PIVAX, an online contributive lexical data base
for heterogeneous MT systems using a lexical pivot

Hong-Thai NGUYEN1,2  Christian BOITET1  Gilles SÉRASSET1
1 GETALP, LIG, Université Joseph Fourier
385, rue de la Bibliothèque, 38041 Grenoble cedex
2 MICA lab, Université de Hanoi & CNRS
Hai Ba Trung, Hanoi, Vietnam

e-mail : Hong-Thai.Nguyen@imag.fr, Christian.Boitet@imag.fr

Abstract

PIVAX is the first online contributive lexical database system allowing to create, maintain and manage the lexical resources of MT systems using a "lexical pivot", and heterogeneous because their language-specific components are developed at different places, with different linguistic approaches and computational tools. Only the most basic language-specific lexical information must be stored in PIVAX, so that developers can use their own tools and protect proprietary information. An API is used for synchronizing PIVAX with other databases. PIVAX is written over the generic Jibikiki platform. It is an evolution of the hypertextual PARAX multilingual database, and begins to be used by a U++C1 project using UNL UWs as lexical pivot.

Introduction

Most large MT systems using dictionaries end up distributing the lexical information in many files with various formats, containing various sorts of information, often represented by cryptic codes defined in other files. For example, an MT system written in Ariane-G5 can have up to 70 dictionary files2. While that maximum is rarely attained, most "language-pairs" use more than 15. Commercial systems such as Systran, Softissimo, ATLAS (Fujitsu), The Translator (Toshiba), etc., have dozens of specialized dictionaries for various domains. ATLAS-II v.13 has a total of 5,44 M entries in its specialized dictionaries. As early as 1986 (Boitet and Nedobekine 1986), we have begun to study how to maintain in a centralized multilingual lexical database (MLDB) the lexical information of multilingual MT systems of the kind that were developed with Ariane-G5, which allows to share analysis, transfer and generation "steps"3, with a view to automatically generate the phase-specific dictionaries from them. That was successful up to a point: during its activity (1986-1995), the BVITAL MT firm managed to do that with its BDTAO tool for all lexical information, except the lexical transfer dictionaries of general (non terminological) words.

There is also a need felt by the MT industry to share lexical data, in particular but not only multilingual terminology, because of its high development cost, and because of the pressure of users who want to port their "user dictionaries" from one MT system to another. There is a tension between that desire and the desire to protect the "codes" specific to each system and in this sense "proprietary". On this side also, past efforts have not been quite successful.

Finally, an increasing number of "pivot" MT systems are developed in an heterogeneous way by independent groups. The pivot can be an annotated natural language, as Esperanto in the DLT project (1982-89), a domain and task-specific interlingua such as the IF (Interface Format) of the CSTAR-II and Nespole! spoken dialogue translation systems, or a linguistico-semantic interlingua such as UNL4.

The PIVAX approach is a compromise between generality and practicality. We would like to develop an architecture for an ideal database able to support any set of MT systems, with arbitrary linguistic architectures, but it seems inherently impossible, because the basic types of information are too different, or at least extremely complex, and because links would have to be maintained at too many levels (between word forms, lemmas, derivational families or "prolexes", and lexies).

PIVAX is the first online contributive lexical database system allowing to create, maintain and man-

---

1 http://www.unl.fi.upm.es/consorcio/index.php
2 21 morphological analysis dictionaries (7 for usual morphs and 14 for fixed idioms), 14 for lexical expansion in analysis (from lemmas to simple lexical units to compound lexical units and acceptions), 7 for lexical transfer, 21 for lexical expansion in the target language, and 7 for morphological generation.

3 In Ariane-G5, a "step" consists in a lattice of "phases" (such as lexical transfer or structural transfer), each being a set of "lingware components" (such as declaration of variables, templates, test or assignment procedures on decorations, grammars or automata, dictionaries) written in some SLLP (Specialized Language for Linguistic Programming).

4 "such as", because the same idea could be developed by basing the UW notation not on English, but on another language, such as Sanskrit to mediate between the "lexical spaces" of the Indian languages.
age the lexical resources of MT systems based on a "lexical pivot", and heterogeneous in the sense that their language-specific components may be developed at different places and with different linguistic approaches and computational tools. Only the most basic language-specific lexical information must be stored in PIVAX, so that developers can use their own tools and protect proprietary information.

In section 1, we present the motivations behind this work, recalling previous related work and experiments, to understand the reasons behind their limits. That leads us to specify the linguistic architecture of PIVAX in section 2. In section 3, we describe the current implementation of PIVAX and its application by the U++ Consortium. The UNL UWs constitute the lexical pivot of the MT system(s), but UNL is not at the center of the macrostructure, it is managed as any natural language.

1 Related work and main requirements

1.1 Related work on MLDB for MT

MNT-CICC. The MMT project\(^5\) was an MT project of the CICC financed by the Japanese ODA and involving 5 Asian languages (Japanese, Chinese, Thai, Malay and Indonesian). It was executed and co-financed by 5 large Japanese companies already offering commercial MT\(^\circ\). Reusing ideas and resources from the EDR project, they adopted the semantic pivot approach, already implemented in the ATLAS-II and PIVOT systems, but tried to make it possible to develop the language-specific parts independently by identifying the "concepts" of the interlingua (IL) vocabulary by definitions in English. But this constraint proved too strong in practice (how to distinguish 2 types of fish by 2 definitions?), and there was no lexical database directly accessible by all developers in real time.

UWGATE. This centralized lexical database of the UNL project (http://www.undl.org) uses file swapping (protected gzip file), even for 1 dictionary entry. Moreover, the response time is often far too long (2 or 3 days or never…).

UNL-deco. This Web service has been developed by G. Sérasset for deconverting from UNL into French. It contains a lexical database accessible on the Web in real-time, but currently limited to French and UNL. In fact, it has not been used to develop the UNL-FRA system, and even less for developing the UNL to French, Russian and Spanish system experimentally used for the translation of the Unesco B@bel website (Boitet, Boguslavskij & Cardenosa 2007), because it does not offer any tool to support lexicographical work (like sorting, filtering, assistance to code assignment…) and its interface is not user-friendly enough, even with only UNL and one language, contrary to PARAX. But PARAX is designed for a stand-alone machine only.

1.2 Main difficulties & limits

The above problems seem mainly to come from the extreme difficulty to build a “universal” lexical database for MT systems: managing all the aspects is almost impossible (from the construction of the lexical data until the automatic extraction from some modular dictionaries in the various systems, e.g. morphological analysis in Systran, lexical transfer in Neon (by Pr Shi X., see http://mt.xmu.edu.cn/). Not only are the theoretical difficulties harder than in the case of a multilingual lexical database for human usage, such as the Papillon-CDM part of the Papillon server (http://www.papillon-dictionary.org), but the practical difficulties are also almost insurmountable (diversity of the formats, nature of the information, problems of IPR). Moreover, if each partner has his own tools and resource formats, and works on a copy of what has to be shared at running time, it is almost impossible to develop in a consistent way.

1.3 Main requirements & available tools

1.3.1 Sketch of a solution to a simplified problem

The above as well as previous experiments within the UNL project have convinced us that developers of dictionaries in such a large, heterogeneous MT system absolutely need to do most of their day-to-day work using a common lexical database. As developers are distant, the only solution today is to put that database on the web, with enough user-friendly tools, so that it is really used for direct work, and not only for episodic synchronization.

Considering the nature of the lexical information, our hypothesis is that it is possible to simplify this problem and to arrive at a problem solvable in theory, implementable, and useful in practice. This simplification has two aspects. The first consists in using a "pivot" macro-structure, which implies to work only with MT systems having a "lexical pivot". The second is that the developers of the language-specific components of the multilingual MT system control the lexical data completely, in particular their “internal coded” representation.

That approach has already been carried out and validated by the lexical database PARAX (Blanc 1999). But this development environment, while excellent for stand-alone work, does not allow co-operative work. PIVAX is thus inspired by PARAX, but ex-

\(^5\) Multilingual MT project, see http://www.mt-archive.info/MTS-1993-Funaki.pdf

\(^6\) Fujitsu (ATLAS-II), Toshiba (AS-Transac), Hitachi (HICATS), NEC (Pivot).
tends it, and aims at a distributed wiki-like environment.

1.3.2 Qualities and limits of available tools

PARAX. This large-scale multilingual lexical database has been implemented in hypertextual form. It comprises a UW dictionary and several monolingual dictionaries (at present French, Japanese and Russian). Each dictionary is a Revolution™ (ex Hypercard) stack. The link between dictionaries is achieved through UNL: each word in a monolingual dictionary entry is associated with a UW from the dictionary built by the Tokyo team (Uchida 2002), which contains more than 100,000 headwords. Each NL dictionary contains 30 to 60 K words. A complete morpho-syntactic description of the words is at present only available in the French dictionary.

Consultation. An example is shown on Figure 1: clicking on a UW in the UW dictionary (at the left) accesses the front pages of the monolingual dictionaries, each of them displaying the word meaning(s) associated to the UW. Similarly, clicking on a word meaning in a word card of a monolingual dictionary opens the other monolingual dictionaries, and displays the associated word meaning(s).

![Figure 1: PARAX interface](image)

This is an efficient working environment, in use by linguists for a long time, hence we try to offer the same kind of interface and functionalities in PIVAX.

Jibiki. To develop PIVAX, we used the Jibiki platform (Sérasset 2005), a generic environment for quickly building web-based contributive lexical databases having any kind of architecture. It now supports several free dictionaries for human usage (Papillon-CDM), a research-oriented lexical database (Papillon-NADIA), a large French-Estonian dictionary (GDEF), and the LexAlp quadrilingual terminology of the Alpine Convention.

To create a lexical database, one describes its macro-structure and microstructure as metadata (XML schemata). The underlying database structure used for physical storage and the corresponding interface are automatically generated. A developer can customize this interface directly by modifying the automatically created presentation file (XHTML) or style sheet (XSL). Due to this architecture, the underlying database structure does not need to be modified if new languages with specific variants of the XML schemata are introduced. Jibiki supports all functions needed in a contribution environment: user & group management, edition in Wiki-like mode, import/export, and administration.

2 Specification

2.1 Typical scenarios

Contributor. A contributor or a linguistic developer can connect and contribute to the lexical pivot (e.g. UWs) and to the various languages, according to his access rights. Each connected user has his own working space. If two users work on the same object in writing mode, PIVAX will issue a warning and:

- show to the user who saved first that the data has been modified,
- offer a possibility to show and merge the two modifications.

Administrator. PIVAX proposes management functions to import, export, backup, etc., at volume or dictionary level.

Integrator. The developers of monolingual components can use PIVAX via a simple API.

2.2 Organization

2.2.1 Macro-structure

The macro-structure has three layers:

- For each natural language (NL) supported, and each formal interlingua\(^8\), there are:
  - one or more volumes of lexies (and associated information). Lexies correspond to word senses in dictionaries.
  - a unique volume of axemes ("monolingual acceptions"). An axeme links synonymous lexies of the same language.

\(^8\) The same database could be used to develop a MT system using UNL, and another using IF (Nespole! project).
• A unique volume of axies (interlingual acce-
tions), linking synonymous axemes.

2.2.2 Micro-structure

Axemes and axies: they are simply links, repre-
sentable as sets of (identifiers of) lexies and axemes,
respectively. As in Papillon-NADIA and in LexAlp,
2 axemes or 2 axies in the same volume can be linked
by a refinement link.

NL volumes: an entry contains a lemma, its class
(POS), its word sense id (if any), a comment (for the
other developers) and proper information specific to
each volume and protectable. Metadata include id
(unique), process_status, status, accessibility.

As said in 1.3.1, we require only a minimal common
information to be shared, that which can identify a
lexie (lemma, POS, sense id). The "proper informa-
tion" field is considered as textual data and can be
treated differently for each volume. For example, a
volume managed by a French academic lab might
contain all the French information found in its MT
system(s), while a volume contributed by Systran
might contain only part of its "proprietary codes".

While the common information is public by neces-
sity, hence visible, it can be write-protected by the
owner of the volume. The "proper information"
can also be read-protected. To achieve that, each XML
code of the logical structure has associated access
rights (as files in Unix). Here is an example of a
hypothetical article in a Systran-owned volume.

```xml
<d:data>
  <p:lexie p:id="lexie.systran.test.12004"
p:process_status="UNPROCESSED"
p:status="UNKNOWN"
owner="Systran"
p:score="0.00003">
    <p:lemma p:access="Public">
      fier
    </p:lemma>
    <p:class p:access="Public">
      Adj
    </p:class>
    <p:comment>
      "fier" can also be a V ("se fier à")
    </p:comment>
    <p:proper_information
      p:access="Hidden">
      /* Systran proprietary codes */
    </p:proper_information>
  </p:lexie>
</d:data>
```

IL volumes: interlingual entries are always defined
at the semantic level, and can aptly be called lexies.
By contrast, the usual MT dictionaries often distin-
guish word senses in the bilingual transfer entries, so
that, when such a dictionary is imported, its "lexies"
have to be later split into real lexies (e.g. "fier_adj"
has 2 word senses). The microstructure of an entry
depends on the IL. In the case of UNL, an entry con-
tains the UW as lexie, a comment in English about its
meaning, and examples of use.

2.3 Import, export and synchronization

Import. The "native" import data method in Jibiki
has been described in (Mangeot-Lerebours, M.
2002). Knowing the microstructure into which to
import, such as CDM (for Papillon), one gives to the
import program the Xpaths of elements correspond-
ing to CDM elements in the XML file to be imported.
Elements not reached are discarded or collected un-
der a special "other" element.

One can also use direct methods to compile the input
dictionary into the XML form of the target PIVAX
volume, either by using a dictionary compilation tool
(Nguyen Doan 1998, Teeraparbseree 2002), or more
directly by writing Perl scripts.

Export. The default output is an XML file which
contains all information about sets of entries. Each
set is retrieved by some search criteria. One can
download this file and transform it into any kind of
presentation by XSL transformations.

Synchronization. There are two cases:

• with another lexical database such as Papillon,
  used in parallel by a team. The problem is that
  we can’t know exactly the status of the data in
  the other database, and which part of the data
  should be synchronized. More research is
  needed. As for now, we have to rely on "frozen
  states".

• with local data developed offline by a single
  person using stand-alone tools (Excel, File-
  Maker, etc.), to manipulate some subset of the
  database. If the exchange format contains
  unique identification keys, then, the envisaged
  solution is to use a program like CVS to com-
  pute the differences with the last state of the data
  on the PC, the current state, and the state of
  PIVAX on the server, when the contributor con-
  nects, and to transform them into a "narrative
  program" using the primitives of PIVAX user
  interface to mimic "what a contributor could have
done online to produce the same changes".

3 Implementation and application to
the U++C common lexical database

3.1 Implementation in Jibiki: main points

We have used Jibiki version 1.0, which is imple-
mented in the Enhydra environment version 6.2,
and uses Postgres as DBMS. It has taken us less than
2 weeks to learn how to use it, to describe the macro-
structure of PIVAX for all volumes of the intended
application to U++C, and to get a first version up and

9 Jibiki platform: http://gohan.imag.fr/ijiwiki/
10 Enhydra environment: http://enhydra.org
running. Customizing the interface was more challenging, because we wanted it to be quite flexible. Version 1 was completed in about 6 weeks.

3.2 PIVAX for U++C

With the help of an intern student, we have imported a large part of the lexical data of the U++C, in about 3 months. Getting access to the data and debugging the Perl scripts took most of the time, but the result was very good (given that imported data was already formatted), and the programs quite fast. Here is the current count:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Type/Language</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNL-Deco</td>
<td>UNL-FRA</td>
<td>39389</td>
</tr>
<tr>
<td>PARAX</td>
<td>UNL-FRA</td>
<td>18978</td>
</tr>
<tr>
<td></td>
<td>UNL-CHN</td>
<td>9315</td>
</tr>
<tr>
<td></td>
<td>UNL-RUS</td>
<td>13817</td>
</tr>
<tr>
<td></td>
<td>UNL-ESP</td>
<td>833</td>
</tr>
<tr>
<td>UNLK</td>
<td>UNL</td>
<td>21618</td>
</tr>
<tr>
<td>UPM UW</td>
<td>UNL</td>
<td>21618</td>
</tr>
<tr>
<td>SurviTra</td>
<td>ENG-Hindi-FRA</td>
<td>445</td>
</tr>
<tr>
<td>Systran</td>
<td>ENG</td>
<td>83486</td>
</tr>
</tbody>
</table>

The first version has been demonstrated at two U++C meetings (Madrid, 6/07 and Grenoble, 7/07). The feedbacks will be incorporated in version 2, which should begin to be used in practice by the U++C developers at the time of this conference or before. A demonstration should be possible.

3.3 PIVAX interface and usage examples

- Basic search and navigation

The search interface is inspired by PARAX: the user types a headword to search (always in the left of the working space), e.g. “tester” in a French monolingual volume. The results appear in paging mode (very useful if there are many answers), and the user can move the mouse to see the details of each entry. The background color shows the status of each entry: whether it is being modified by another user, or being processed, or free for editing.

To edit an entry, one clicks on the button next to it. PIVAX then searches all synonymous entries (linked to it by axemes and axies links) and shows them in the next columns. Each group of synonymous entries has a particular color. Users can move, show, and resize columns.

When one finds some problems in an entry, but doesn’t want to change the navigation context, one can click on the (+) button next to the entry, to memorize it to a quick list. The quick list is destroyed at session logoff.

- Advanced search

Advance search allows to express a boolean combination of several search criteria and to name it in order to reuse it later.

- Editing

As said earlier, developers can easily prepare a customized interface for each volume. A contributor can fill the fields in each volume entry like: headword, category, definition, etc.

When a user links the current entry to another entry, a pop up menu appears to assist the search. The relation type is shown on a pop down menu in each corresponding target entry.

The current entry can be linked to some users so that they are notified when any modification happens. Users can update (thus storing the modification but not changing the entry status), undo, and save permanently (setting the status of entry to "finished").
User profiles exist already in the standard Jibiki platform with password modification, group management, etc. In PIVAX, we have added alias and interface parameters management.

PIVAX is written over the generic Jibiki platform and is an evolution of the hypertextual PARAX multilingual database. Its first instance is dedicated to a U++C project where UNL UWs constitute the lexical pivot. It should begin to be used in practice shortly. However, there is still interesting work to do on how to synchronize it with other lexical databases or data collections (files) managed by teams of developers such as MT teams, and with data prepared offline by individual contributors.

Acknowledgements

This work has been partially supported by a MIRA (Rhône-Alpes Region) grant. Also, it could not have been built so quickly without the help of E. Blanc, who introduced us to PARAX.

References


U++C http://www.unl.fi.upm.es/consorcio/index.php