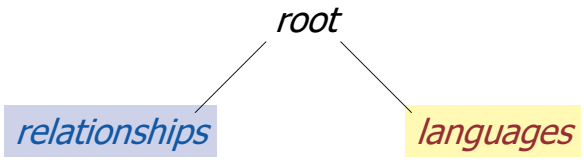
It has been necessary to make choices between several possible translations,

and to reorganize nesting, in order to obtain an accurate XML schema for proper names and their relationships,

that I will brieffly present now.

XML Schema for proper names

- Global structure: two parts
  - Top level (semantic […]) → Relationships
  - Linguistic and instance levels → Languages

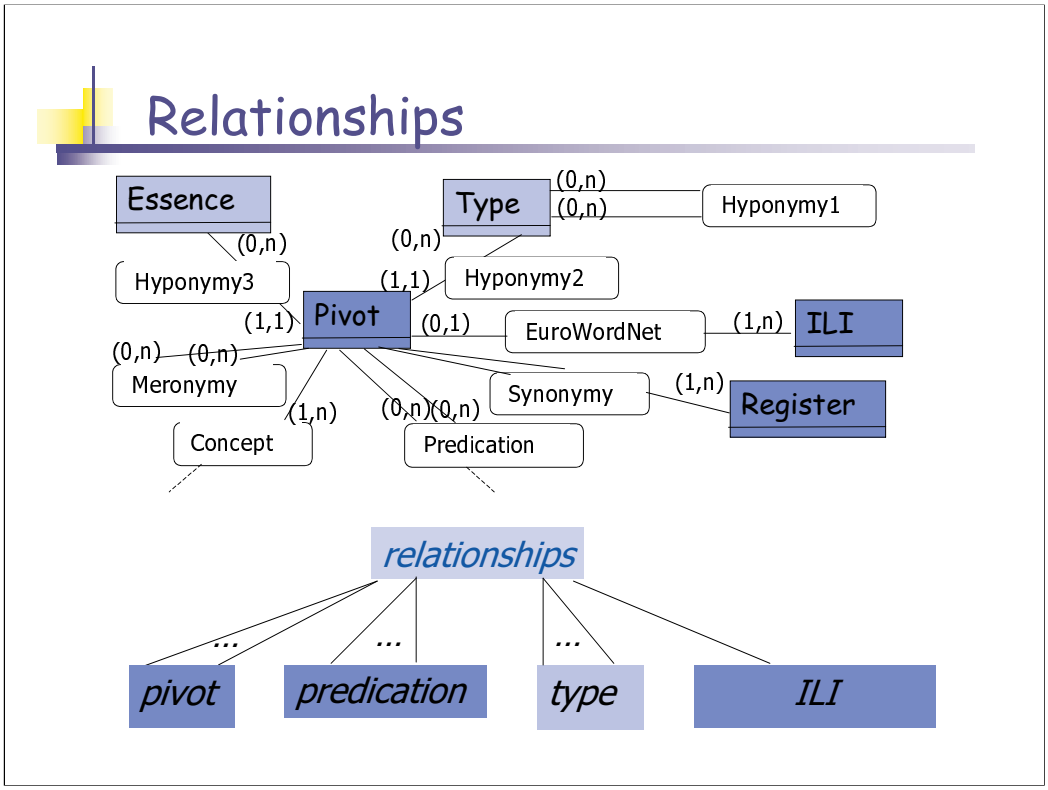


The XML schema is divided into two parts :

One for the top level of the conceptual model,

And one for the lingusitic level.

Notice that these element names where not obtained from an entity in the conceptual model, but we have add it to get an XML logical structure.

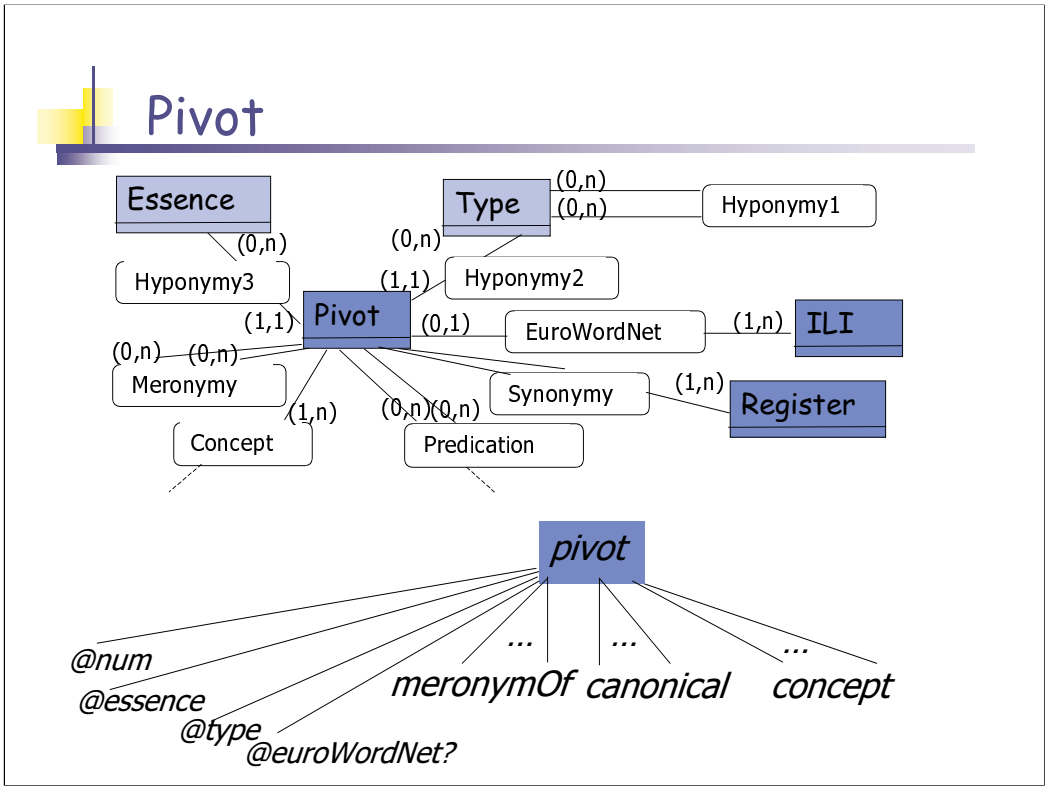


Under the first element we have:

a set of elements called pivot, a set of elements called predication, a set of elements called type.

And an element called ILI, which is a list of Inter Lingual Indexes.

Notice that elements pivot, type and ILI come from entities, whereas element predication comes from a relationship.



All relationships have been translated considering the pivot as the central concept in this part.

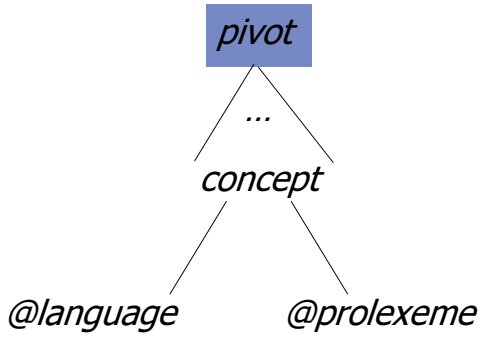
This is why we find many references in the element pivot,

They describe pivot's features, for example its type.

Relationships meronymy, synonymy and concept have been translated by creating new elements,

which refer to another pivot for the meronymy and the synonymy, and to a prolexeme for the concept.

Link to linguistic level



pkey(Concept) relative (Pivot) = <@language>

One pivot can't be related to more than one prolexeme in the same language

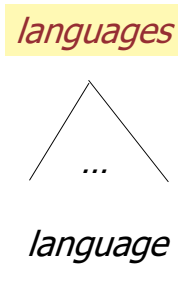
Indeed, the link between the two parts is done via elements concept and also predication.

An element concept contains two attributes, the first references the language and the second references the prolexeme in that language.

Recall that one pivot can reference only one prolexeme in each language.

Similarly, elements prolexeme, which are stored in the second part of the document, have also a reference to their pivot.

Linguistic and instance levels



The second part of a prolex XML document contains a set of elements called language, one for each represented language.

Each element language contains the description of prolexemes, structured according to the conceptual model.

I can not enter in details... See the paper.

Conclusion and Future work

- Current database Prolex:
  - Stored using a Relational DBMS
    - http://tln.li.univ-tours.fr/tln_prolex/prolex.php
  - XML views of this database content
    - In order to export current resources and import new resources (in cooperation with several european partners)
- Future work:
  - To developp tools using XML views of database Prolex
  - To integrate proper name description in a standard markup language s.t. LMF (Lexical Markup Framework)
  - To augment current TEI (Text Encoding Initiative) proposal concerning proper names (named entities)

As a conclusion, the current database Prolex can be seen at this address, mainly for French proper names, but we have already data for other languages: English, Dutch, Italian, German, Spanish and Portuguese.

We have designed an XML schema in order to get views of this database content, for exchange and integration purposes.

Future work will be to developp tools using such XML views,

and to see how to integrate our description of proper names in standards such as LMF for the resource format,
Or the TEI for proper names tagging in texts.