Basque: A Case Study in Generalizing LaTeX Language Support

Jagoba Arias Pérez
Alameda Urquijo s/n
Bilbao, 48013
Spain
jtparpej@bi.ehu.es
http://det.bi.ehu.es/~apert

Jesús Lázaro
Alameda Urquijo s/n
Bilbao, 48013
Spain
jtplaarj@bi.ehu.es
http://det.bi.ehu.es/~apert

Juan M. Aguirregabiria
B^o Sarriena s/n
Leioa, 48940
Spain
wtpagagj@lg.ehu.es
http://tp.lc.ehu.es/jma.html

Abstract

The multilingual support of LaTeX presents many weak points, especially when a language does not present the same overall syntactic scheme as English. Basque is one of the official languages in the Basque Country, being spoken by almost 650,000 speakers (it is also spoken in Navarre and the south of France). The origins of the Basque language are unknown. It is not related to any neighboring language, nor to other Indo-European languages (such as Latin or German). Thus, dates, references and numbering do not follow the typical English pattern. For example, the numbering of figure prefixes does not correspond to the `\figurename\thefigure` structure, but is exactly the other way round. To make matters worse, the presence of declension can turn this usually simple task into a nightmare. This article proposes an alternative structure for the basic classes, in order to support multilingual documents in a natural way, even in those cases where the languages do not follow the typical English-like overall structure.

1 Introduction

The origins of LaTeX are tied closely to the English language. Since those days, however, it has spread to many different languages and different alphabets. The extent of the differences among these languages is not only related to lexical issues, but to the structure of the languages themselves.

The main problem arises when the syntactic structure of the language does not follow the English patterns. In these cases the adoption of a new multilingual approach is required in order to produce documents for these languages.

Although LaTeX is a highly parameterizable environment, it lacks resources to alter the order of the parameters themselves. This is due to the fact that both Germanic languages (such as English and German) and Romance languages (such as French, Italian, Spanish) — and therefore the most widely spread European research languages that use the Latin alphabet — share a similar word order for numeric references. To make matters worse, the presence of declension in structure such as dates and numbers leads to a complicated generalization of procedures.

This paper describes an alternative structure for the basic classes, in order to support multilingual documents in a natural way, even in those cases where the languages do not follow the typical English-like overall structure. Specifically, the paper focuses on Basque, one of the official languages in the Basque Country, being spoken by over half a million speakers (it is also spoken in Navarre and the south of France).

The rest of the paper is organized as follows: section 2 describes the specific details of the Basque language, in section 3 a brief description of prior work is presented, section 4 describes the different approaches that can be followed to solve the problem, section 5 shows the advantages and drawbacks of the different solutions and finally, in section 6 some brief conclusions are presented.

2 Specific details of the Basque language

The origins of the Basque language are unknown. It is not related to any neighboring language, nor to other Indo-European languages (such as Latin or German). This is one of the reasons why word order and numbering schemes are different from those in English.

Dates and numbers. Basque uses declension instead of prepositions as many other languages. The main difference from other languages that use declension, such as German, is that in Basque numbers are also fully declined, even in common structures such as dates. These declensions depend not only on the case, number and gender, but on the last sound of the word. Another peculiarity of Basque is the use of a base 20 numerical system instead of the traditional decimal one.

This forces us to take into account not just the last figure of the number but the last two figures, in order to determine the correct declension for the number [3]. In the following example, two dates are represented using ISO 8601 and its translation into Basque.

2004-01-11 : 2004ko urtarrilaren 11n
2005-01-21 : 2005eko urtarrilaren 21ean

Note that although both days end in the same figure, the declension is slightly different. The same happens to the year. The extra phonemes have been added to avoid words that are difficult to pronounce. This makes automatic date generation difficult, because it must take into account all the possible cases (as base 20 is used, there may be as many as 20 different possibilities). The different number endings are shown in table 2. Note that there are only twenty possible terminations, and two declension classes are necessary.

Table 1: Endings

Number	Ending (year)	Ending (day)
00	ko	-
01	eko	ean
02	ko	an
03	ko	an
04	ko	an
05	eko	ean
06	ko	an
07	ko	an
08	ko	an
09	ko	an
10	eko	ean
11	ko	n
12	ko	an
13	ko	an
14	ko	an
15	eko	ean
16	ko	an
17	ko	an
18	ko	an
19	ko	an
20	ko	an

Word order When numbering a certain chapter, section, etc., in English-like languages the order is always the following: first, the item class (e.g. “figure”) is named and, afterwards, the number is written. For example, we have “Figure 1.1” or “Table 2.3”. However, this is not the case in Basque. In this language, we must reverse this order: “1.1 Irudia” or “2.3 Taula”. The same applies for chapters, sections and other kind of text partitioning structures.

3 Related Work

Multilingual support for LaTeX is traditionally performed using the Babel package [2]. In this package, the overall structure of documents, such as books, articles, etc., is fitted to different languages by using different variables for the different strings in each language.

For example, we can take the way figure captions are numbered in these types of documents: a variable called `\figurename` contains the string corresponding to the word “figure” in the first part of the caption, while another variable, `\thefigure` contains the number assigned to that caption. When a new figure is inserted in the document, the string preceding the caption is always formed by using a concatenation of both variables. However, this process is not performed by Babel, which would allow a general description of the language, but in the different files that describe the document format: `book.cls`, `article.cls`, etc. Thus, some of the work that should be performed by the module in charge of the multilingual support is made by the formatting part of the typesetting software.

The file `basque.ldf` [1] currently provides support for Basque in Babel. In this file, the most commonly used words have been translated. However, this does not solve the problem of the different order of strings. In [1], a possible solution is proposed using a new package for the document definition: instead of using the multilingual capabilities of Babel to solve the problem, a new document formatting file is defined, where the specific corrections for the language are performed. The limitation for multilingual document generation is obvious in this scheme: the format must be redefined whenever the language of the document is changed. Besides, a new definition for every single class of document must be performed for this particular language — as we are not philologists, we do not know if the same happens in other languages.

4 Approaches to the Solution

The solution to the problem described in this paper must deal with the following issues:

- It must respect all the translations of the different strings generated automatically.
- It must respect not only the translation, but the order of words as well.
- The last problem to solve is the use of the `\selectlanguage` directive, which would allow us to change the hyphenation patterns and the automatic text generation structures dynamically in the same document. This directive is particularly useful for documents which contain the same text in different languages (e.g. user's guides, where the manual has been translated).

The main possible avenues to the solution are the following:

- **Use of specific classes for the language:** This solution implies the redefinition of every document format, in order to embed the corresponding word order alteration for automatic string generation. The main drawback of this alternative is the need for rewriting and adapting all the existing document formats.
- **Use of a specific package for the language:** A second possibility could include the definition of a new package for those languages that require a word order alteration. This package should redefine the `\fnum@figure` and the `\fnun@table` variables (among others, which define the chapter or section name) in order to adapt them to the needs of the languages used. A macro should be used to switch between the two nodes.
- **Inclusion of order parameters in the document class definition files:** This option requires that a new input parameter is defined in the document class to define the order of the words. Basically, it is the same solution as the first one, but merging all the different files for a document class into a single (larger and more complex) file.
- **Redefinition of existing multilingual support files:** This solution implies the addition of several lines to every language support file, where the definition of the automatic strings such as the figure captions or the table captions is performed. For example, for the case of table and figure captions, the definitions for the Basque language would be the following:

  ```
  \def\fnum@figure{\thefigure~\figurename}
  \def\fnum@table{\thetable~\tablename}
  ```

  These definitions should go into the `basque.ldf` file, immediately after the definition of the terms for caption or table names. Thus, whenever a `\selectlanguage` directive is introduced in the document, the Babel package will read the definitions for the new language, which will include the definitions for every string.

5 Comparison of Solutions

We use the following criteria to compare the different solutions:

- **Extent of modification to existing files:** This criterion measures how many existing files will be altered to fix the problem and how complicated this alteration is.
- **Addition of new files:** This criterion measures how many new files are to be added to the LaTeX distribution for each solution.
- **The `\selectlanguage` issue:** This criterion measures how well the solution deals with possibly changing the language of the document dynamically.
- **How easily new automatically-generated strings are included:** In the future, translation of new strings may be required. Therefore, the proposed solution must provide an easy way to include these new strings.

5.1 Extent of Modification

Here is how the solutions fare with respect to the first criterion:

- **Use of specific classes for the language:** This option does not require that any file be modified, because new definitions are described in new files.
- **Use of specific package for the language:** This approach requires no modifications of existing files, since all modifications are included in a new package.
- **Inclusion of order parameters in the document class definition files:** This alternative entails the redefinition of every document class. These should admit a language parameter to determine the correct word order.
- **Redefinition of existing multilingual support files:** This choice implies that every file containing the translation and definition of the automatically-generated strings provides order information for them, and therefore, all the files in Babel should be changed.

5.2 Addition of New Files

Here's how the solutions fare with respect to adding new files:

- **Use of specific classes for the language:** This option requires all document classes to be rewritten for every language that does not follow the English-like structure.
- **Use of specific package for the language:** This approach requires one new file for every language that has not been described successfully in the Babel approach.
- **Inclusion of order parameters in the document class definition files:** This alternative entails no new files, as it is based on the modification of the existing files.
- **Redefinition of existing multilingual support files:** This choice does not need new files, as it is based on the modification of the existing files.

5.3 The `\selectlanguage` Issue

Depending on how generalization of the multilingual support is implemented, the different solutions may (or not) solve the `\selectlanguage` problem:

- **Use of specific classes for the language:** This option does not really use Babel and its macros. As part of the translation of automatic strings is performed by the file defining the format of the document class, support for the `\selectlanguage` directive should be implemented in each document class for every language (not only for those incorrectly supported by the Babel system, but for all of them).
- **Use of specific package for the language:** This approach requires one new file for every language. Hence, a macro would be required in each package to leave things as they were *before* the package was initiated.
- **Inclusion of order parameters in the document class definition files:** This alternative cannot solve the problem, because the order specification is only made at the beginning of the document. A macro could be added to alter its value dynamically throughout the document, but it would be an artificial patch that would not fit naturally in the Babel structure.
- **Redefinition of existing multilingual support files:** This choice does solve the problem, because when a new `\selectlanguage` command is issued, the definitions for the new language are reloaded. It requires no new macro definitions to suit the Babel scheme for multilingual documents.

5.4 Inclusion of New Strings

Here's how the solutions fare with respect to the possibility of including further modifications for strings that could be necessary in the future:

- **Use of specific classes for the language:** As some of the linguistic characteristics of the document are included in the document class, this option does not provide a straightforward method for including changes for problems that may arise.
- **Use of specific package for the language:** The use of a package gives flexibility to the scheme, allowing the insertion of new macros to adapt to the peculiarities of the language. However, the range of possibilities is so wide that a very well-defined structure must be laid down in order to keep a modicum of coherence for creating a document in a different language.
- **Inclusion of order parameters in the document class definition files:** This scheme requires updating several files whenever a new string or scheme must be added.
- **Redefinition of existing multilingual support files:** As this choice uses a single file for every language, it makes updating the elements for Babel very easy.

Table 2: Solution comparison

Solution	Mod.	Cr.	Multi.	Updates
Specific class	X	✓	Dif.	Dif.
Specific pack.	X	✓	Dif.	Dif.
Parameters	✓	X	Dif.	Dif.
Redefinition	✓	X	✓	✓

6 Conclusions

This paper discusses some alternatives to solve the ordering problems that may arise in multilingual documents.

The characteristics of the different proposed solutions are summarized in table 2. Among the solutions, the most suitable would be the redefinition of all the existing Babel files. The reason is simple: it requires the addition of two lines to approximately 45 files, and allows the update of the system in the future, as it maintains *all* the translating issues within their natural context (Babel).

References

[1] Juan M. Aguirregabiria. Basque language definition file. `http://tp.lc.ehu.es/jma.html`, 2001.

[2] Johannes Braams. Babel, a multilingual package for use with LaTeX's standard document classes. `CTAN://macros/latex/required/babel/`, 2001.

[3] Euskaltzaindia. Data nola adierazi. `http://www.euskaltzaindia.net/arauak/dok/ProNor0037.htm`, 1995.