Managing languages within MlBibTeX

Jean-Michel Hufflen

Abstract

We explain how the information about natural languages used throughout documents is managed in MlBibTeX, our multilingual reimplementation of BibTeX. That allows us to show how the interface between MlBibTeX and LaTeX or ConTeXt's tools for multilinguism — e.g., the babel package — is organised, by means of a powerful data structure. We also show how the generated texts for LaTeX are built. In fact, they take as much advantage as possible of the multilingual packages of LaTeX's recent versions.

Keywords MlBibTeX, multilingual features, multilingual LaTeX packages, ConTeXt, tries, multilingual method, Scheme.

1 Introduction

The bibliography of a printed document, that is, the list of its bibliographical *references*, can be prepared manually, in which case its items may not be directly reusable elsewhere. The layout of bibliographies is ruled by *styles* that are influenced by cultural background. As a consequence, it can vary from a document to another: for example, the bibliography of some documents[1] use *plain* styles where items are labelled with numbers, some use *alpha* styles based on keys built from authors' last names and publication's years, e.g., '[Robeson 1965]' or '[Rob65]' — see [34, § 13.5.1] for a survey of available styles and corresponding layouts. In addition, some information may depend on the printed document's language: let us consider the date of a publication, a publisher may require that month names are printed in English for the bibliography of a document written in English, in French (resp. German, ...) for a document written in French (resp. German, ...). So managing bibliographical references already typeset for a particular document is tedious, and it is better for such references to be automatically generated from a database containing bibliographical *entries*.[2] In particular, this allows us to put as much information as we want within entries, even if some parts of information do not appear within generated texts.

As an accurate example, the bibliography program BibTeX [36] is often used to build 'References' sections for documents suitable for LaTeX [34, § 12.1.3]. BibTeX searches bibliography (`.bib`) files for keys cited throughout a document: to do that, it uses information put in *auxiliary* (`.aux`) files produced by LaTeX [34, Fig. 12.1]. BibTeX's bibliography styles[3] are programmed using a stack-based language [34, § 13.6]. By means of such a bibliography program, we should be able to fill in all the fields of a bibliographical entry once, and derive as many references as we want, according to layouts expressed by bibliography styles. This is true in most cases, but not always, depending on the expressive power of bibliography styles. For example, let us consider *annotated* bibliographies: the annotations should be expressed in the document's language. If we wish to avoid the duplication of bibliographical entries according to the language of an added annotation, such annotations can be given different field names:

```
english-ANNOTE = ..., french-ANNOTE = ...,
...
```

but in this case, we have to generate several bibliography styles differing only by the name of the chosen annotation. This example shows that BibTeX was not ideally designed for multilingual applications. There have been some attempts to insert multilingual features into texts generated by BibTeX — e.g., in the jurabib package [34, pp. 733–735] and the `custom-bib` tool (usable by applying LaTeX to the `makebst.tex` program) [34, § 13.5.2] — but BibTeX itself does not take enough advantage of multilingual features of LaTeX's recent versions. In addition, we think that the language BibTeX uses for bibliography styles leads to non-modular programs, monolithic and hard to maintain, as we explained in [17].

MlBibTeX aims to ease the development of multilingual bibliographies, without giving any privilege to a particular language, as the babel package does for documents written with LaTeX's modern versions [34, Ch. 9]. MlBibTeX's current version (1.3), described in [18] and developed in Scheme [25], is usable to generate bibliographies for LaTeX documents. This bibliography processor also opens a window towards the world of XML,[4] which has become a central formalism for document interchange. Since parsing a `.bib` file results in a tree that can be viewed as an XML tree, this choice more easily allows us to build other output files than `thebibliography` environments for LaTeX [34, § 12.1.2]. In particular, we can generate (X)HTML[5] pages for bibliographies to be

[1] This article, for example.

[2] Within MlBibTeX ('**M**ulti**L**ingual BibTeX'), we use precise terminology: bibliographical *entries* are specified in bibliography (`.bib`) files, and bibliographical *references* — in `.bbl` files for use with LaTeX — are what a word processor typesets.

[3] A representative selection of bibliography styles usable with BibTeX is given in [34, Table 13.4].

[4] E**X**tensible **M**arkup **L**anguage. Readers interested in an introduction to this metalanguage can refer to [39].

[5] (E**X**tensible) **H**yper**T**ext **M**arkup **L**anguage. XHTML is a reformulation of HTML using XML conventions. [35] is a good introduction to these languages.

```
@BOOK{robeson1965,
        AUTHOR = {first => Kenneth, last => Robeson},
        TITLE = {The Polar Treasure},
        PUBLISHER = {Bantam},
        SERIES = {Doc Savage},
        NUMBER = 4,
        NOTE = {[Titre de la traduction française : ‘‘Le trésor polaire’’] ! french
                [Titel der deutschen \"{U}bersetzung: ‘‘Das Wrack im Eis’’] ! german
                [T\’{i}tulo de la traduci\’{o}n al Espa\~{n}ol: ‘‘El tesoro del Polo’’] ! spanish}
        YEAR = 1965,
        MONTH = apr,
        LANGUAGE = english}
```

Figure 1: Example of an entry using MlBibTeX's features.

displayed on the *Web*, or XML files written according to the rules of XSL-FO[6] [43]. In fact, bibliography styles are now programmed using a new language, called nbst,[7] close to XSLT[8] [41], the language of transformations for XML documents.

We have already written some documents about MlBibTeX's implementation. In [19], we explain why we have started a new implementation using Scheme [25], after a first project in C [26]. We have also begun to describe the broad outlines of this implementation using Scheme in [22]. Here[9] we explain how the information about the natural languages used throughout bibliographies — and LaTeX documents — is organised. In next section, we show the drawbacks of deferring the generation of multilingual bibliographies to LaTeX. Then Section 3 exposes the notion of *language identifiers*, introduced in MlBibTeX. Section 4, explains how our data structure for handling language identifiers is built and how it allows us to generate multilingual bibliographies. We do not describe this data structure in Scheme directly, but using an abstract way, so that we can see that it could be implemented in any programming language.[10] Finally, we show that this data structure should be able to evolve for MlBibTeX's future versions.

We assume that readers are familiar with the multilingual babel package of LaTeX 2_ε, developed by Johannes Braams and described in [34, Ch. 9]. We also assume that readers can understand some simple macros, expressed using TeX's language [28, Ch. 20]. About BibTeX, XML, and Scheme, basic knowledge is sufficient to read this article, as well as basic notions about the use of trees in programming. Some notions related to specialised structures for searching strings are recalled in footnotes.

2 Difficulty related to languages

2.1 Accents and other diacritical signs

Let us consider the `robeson1965` entry given in Figure 1. It looks like a BibTeX entry, but some syntactic features indisputably belonging to MlBibTeX can be noticed: more user-friendly syntax for person names (`AUTHOR` and `EDITOR` fields), the use of multilingual switches (‘`[...] ! ...`’) within the value of the `NOTE` field. These notations are detailed in [18].

As mentioned in the introduction, such an entry is viewed as an XML tree in the sense that we can address its parts by using the XPath language [42]. As an example, Figure 2 gives the representation of the value associated with the `NOTE` field.[11] We can remark that quotations are uniformly expressed by using the American quotation marks (‘‘`...`’’) within a `.bib` file (see Figure 1), but each quotation is transformed into an XML element — an occurrence of the `emph` element with accurate attributes[12] — so putting quotation marks belonging to other languages is eased: ‘ « ... » ’ in French, ‘„...“ ’ in German, etc. More exactly, bibliography stylesheets are in charge of this. Likewise, we can remark that some accented letters can be typed directly by end-users (see the group expressed in the French language in Figure 1)

[6] E**X**tensible **S**tylesheet **L**anguage — **F**ormatting **O**bjects: this language aims to describe high-quality print outputs. There are some XSL-FO *processors*, in order to get printable files; an example from the TeX world is Passive TeX [37, p. 180]: it gets `.dvi` or `.pdf` files. An introductory reference to XSL-FO suitable for LaTeX users is [24].

[7] **N**ew **B**ibliography **ST**yles.

[8] E**X**tensible **S**tylesheet **L**anguage **T**ransformations.

[9] The present article is a renewed and updated version of previous material. A first version was initially designed for the PracTeX 2005 conference. Later, this presentation was given at a conference of the German-speaking TeX user group — at Berlin, in 2006 — and was entitled *Sprachen in MlBibTeX*.

[10] Besides, the implementation used within our preliminary project in C [19] was quite close to the current one.

[11] In fact, MlBibTeX internally uses the conventions of SXML (Scheme implementation of XML) [27]. See [22] for more details.

[12] Readers interested in a description of elements and attributes used throughout the XML versions of `.bib` files can refer to [16]: that is an earlier version, but changes are slight.

```
<note>
  <group language="french">
    Titre de la traduction française :
    <emph emf="no" quotedbf="yes">
      Le trésor polaire
    </emph>
  </group>
  <group language="german">
    Titel der deutschen Übersetzung:
    <emph emf="no" quotedbf="yes">
      Das Wrack im Eis
    </emph>
  </group>
  <group language="spanish">
    Título de la traducción al Español:
    <emph emf="no" quotedbf="yes">
      El tesoro del Polo
    </emph>
  </group>
</note>
```

Figure 2: Multilingual note as an XML tree.

or by using TEX commands (see the group written in Spanish), but as shown in Figure 2, the characters resulting from these commands are directly put within text nodes.[13]

Letters with accents and other diacritical signs illustrate some deficiency of the information put into `.aux` files. As a consequence, we have to parse the preamble of `.tex` source files[14] to get this information. When we generate texts, we cannot know if we can insert such letters directly, or if we have to write TEX commands to produce them. This last solution works in any case, provided that such commands belong to LATEX's basic set. That is true for commands that produce most accents (acute, grave, circumflex, ...) but there is a simple counter-example: the French *guillemets*.

> Concerto comique n° 25 en sol mineur « Les Sauvages et la Furstenberg ».

Such a French title may be mentioned within a book written in any language. In other words, we may have to write French guillemets even within a document written in English. The LATEX commands to produce them depend on the package used to write French fragments: either '`\og`' and '`\fg`' for the `frenchb` option[15] [5] of the babel package, or '`\guillemets`' and '`\endguillemets`' for the frenchpro package[16] [11]. There are other commands to get these guillemets, but they depend on other packages: if the fontenc package [34, § 7.5.3] has been loaded with the `OT1` option,[17] the commands `\guillemotleft` and `\guillemotright` produce opening and closing guillemets as single characters. If LATEX uses its default encoding (`OT1`), these commands are not provided and using them causes errors. However, getting French guillemets by combination of several characters is possible as shown in [21, Fig. 3]. Besides, a more immediate way exists if your keyboard allows you to type these guillemets, in which case using the inputenc package [34, § 7.5.2] with the `latin1` option allows LATEX to process them directly. If these characters are unavailable on your keyboard, you can use the sequences '`<<`' and '`>>`' with the `frenchb` option [5] of the babel package.[18]

These details may seem to be anecdotal, nevertheless they show how difficult the automatic generation of such texts is, especially if we wish to generate 'nice' texts, that is, readable by human agents. An end-user can solve this problem by typing LATEX commands directly within `.bib` files, but as a consequence, such bibliography files become difficult to be shared among several users, unless they make sure that the same packages with compatible options will be loaded when texts are processed. In addition, some files can be unusable for building other output files than those suitable for LATEX:[19] in particular, this point can obstruct the generation of bibliographies suitable for ConTEXt, another format built on TEX, created by Hans Hagen [14].

2.2 Using advanced LATEX commands

Let us consider again the value of the `NOTE` field within the `robeson1965` entry of Figure 1. This multilingual text could be transformed for use with LATEX by means of the `\iflanguage` command of the babel package [34, § 9.2.1] as follows:

```
NOTE ↦ \iflanguage{frenchb}{...}{%
         \iflanguage{germanb}{...}{%
          \iflanguage{spanish}{...}{}}}
```

[13] Although this behaviour only holds about the Latin 1 encoding (ISO-8859-1) presently. Future versions will probably extend it to the other encodings summarised in [13, Table C.4].

[14] That is, the commands located before the document itself, introduced by '`\begin{document}`'.

[15] This option has two aliases: `french` and `francais`. We will come back to this point later.

[16] This is a successor of the french package described in [7]. For reasons explained in [8, 9, 10], it has been replaced by a *freeware* version frenchle — 'french *al**LÉ**gé*' (for 'lightened') — [12] and a *shareware* version frenchpro — 'french **PRO**fessional' — [11]. The development of the freeware version seems to have stopped since B. Gaulle's death, in August 2007. Coming back to the French guillemets, the frenchle package provides only some compatibility with the commands '`\og`' and '`\fg`' of the babel package's `frenchb` option [12, § 7].

[17] That is, if the Cork encoding is used, with a range of 256 characters.

[18] ... or the frenchpro package [11].

[19] ... although MlBibTEX's next version will probably be able to solve this problem: cf. [20].

provided that this entry is cited only by documents written with the babel package, loaded with at least the options `frenchb`, `germanb`,[20] and `spanish`. However, if a user writes only in French by means of the frenchle package, this source text is unusable.[21] In addition, let us consider that we are building a bibliography for a document whose main language is French. Therefore, the bibliographical reference for `robeson1965` is surrounded as follows:

```
\bibitem[...]{robeson1965}
\begin{otherlanguage*}{english}Robeson
  (Kenneth)...
\end{otherlanguage*}
```

[34, § 9.2.1]. Besides, MlBibTeX offers a choice [18] between two kinds of bibliography styles:

- a *language-dependent* style, that is, each bibliographical item is expressed only in the entry's language,
- a *document-dependent* style, that is, each bibliographical item is expressed in the document's language, as far as possible.

In the first case, nothing need be done because the `robeson1965` entry characterises a book in English. In the second case, the French version of the `NOTE` field should be put and the previous text of this note should be surrounded as follows:

```
NOTE ↦ \begin{otherlanguage*}{frenchb}%
       \iflanguage{frenchb}{...}{%
        \iflanguage{germanb}{...}{%
         \iflanguage{spanish}{...}{}}}%
       \end{otherlanguage*}
```

Even if such texts are generated automatically, we can see that they are quite complicated.

Now let us consider the entry given in Figure 3: it concerns the English translation of a French book, so most information is given in English, except for the author's name and the original title, given in French. We wish these French fragments to be hyphenated correctly if need be, but if there is no way to typeset French fragments, we accept them to be typeset according to the rules of the language in use at this point. So we can write a robust version of the `\foreignlanguage` command provided by the babel package [34, § 9.1.2], here called `\putwrtlanguage`:[22]

[20] Like the babel package's option for French (cf. footnote 15), '`german`' is an alias and the option's actual name is `germanb`.

[21] In fact, frenchle may be used as an option of babel [12, § 6.5]. However, frenchle has been developed to be loaded as a package, in which case babel's commands are unknown.

[22] Let us recall that the commands giving access to languages are defined by natural numbers, thus we can use `\ifnum` to compare them.

```
@ARTICLE{ayerdhal2001,
       AUTHOR = {[last => Ayerdhal] : french}
       TITLE = {Flickering},
       JOURNAL = {Interzone},
       NUMBER = 167,
       PAGES = {6--13},
       NOTE = {English translation of
              ``[Scintillements] : french'', by
              Sheryl Curtis},
       YEAR = 2001,
       MONTH = may,
       LANGUAGE = english}
```

Figure 3: English translation of a French writer's book.

```
\def\putwrtlanguage#1#2{%
 \expandafter%
 \ifx\csname l@#1\endcsname\relax%
  \typeout{Language #1 unusable.}#2\else%
  \ifnum\csname l@#1\endcsname=\language%
   #2\else%
   \foreignlanguage{#1}{#2}%
  \fi%
 \fi}
```

This command could be used to process the two French fragments of the bibliographical reference for `ayerdhal2001`:

```
 \putwrtlanguage{frenchb}{Ayerdhal}
 \putwrtlanguage{frenchb}{Scintillements}
```

So this command is used twice when this bibliographical reference is processed. That is, checking whether the `frenchb` language is known is performed twice, although the answer is always the same. Either the `\l@frenchb` command is available for the whole of the document, or it is not at all. However, this replication does not result in great loss of efficiency: we can imagine that TeX can check a command's existence quickly. But in this first version, we assumed that the multilingual tool used was the babel package. If we take LaTeX's other multilingual packages into account — frenchle, german [38], ngerman and polski [4, § F.7] — our command looks like:[23]

```
\def\putwrtlanguage#1#2{%
 \@ifpackageloaded{babel}{\expandafter...
      ...          % (As previously.)
  }{\@ifpackageloaded{frenchle}{%
     \ifthenelse{\equal{#1}{french}}{%
      \french#2}{\english#2}}{%
```

[23] The frenchle package is not wholly multilingual in the sense that it deals with the French language, and can revert to LaTeX's original configuration — by means of the `\english` command [12, § 6.5] — in which case texts are supposed to be in English, as we do in the second version of the `\putwrtlanguage` command.

```
\@ifpackageloaded{german}{...}{%
 \@ifpackageloaded{ngerman}{...}{%
  \@ifpackageloaded{polski}{...}{%
   #2}}}}}
```

The waterfall of tests makes the command slower, and as many times as it is called, the corresponding results will be retrieved more and more slowly.[24]

These two examples show that implementing multilingual bibliographies by means of LaTeX commands only results in complicated texts. In addition, these texts are suitable for LaTeX only. If we wish to derive bibliographies for another word processor — e.g., ConTeXt — we have to put the same basic algorithms into action, but with the library of another language. So it seems to be better for such algorithms to be put into action by the bibliography processor itself.

3 Language identifiers

If we consider the results of working groups related to XML, natural languages throughout bibliographical data bases should be specified using the two-letter language, optionally followed by a two-letter country code,[25] described in [1] and [13, § C.1]. This convention allows the general reference to a language as well as a more precise reference to a local variant of it. For example, '`en`' is for the English language in general, whereas '`en-UK`' (resp. '`en-US`') is for British (resp. American) English only. In particular, using this convention would simplify an interface with the ConTeXt format, which also uses these codes. For example, ConTeXt uses the statement '`\language[fr]`' to change the document's current language into French [14, Ch. 7]. When MlBibTeX's first version was designed [15], it aimed at being a 'better BibTeX', mainly usable in cooperation with LaTeX; we did not relate this to XML features. Besides, we knew that many users of BibTeX put LaTeX commands within values of BibTeX fields. For example, it seemed to be interesting to process differently the texts written in French by using the successors of the frenchle package and those using the `frenchb` option of the babel package. The compromise we have settled is:

- a *language identifier* of MlBibTeX is a non-ambiguous prefix of:
  - either an option of the babel package,
  - or a multilingual *ad hoc* package;

  the multilingual *ad hoc* packages we recognise are frenchle, german, ngerman, and polski;
- by 'non-ambiguous', we mean that a language identifier can denote several ways to get access to the *same* language.

  As examples:[26]
- '`po`' is ambiguous because that it may start 'Polish' or 'Portuguese', two different languages;
- '`frenchb`' is a language identifier that gets access to only the `frenchb` option of the babel package;
- "`fr`' and `fre`' are not ambiguous and get access to either babel's option or the frenchle package. The '`french`' identifier has the same property. Since it can get access to the `frenchb` option of babel, do not confuse this feature with aliases handled by the babel package. The language definition file for French is `frenchb.ldf`,[27] but this option may be loaded by '`frenchb`', '`french`' or '`francais`' (see footnote 15). This last identifier is unusable with MlBibTeX, because it only recognises the names of the `.ldf` files located at babel's directory.[28]

4 Implementation issues

4.1 Implementing language identifiers

The language identifiers handled by MlBibTeX obviously form a dictionary. As we show in the previous section, we have to look into this dictionary not only for complete language identifiers but also for non-ambiguous prefixes. So this dictionary's implementation must be efficient. *Tries*[29] are the best implementation to put into action such information retrieval. Such a trie implementing our dictionary is pictured in Figure 4. The root is an array indexed by all the letters of the alphabet. Each component is either a null pointer, in which case the word does not exist within the dictionary, or an access to another letter-indexed array if there are words beginning with the recognised prefix, or a pointer to a resource if a

[24] Also, any TeXnician will have noticed that this new version requires the ifthen package [34, § A.3.2].

[25] For a fragment of a document, such codes are used by the predefined `xml:lang` attribute [39, p. 276]. Within DocBook documents, this attribute is named `lang` [45, p. 81].

[26] In the following we assume that the available packages and options are those of TeX Live 2008.

[27] '`.ldf`' is for '**L**anguage **D**efinition **F**ile', see [34, § 9.5.3].

[28] In the directory `.../texmf-dist/tex/generic/babel` if we consider the TeX Live implementation.

[29] Here is some terminology about trees implementing dictionaries:

- a **digital tree** is a tree for storing strings in which nodes are organised by substrings common to two or more strings;
- a **trie** is a particular case of a digital tree: there is only one node for every common prefix.

The name 'trie' originates from the central letters of the word 're**TRIE**val' [6]. A good but old-fashioned description of this structure has been given by Donald E. Knuth: cf. [31, § 6.3] & [30, Ch. 6, §§ 17–31]. Tries are used within TeX's program [29, §§ 920–924].

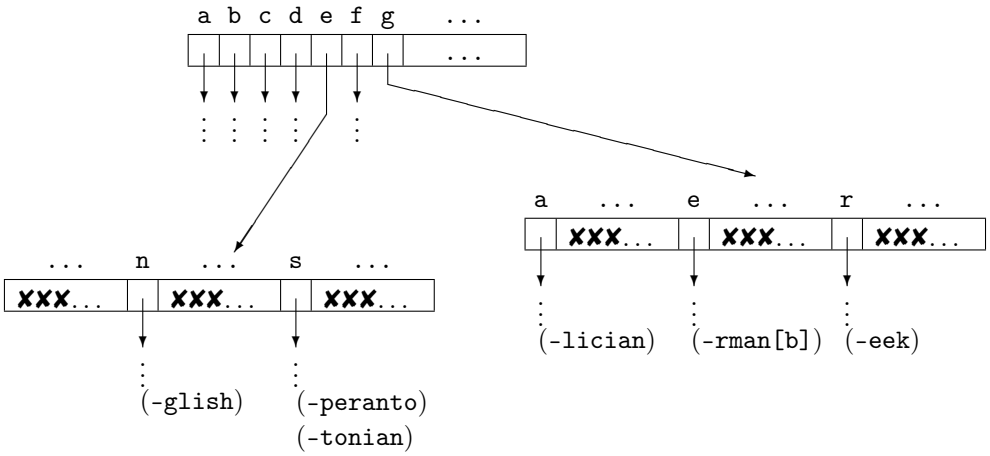


Figure 4: Searching for language identifiers by means of a trie.

word's end has been reached. In the trie in Figure 4, we see that the only language identifiers beginning with 'e' are those whose second letter is 'n' (for 'english') or 's' (for 'estonian'). Likewise, we can see that the language identifiers beginning with 'g' are the non-ambiguous prefixes of `galician`, `german[b]` and `greek`.

As the authors of [2] noticed, such an implementation by means of an array can be very space-consuming since there is many empty locations in the arrays of a trie. That is particularly true in our case, since there is only a few words denoting natural languages' names, in comparison with the whole of a dictionary for a complete language. So we decided to implement such tries by **ternary search trees**,[30] as shown in Figure 5, where the trie of MlBibTeX's language identifiers is sketched. Such a ternary search tree either is a leaf, or has three branches. Left and right branches — pictured in Figure 5 by a double-headed arrow — give access to letters less and greater than the current one. A middle branch gets access to the following letter of a word. A boxed character[31] means that this character comes last in the shortest non-ambiguous prefix of a language identifier.

At MlBibTeX's installation, we consider the *ad hoc* packages' names and the `.ldf` files located in the babel's package directory. These names are used to build a *height-balanced* ternary search tree.[32] In our case, this property means that if we are located at any node within our ternary search tree, the numbers of the letters left and right to the current one differ by one at most.

4.2 Multilingual method information

When MlBibTeX searches language identifier' trie for a non-existing or ambiguous identifier, the result is `#f`, the 'false' value in Scheme [25, § 6.3.1]. Otherwise, the result is a linear list whose elements — called **multilingual methods** w.r.t. MlBibTeX's terminology — are organised this way:

(⟨*marker*⟩ ⟨*opening*⟩ . ⟨*closing*⟩)

where:

⟨*marker*⟩ specifies a method used to switch to the language denoted by the identifier, e.g., an option of the babel package or an *ad-hoc* package;

⟨*opening*⟩ is a thunk[33] that results in a string put before a fragment written in the corresponding natural language;

⟨*closing*⟩ the same, but the string result is put after a fragment in the corresponding language.

Figure 6 shows how the language identifiers for French allow us to get access to the different ways to surround a fragment written in French. Given a character within our trie, a dashed arrows points to the result of the function searching for a string ending with this character.[34] There exist *default* multilingual methods, e.g.:

[30] Another solution could have used a compact implementation of *maps*, as provided by Python [32, pp. 49–51]. Often *hash tables* — associating *keys* with *values* by means of a *hash code* — are used for such search. They are directly provided by Common Lisp [40], Perl [44, Ch. 2], and Ruby [33], but are not as efficient as ternary search trees, as shown in [3].

[31] The '`#\...`' notation for a character comes from Scheme [25, § 6.3.4].

[32] The **height** of a tree is the maximum distance of any leaf from the root of a tree. In a **height-balanced** ternary tree, left and right branches differ in height by no more than one and each of these two branches is recursively height-balanced, too. Searching balanced trees is more efficient on average. See [31] for more details about this notion.

[33] In functional programming, this word denotes a zero-argument function.

[34] In fact, the actual implementation — more efficient — is slightly different, due to some advanced features of Scheme. But our functions behave exactly as shown in Figure 6.

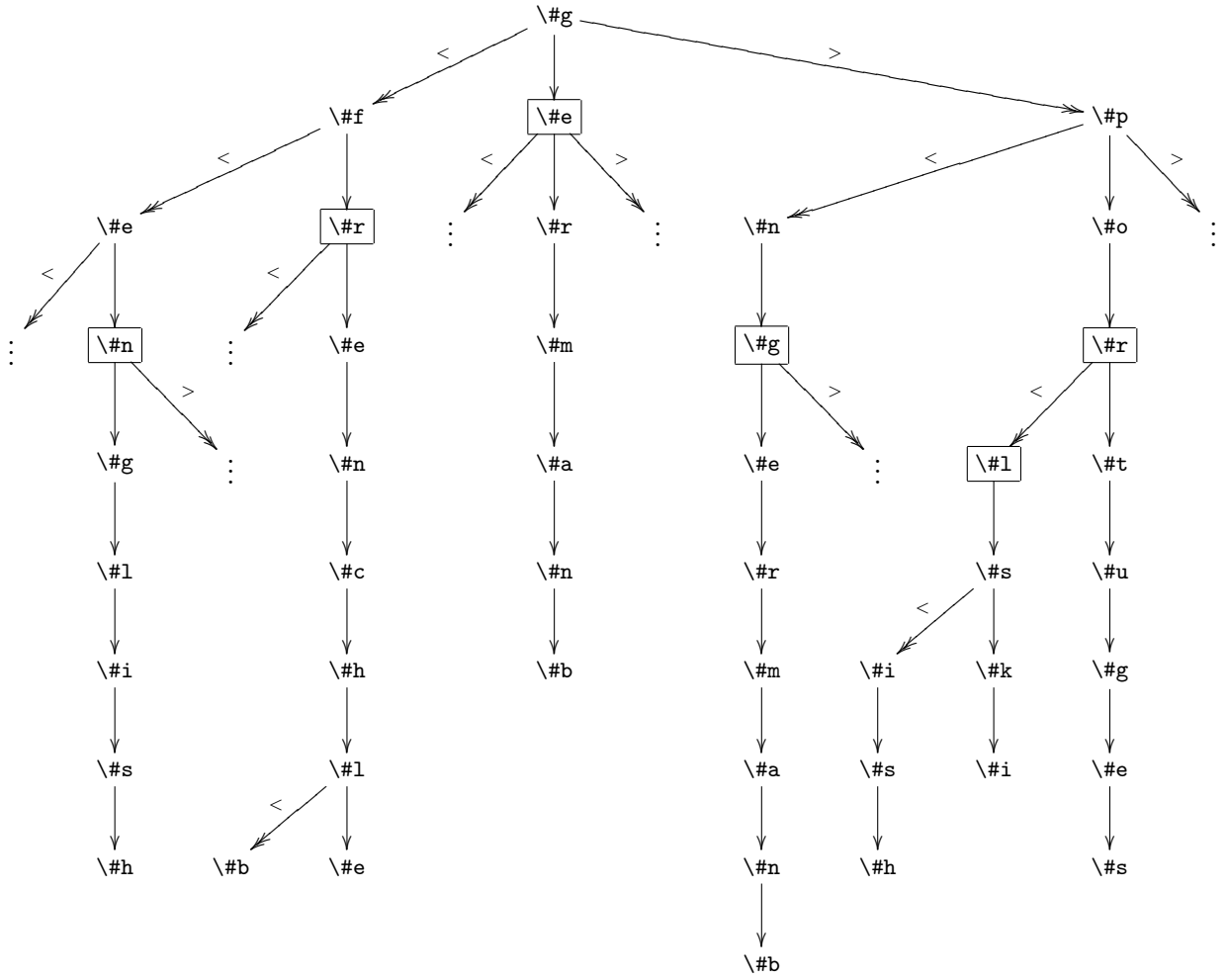


Figure 5: Implementing a trie by means of a ternary search tree.

```
((*frenchle*) (lambda () "{\english") .
              (lambda () "}"))
```

being used for natural languages other than French if the frenchle package has been loaded when the document is processed.[35]

In Figure 6, we can remark that the approach used in ConTEXt is included in such lists, except if the language identifier gets access to one method suitable for LATEX even though other methods for the same language exist. So the identifiers `fr`, `fre`, ... `french` can be used when a bibliography for ConTEXt is derived, but neither `frenchb` nor `frenchle`.

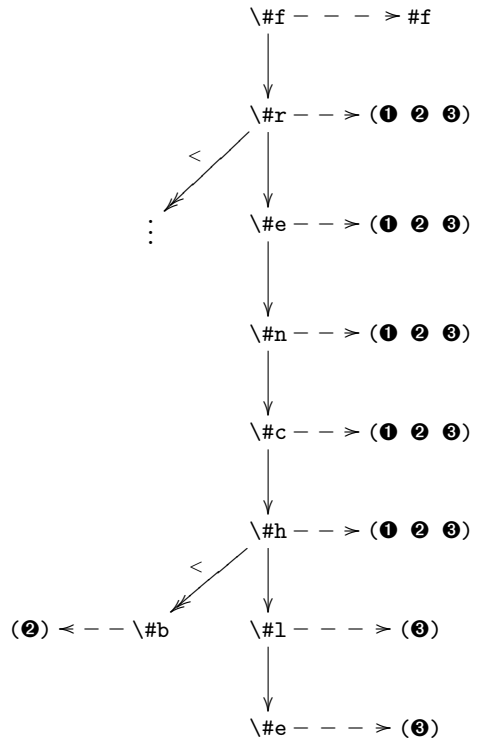
5 Conclusion

This article is an introduction to MlBibTEX's implementation core. We have tried to be precise as far as possible and avoid low-level details. Our goal was to show our realisation as a compromise between user-friendliness and a high-performing implementation. At the time of writing, the available backends are LATEX and ConTEXt. As we wrote in [23], 'when we began [our adaptation of MlBibTEX to ConTEXt] (...), we were afraid we would have to reprogram some important parts of MlBibTEX'. As shown by our examples, the management of language identifiers did not need a major revision when we integrated a backend for ConTEXt. So we think that our implementation is robust. Other adaptations to other backends — e.g., for (X)HTML — should confirm that. We are confident.

6 Acknowledgements

After discussion with some people, I realised that tries were not very well-known. Implementing them is not a small exercise, but is very worthwhile... and actually useful within natural language processing. I

[35] But the language identifier *must* be recognised. If not, any multilingual method, even a default one, cannot be used.



where:

```
❶ ≡ ((*context*) (lambda () "{\language[fr]") .
                 (lambda () "}"))
❷ ≡ ((*babel-option*)
     (lambda ()
       (if ((l-available-languages 'check?) 'frenchb)
           "\begin{otherlanguage*}{frenchb}"
           "")) .
     (lambda ()
       (if ((l-available-languages 'check?) 'frenchb)
           "\end{otherlanguage*}"
           "")))
❸ ≡ ((*frenchle*) (lambda () "{\french ") .
                  (lambda () "}"))
```

Figure 6: Multilingual methods associated with a language identifier.

hope that this article will contribute to demystifying this structure. Thanks to Karl Berry and Barbara Beeton, who kindly proofread this article.

References

[1] Harald Tveit Alvestrand: *Request for Comments: 1766. Tags for the Identification of Languages*. UNINETT, Network Working Group. March 1995. http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1766.html.

[2] Jun-Ichi Aoe, Katsuhi Morimoto and Takashi Sato: "An Efficient Implementation of Trie Structures". *Software — Practice and Experience*, Vol. 22, no. 9, pp. 695–721. September 1992.

[3] Jon L. Bentley and Robert Sedgewick: "Algorithms for Sorting and Searching Strings". In: *Proc. 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 360–369. January 1997.

[4] Antoni Diller: *LaTeX wiersz po wierszu*. Wydawnictwo Helio, Gliwice. Polish translation of *LaTeX Line by Line* with an additional annex by Jan Jelowicki. 2001.

[5] Daniel Flipo: *Documentation sur le module frenchb de babel*. Version 2.3c. February 2009. http://daniel.flipo.free.fr/frenchb/frenchb2-doc.pdf.

[6] Edward Fredkin: "Trie Memory". *Communications of the ACM*, Vol. 3, no. 9, pp. 490–499. September 1960.

[7] Bernard Gaulle : « Comment peut-on personnaliser l'extension french de LaTeX ? ». *Cahiers GUTenberg*, Vol. 28–29, p. 143–157. Actes de la x^e conférence TeX européenne. Saint-Malo, France. Mars 1998.

[8] Bernard Gaulle : « À propos de french ». *Lettre GUTenberg*, Vol. 15, p. 16–17. Juillet 1999.

[9] Bernard Gaulle : « Nouvelles french ». *Lettre GUTenberg*, Vol. 16, p. 11–12. Décembre 1999.

[10] Bernard Gaulle : « À propos de french ». *La lettre GUTenberg*, Vol. 20, p. 5–7. Octobre 2001.

[11] Bernard Gaulle : *Notice d'utilisation de l'extension frenchpro pour LaTeX. Version V5,995*. Avril 2005. http://www.frenchpro6.com/frenchpro/french/ALIRE.pdf.

[12] Bernard Gaulle : *L'extension frenchle pour LaTeX. Notice d'utilisation. Version V5,9993*. Février 2007. http://www.tug.org/texlive/Contents/live/texmf-dist/doc/latex/frenchle/frenchle.pdf.

[13] Michel Goossens and Sebastian Rahtz, with Eitan M. Gurari, Ross Moore and Robert S. Sutor: *The LaTeX Web Companion*. Addison-Wesley Longmann, Inc., Reading, Massachusetts. May 1999.

[14] Hans Hagen: *ConTeXt, the Manual*. November 2001. http://www.pragma-ade.com/general/manuals/cont-enp.pdf.

[15] Jean-Michel Hufflen: "MlBibTeX: a New Implementation of BibTeX". In: Simon Pepping, ed., *EuroTeX 2001*, pp. 74–94. Kerkrade, The Netherlands. September 2001.

[16] Jean-Michel Hufflen: "Multilingual Features for Bibliography Programs: From XML to MlBibTeX". In: *EuroTeX 2002*, pp. 46–59. Bachotek, Poland. April 2002.

[17] Jean-Michel Hufflen: "European Bibliography Styles and MlBibTeX". *TUGboat*, Vol. 24, no. 3, pp. 489–498. EuroTeX 2003, Brest, France. June 2003.

[18] Jean-Michel Hufflen: "MlBibTeX's Version 1.3". *TUGboat*, Vol. 24, no. 2, pp. 249–262. July 2003.

[19] Jean-Michel Hufflen: "A Tour around MlBibTeX and Its Implementation(s)". *Biuletyn GUST*, Vol. 20, pp. 21–28. In *BachoTeX 2004 conference*. April 2004.

[20] Jean-Michel Hufflen: "MlBibTeX: beyond LaTeX". In: Apostolos Syropoulos, Karl Berry, Yannis Haralambous, Baden Hugues, Steven Peter and John Plaice, eds., *International Conference on TeX, XML, and Digital Typography*, Vol. 3130 of *LNCS*, pp. 203–215. Springer, Xanthi, Greece. August 2004.

[21] Jean-Michel Hufflen: "Making MlBibTeX Fit for a Particular Language. Example of the Polish Language". *Biuletyn GUST*, Vol. 21, pp. 14–26. 2004.

[22] Jean-Michel Hufflen: "MlBibTeX in Scheme (*First Part*)". *Biuletyn GUST*, Vol. 22, pp. 17–22. In *BachoTeX 2005 conference*. April 2005.

[23] Jean-Michel Hufflen: "MlBibTeX Meets ConTeXt". *TUGboat*, Vol. 27, no. 1, pp. 76–82. EuroTeX 2006 proceedings, Debrecen, Hungary. July 2006.

[24] Jean-Michel Hufflen: "Introducing LaTeX users to XSL-FO". *TUGboat*, Vol. 29, no. 1, pp. 118–124. EuroBachoTeX 2007 proceedings. 2007.

[25] Richard Kelsey, William D. Clinger, Jonathan A. Rees, Harold Abelson, Norman I. Adams iv, David H. Bartley, Gary Brooks, R. Kent Dybvig, Daniel P. Friedman, Robert Halstead, Chris Hanson, Christopher T. Haynes, Eugene Edmund Kohlbecker, Jr, Donald Oxley, Kent M. Pitman, Guillermo J. Rozas, Guy Lewis Steele, Jr, Gerald Jay Sussman and Mitchell Wand: "Revised^5 Report on the Algorithmic Language Scheme". *HOSC*, Vol. 11, no. 1, pp. 7–105. August 1998.

[26] Brian W. Kernighan and Denis M. Ritchie: *The C Programming Language*. 2nd edition. Prentice Hall. 1988.

[27] Oleg E. Kiselyov and Kirill Lisovsky: "XML, XPath, XSLT Implementations as SXML, SXPath, and SXSLT". In: *International Lisp Conference 2002*. San Francisco, California. October 2002.

[28] Donald Ervin Knuth: *Computers & Typesetting. Vol. A: The TeXbook*. Addison-Wesley Publishing Company, Reading, Massachusetts. 1984.

[29] Donald Ervin Knuth: *Computers & Typesetting. Vol. B: The Program*. Addison-Wesley Publishing Company, Reading, Massachusetts. 1986.

[30] Donald Ervin Knuth: *Literate Programming*. No. 27 in Lecture Notes. Center for the Study of Language of Information. 1992.

[31] Donald Ervin Knuth: *Sorting and Searching*, Vol. 3 of *The Art of Comuter Programming*. 2nd edition. Addison-Wesley, Reading, Massachusetts. 1998.

[32] Alex Martelli: *Python in a Nutshell*. O'Reilly. March 2003.

[33] Yukihiro Matsumoto: *Ruby in a Nutshell*. O'Reilly. English translation by David L. Reynolds, Jr. November 2001.

[34] Frank Mittelbach and Michel Goossens, with Johannes Braams, David Carlisle, Chris A. Rowley, Christine Detig and Joachim Schrod: *The LaTeX Companion*. 2nd edition. Addison-Wesley Publishing Company, Reading, Massachusetts. August 2004.

[35] Chuck Musciano and Bill Kennedy: *HTML & XHTML: The Definitive Guide*. 5th edition. O'Reilly & Associates, Inc. August 2002.

[36] Oren Patashnik: *BibTeXing*. February 1988. Part of the BibTeX distribution.

[37] Dave Pawson: *XSL-FO*. O'Reilly & Associates, Inc. August 2002.

[38] Bernd Raichle: *Die Makropakete „german" und „ngerman" für LaTeX 2ε, LaTeX 2.09, Plain-TeX and andere darauf Basierende Formate*. Version 2.5. Juli 1998. Im Software LaTeX.

[39] Erik T. Ray: *Learning XML*. O'Reilly & Associates, Inc. January 2001.

[40] Guy Lewis Steele, Jr., Scott E. Fahlman, Richard P. Gabriel, David A. Moon, Daniel L. Weinreb, Daniel Gureasko Bobrow, Linda G. DeMichiel, Sonya E. Keene, Gregor Kiczales, Crispin Perdue, Kent M. Pitman, Richard Waters and Jon L White: *Common Lisp. The Language. Second Edition*. Digital Press. 1990.

[41] W3C: *XML Names*. W3C document. Edited by Tim Bray, Dave Hollander, and Andrew Layman. January 1999. `http://www.w3.org/TR/1999/REC-xml-names-19990114/`.

[42] W3C: *XSL Transformations (XSLT). Version 1.0*. W3C Recommendation. Edited by James Clark. November 1999. `http://www.w3.org/TR/1999/REC-xslt-19991116`.

[43] W3C: *Extensible Stylesheet Language (XSL). Version 1.0*. W3C Recommendation. Edited by James Clark. October 2001. `http://www.w3.org/TR/2001/REC-xsl-20011015/`.

[44] Larry Wall, Tom Christiansen and Jon Orwant: *Programming Perl*. 3rd edition. O'Reilly & Associates, Inc. July 2000.

[45] Norman Walsh and Leonard Muellner: *DocBook: The Definitive Guide*. O'Reilly & Associates, Inc. October 1999.