Beyond BibTeX

Jean-Michel Hufflen

Abstract

This article is related to the production of bibliographies for documents typeset by means of LaTeX or ConTeXt. We explore some recent ways to go beyond what is allowed by BibTeX. Then we explain why the programs `mlbiblatex` and `mlbibcontext` seem to us to be promising.

Keywords BibTeX, MlBibTeX, `mlbiblatex`, `mlbibcontext`, LaTeX, ConTeXt MkII, MkIV, LuaTeX, biblatex package, `bib` module.

Sommario

Questo articolo tratta la produzione di bibliografie di documenti composti con LaTeX e ConTeXt. Esploriamo alcune tecniche recenti che soppiantano quanto permesso da BibTeX. Quindi illustriamo perché i programmi `mlbiblatex` e `mlbibcontext` ci sembrano promettenti.

Parole chiave BibTeX, MlBibTeX, `mlbiblatex`, `mlbibcontext`, LaTeX, ConTeXt MkII, MkIV, LuaTeX, pacchetto biblatex, modulo `bib`.

Introduction

For many years, BibTeX (Patashnik, 1988a) was unrivalled as the bibliography processor associated with the LaTeX typesetting system. Let us recall that BibTeX extracts *bibliographical keys* from an auxiliary (.aux) file built by LaTeX: let *f* be a file name without suffix, the command '`bibtex f`' is equivalent to '`bibtex f.aux`'. This .aux file also provides a list of *bibliography database* (.bib) files to be searched. The corresponding *entries* are extracted, formatted according to a *bibliography style*, and the result is a *f*.bbl file of *bibliographical references*[1] for the *f*.tex document. Most of bibliography styles sort the relevant references according to the conventions they are designed to enforce. The ones that do not are *unsorted*, they just list these references according to the order of first citation throughout the document.

However, BibTeX is ageing: the current version has been in use for more than a decade. More precisely, its bibliography styles (.bst files) are written using an old-fashioned language using postfixed notations and based on manipulating a stack (Patashnik, 1988b); new requirements have appeared during the last decade (Hufflen, 2011)—multilingualism, encodings,[2] other typesetting systems built out of TeX, e.g., ConTeXt (Hagen, 2001)—beyond the capabilities of this venerable program.

Another approach has been recently proposed whereby references are stored into .bbl files as *structures*, and formatting 'References' sections is entirely deferred to the typesetting system. The LaTeX commands of the biblatex package[3] (Lehman, 2011) are in charge of such formatting. A similar framework had been put into action by Taco Hoekwater's `bib` module of ConTeXt (Hoekwater, 2001). Even if the biblatex package's functions may be used with BibTeX as bibliography processor—in which case the biblatex bibliography style is applied—another bibliography processor, more recent, is recommended: biber (Charette and Kime, 2011). This bibliography processor is a good replacement for BibTeX, although it is quite slow—it has been developed using Perl[4]—but can generate only references suitable for the biblatex package, not for ConTeXt's `bib` module.

In the next section, we explain why the difficulty of writing a new bibliography processor from scratch is mainly related to the format of bibliography database (.bib) files. Then we recall how the biblatex package and `bib` module work. In Section 3, we show that some features of MlBibTeX[5] may be of interest when we aim to process bibliographies for biblatex and ConTeXt. Let us recall that MlBibTeX is a reimplementation—written using the Scheme programming language (Kelsey et al., 1998)—of BibTeX with particular focus on multilingual features. It was presented at the GUTenberg 2004 meeting (Hufflen, 2005). Section 4 introduces our programs `mlbiblatex` and `mlbibcontext`, they are variants of MlBibTeX, suitable for biblatex and ConTeXt respectively. Finally we discuss some technical points.

1. This distinction between *entries* and *references* originates from MlBibTeX's terminology. We follow it throughout this article.

2. In fact, some slight extensions, built out of the source files of BibTeX—e.g., BibTeX8 (Mittelbach et al., 2004, § 13.1.1) and BibTeXu (Voß, 2011, § 4.3)—have been designed in order to deal with encodings. Those recognised by LaTeX are given in (Mittelbach et al., 2004, § 7.5.2).

3. An introduction to this package in Italian is (Pantieri, 2009).

4. **P**ractical **E**xtraction and **R**eport **L**anguage. A good introduction to this language is (Wall et al., 2000).

5. **M**ulti**L**ingual BibTeX.

```
@BOOK{pollotta2007,
        AUTHOR = {Nick Pollota},
        TITLE = {Neutron Force},
        SERIES = {Don Pendleton's \emph{Stony
                  Man}},
        NUMBER = 89,
        PUBLISHER = {Gold Eagle},
        TOTALPAGES = 352,
        YEAR = 2007,
        MONTH = jun}
```

FIGURE 1: Example using BIBTEX's format.

1 Subtleties of the .bib format

Many LATEX end-users have a *huge* number of .bib files encoded in the format defined by BIBTEX (Mittelbach et al., 2004, § 13.2). In addition, many organisations are making .bib files available on the Web, e.g., ACM[6]'s digital library,[7] IEEE[8] Xplore's digital library,[9] or CiteSeerX.[10] So a new bibliography processor designed to work in conjunction with LATEX should be able to deal with this .bib format. At first glance, it is not very complicated, all the metadata of a bibliographical entry given using the syntax '`KEY = value`', as you can see in Fig. 1. In reality, this format is more subtle. For example, values may be surrounded by double quotes:

```
TITLE = "Il nome della rosa"
```

in which case a double quote character used within such a value must be surrounded by braces:

```
TITLE = "Die Energiej{\"a}ger"
```

Values may also be surrounded by braces:

```
TITLE = {Grande Jonction}
```

in which case a double quote character can be used alone within such a value:

```
TITLE = {Murcos Verm\"achtnis}
```

The syntax for person names—see (Hufflen, 2006) for more details—is accurate for simple cases, but may be surprising in such a case:

```
AUTHOR = {Jean {Le Clerc de la Herverie}}
```

(removing the braces surrounding 'Le Clerc de la Herverie' causes 'Herverie' to be interpreted as the last name, 'Jean Le Clerc' as the first name, and 'de la' as a particle). An additional point comes from the widespread habit of inserting LATEX commands inside value parts of BIBTEX fields, like the following snippet:

6. **A**ssociation for **C**omputing **M**achinery.
7. See `http://dl.acm.org`.
8. **I**nstitute of **E**lectrical and **E**lectronics **E**nginners.
9. See `http://ieeexplore.ieee.org/Xplore`.
10. See `http://citeseerx.ist.psu.edu`.

```
\documentclass{article}

\usepackage[%
 bibstyle=numeric,citestyle=authoryear]{%
 biblatex}
\addbibresource{sm.bib}
%%  You must put the ".bib" suffix.

\begin{document}

Do you know \parencite{pollotta2007}? This is
an interesting thriller written by
\citeauthor{pollotta2007} and came out in
\citeyear{pollotta2007}.

\printbibliography

\end{document}
```

FIGURE 2: Example using the biblatex package.

```
TITLE = {\em Babylon Babies}
```

That complicates the generation of Web output.[11] Moreover, the following declaration:

```
TITLE = {\emph{Cosmos Incorporated}}
```

yields a title's specification which would be correctly interpreted by LATEX, but ConTEXt would not recognise the `\emph` command.

In other words, it is quite easy to transform the syntax '`KEY = value`' into '`<KEY>value</KEY>`' if we adopt XML[12]-like syntax, or '`(KEY value)`' if some Lisp[13]-like syntax is preferred. On the contrary, destructuring fields' values may be more complicated. That is why you can find many converters from .bib files into other formats, but at the first level. Roughly speaking, only few of these programs run the risk of analysing the contents of fields' values.

2 Deferring reference-formatting

2.1 The biblatex package

Let us consider the example of a LATEX document given in Fig. 2 and citing the bibliographical entry given in Fig. 1. This document uses the biblatex package. As aforementioned, one bibliography style—biblatex—is associated with this package, so the `\bibliographystyle` command is not to be used. The `\bibliography` command, 'traditionally' used, has been replaced in the preamble by the `\addbibresource` command of biblatex (Lehman, 2011, § 3.5.1). Let us recall that

11. This example with the `\em` command may seem to be quite artificial. However, some LATEX commands are sometimes used as *workarounds* inside values denoting person names. A good example is given in (Mittelbach et al., 2004, p. 767).
12. e**X**tensible **M**arkup **L**angage.
13. **LIS**t **P**rocessor.

```
\entry{pollotta2007}{book}{}
 \name{author}{1}{}{%
  {{}{Pollota}{P.}{Nick}{N.}{}{}{}{}}}
 \list{publisher}{1}{{Gold Eagle}}
 \strng{namehash}{PN1}\strng{fullhash}{PN1}
 \field{labelyear}{2007}\field{sortinit}{P}
 \field{number}{89}
 \field{series}{%
  Don Pendleton's \emph{Stony Man}}
 \field{title}{Neutron Force}
 \field{month}{06}\field{year}{2007}
\endentry
```

(The TOTALPAGES field does not appear here because it is not known by biblatex.)

FIGURE 3: Reference used by the biblatex package.

```
\usemodule[bib]  %  Not needed for MkIV.
\setupbibtex[database=sm]
\setuppublications[numbering=yes]

\starttext

Did you read \cite[pollotta2007]?

\placepublications

\stoptext
```

FIGURE 4: Citations and bibliographies in ConTEXt.

the \bibliography command serves two purposes when BibTEX's 'standard' bibliography styles are used: it specifies both .bib files and the place where the bibliography is to be typeset. The new command \addbibresource just specifies a .bib file to be searched in order to build the bibliography, whereas inserting the 'References' section within the document is done by the \printbibliography command. The biblatex package provides as well commands to access the value of fields such as AUTHOR, TITLE, YEAR; for example, \citeauthor and \citeyear in Fig. 2 are rendered as follows:

> Do you know (Pollotta 2007)? This is an interesting thriller written by Pollotta and came out in 2007.

The biblatex package allows citations to be controlled precisely: you may number them or put the author-date system (Mittelbach et al., 2004, § 12.3) into action. You can do that globally by means of the style option, or separate the management of 'References' sections and the citations throughout the document, by means of the options bibstyle and citestyle, as we do in Fig. 2.

Many extensions have been introduced by biblatex. Numerous additional fields are recognised, for example, SUBTITLE, for a work's subtitle, in addition to its title (Lehman, 2011, § 2.2). Likewise, the grammar defining the PAGES field has been refined. Additional *entry types* can be handled, for example, @BOOKINBOOK, for items originally published as a standalone book and reprinted in collected works of an author (Lehman, 2011, § 2.1). In addition, you can use the standard fields YEAR and MONTH, or replace them by the DATE field, which allows the specification of *date ranges* (Lehman, 2011, § 2.3.8):

```
DATE = {2012-10-27/2012-10-28}
```

The specification of fields recognised by biblatex use *types*: for example, AUTHOR is a *name list*, SUBTITLE and TITLE are *literals*. Some types are described by means of regular expressions, e.g., the *date* type. This type information appears in .bbl files suitable for biblatex, as shown in Fig. 3.

2.2 biblatex with biber

Roughly speaking, if you use the biblatex package in conjunction with BibTEX, you go on using the latter for tasks it does not perform satisfactorily.[14] That is particularly true when entries are sorted since BibTEX's sort procedures have been designed to work only on 'pure' ASCII[15] texts.[16] In other words, accents and other diacritical signs are processed improperly. For this use case, biber becomes an interesting bibliography processor because it correctly sorts entries in encodings other than ASCII. It can be specified by the backend option of the biblatex package:

```
\usepackage[backend=biber]{biblatex}
```

This option causes a .bcf[17] configuration file—using XML-like syntax—to be built. Let *f* be a file name without suffix, the command '`biber f`' is equivalent to '`biber f.bcf`'.

The biblatex package's sorting option, usable with this backend, allows a bibliography to be sorted w.r.t. a *sorting scheme*:

```
\usepackage[backend=biber,sorting=nyt]{%
 biblatex}
```

This predefined sorting scheme nyt uses three successive sort keys, based on authors or editors' **N**ames, then **Y**ears of publication, and **T**itles. The \DeclareSortingScheme command (Lehman, 2011, § 4.5.1) allows the definition of a new sorting scheme.[18] In fact, the tools related to the biblatex package use additional fields for sorting (Lehman, 2011, § 2.2.3): the first pass is controlled by the

14. ... although some points have been improved with BibTEX8.
15. **A**merican **S**tandard **C**ode **I**nformation **I**nterchange.
16. Let us recall that BibTEX's first version came out in 1985.
17. **B**iblatex **C**ontrol **F**ile.
18. Such a definition is to be put in a document's preamble.

```
\startpublication[k=pollotta2007,t=book,
 a={{Pollota}},y=2007,n=1,s=Pol07]
\author[]{Nick}[N.]{}{Pollota}
\pubyear{2007}\title{Neutron Force}
\series{Don Pendleton's {\em Stony Man}}
\volume{89}\pubname{Gold Eagle}\month{6}
\stoppublication
```

(The TOTALPAGES is not known by ConTEXt bib module, either.)

FIGURE 5: Reference used by ConTEXt.

PRESORT field; by default, some fields only used for sorting—such as SORTNAME, SORTYEAR—take precedence over the corresponding fields for 'actual' information—that is, AUTHOR or EDITOR, YEAR.[19]

2.3 ConTEXt's bib module

Let us consider Fig. 4 as an example of a source text for ConTEXt using a bibliographical reference. This reference, as it should be produced by a bibliography processor, is given in Fig. 5. The bib module (Hoekwater, 2001) can be used with ConTEXt MkII (CON, 2012a), it has been reimplemented in ConTEXt MkIV (LuaTEX) by Hans Hagen (CON, 2012b).

3 What can MlBibTEX provide?

As aforementioned, some operations related to the .bib format are quite difficult; MlBibTEX does some easily, and this can be profitable when .bbl files are to be generated for biblatex or ConTEXt. An important point is the check operations performed by MlBibTEX, stricter than BibTEX's, what may be useful. Last but not at least, some syntactical enrichment ease .bib files' syntax, and this may be of interest for biblatex users.

3.1 Destructuring

As mentioned in the introduction, MlBibTEX has been first designed and developed as possible replacement for BibTEX (Hufflen, 2003). As part of this task, we put into action an analysis of the values associated with BibTEX fields, as deeply as possible. We have precisely designed an internal format for bibliographical items. In fact, when MlBibTEX's parser processes a .bib file, we can consider that it builds an XML tree of this file. More precisely, this program builds expressions according to the SXML[20] format (Kiselyov, 2005). For example, Fig. 1's entry is translated into the XML tree given in Fig. 6. We can see that the author's name has been split into these components. Likewise, some basic commands of TEX and

19. This *modus operandi* is not specific to biber, it is implemented within BibTEX's biblatex bibliography style.

20. **S**cheme implementation of **XML**.

```
<book id="pollotta2007" from="sm.bib">
  <author>
    <name>
      <personname>
        <first>Nick</first>
        <last>Pollota</last>
      </personname>
    </name>
  </author>
  <title>Neutron Force</title>
  <publisher>Gold Eagle</publisher>
  <number>89</number>
  <series>
    Don Pendleton's <emph>Stony Man</emph>
  </series>
  <totalpages>352</totalpages>
  <year>2007</year>
  <month><jun/></month>
</book>
```

(The from attribute of the book element is set to the base name of the .bib file originally containing this entry.)

FIGURE 6: Fig. 1's example given using XML syntax.

LATEX—e.g., \em or \emph—are recognised and replaced by XML tags.

3.2 Additional check

When BibTEX users begin to run MlBibTEX, the most surprising feature is that the latter performs a more precise analysis of .bib files. When a field name is not recognised, a warning message is emitted.[21] That *modus operandi* may be viewed as an advantage: for example—let us recall that EDITOR is optional within an entry of type @INPROCEEDINGS—if you inadvertently typed 'EDITORS = ...' instead of 'EDITOR␣ = ...' inside such an entry, MlBibTEX will warn you whereas BibTEX will silently ignore that field. This feature may also be viewed as a drawback: if you specify a MONTH field, the associated value *must* be a symbol among jan, feb, ..., dec. Otherwise, MlBibTEX stops with an error message. This convention may appear as too restrictive, but MlBibTEX can sort w.r.t. month names, whereas BibTEX or biber do not. To perform such an operation, month names must be recognised. Likewise, when years are to be sorted, MlBibTEX applies a numerical sort whereas BibTEX and biber sort years as strings, so the value associated with a YEAR field must be an integer;[22] otherwise, an error message is emitted. More precisely, the fields subject to additional check are:

- the standard fields AUTHOR, EDITOR, MONTH, PAGES, and YEAR;

21. ... but this is just a warning message; the corresponding information is not lost.

22. Negative values, for years BCE, are allowed.

```
(<italian? "Monteverdi" "Monteverdi")                                    ⟹ #f
(<italian? "Monteverdi" "Monteverdi" (lambda () #f) < 'uppercase-1st) ⟹ #f ; Default values.
(<italian? "Monteverdi" "Monteverdi" (lambda () 'ok))                    ⟹ ok ; Equal strings.
(<italian? "Monteverde" "Monteverdi")                                    ⟹ #t
(<italian? "Monteverde" "Monteverdi" (lambda () 'ok) >)                  ⟹ #f ; Descending order.
(<italian? "Monteverdi" "MonteVerdi" (lambda () 'ok))          ⟹ #f
(<italian? "Monteverdi" "MonteVerdi" (lambda () 'ok) < #f)     ⟹ ok ; Case-insensitive equality.
(<italian? "Monteverdi" "MonteVerdi" (lambda () 'ok) < 'lowercase-1st)⟹ #t ; Lowercase letters
                                                                         ; take precedence.
(<italian? "Monteverde" "Monteverdi"
           (lambda ()
             (<italian? "Giuseppe" "Claudio"
                        (lambda () (<arithmetical? 1567 1643 (lambda () ...)))))) ⟹ #f
```

FIGURE 7: Order relations handled by MlBIBTEX.

- the field `DAY`, used by numerous styles;[23]
- the fields `GENDER` and `TOTALPAGES`, used by some bibliography styles associated with the jurabib package, as mentioned in (Mittelbach et al., 2004, § 12.5.1);
- two special fields used by MlBIBTEX: the `LANGUAGE` field, giving an entry's language (Hufflen, 2003), and the `LASTSORTKEY` field, allowing us to sort entries whose all the other sort keys are equal (Hufflen, 2008).

3.3 Order relations

In (Hufflen, 2007), we showed how the lexicographic order relations handled by MlBIBTEX were built. These order relations—implemented by means of Scheme functions—are language-dependent. A very simple use of the `<italian?` order relation—for Italian words—to compare two strings is given by the first example of Fig. 7—'`#t`' (resp. '`#f`') stands for the 'true' (resp. 'false') value in Scheme. In reality, these functions are more powerful since they use optional arguments—controlling the behaviour—in addition to the two strings to be compared:

- the third is a *thunk*[24] that is called if the two strings are equal;
- the fourth is `<` (resp. `>`) for an ascending (resp. a descending) order;
- the fifth is `#f` for a case-insensitive comparison, `uppercase-1st` (resp. `lowercase-1st`) if uppercase (resp. lowercase) letters take precedence when two strings are different only by the case.

Fig. 7's second example shows the default values of these three additional arguments. By default, these functions implement *strict* order relations, that is, *irreflexive*, asymmetric, and transitive; as `<` for numbers. But if you would like such an order relation to be non-strict—that is, reflexive, asymmetric, and transitive—just bind the third argument to the thunk `(lambda () #t)`. The sixth example shows that our `<italian?` function defaults to a case-sensitive relation in which uppercase letters take precedence over lowercase ones, the seventh example shows how to proceed if you would like lowercase letters to take precedence. Finally, the last example shows how the third argument can be used to *chain* order relations: the idea is to sort person w.r.t. last names, then first names, birth dates, and eventually other information.[25] This feature—sketched in (Hufflen, 2008, § 4)—makes possible the implementation of multiple levels of sorting. More details about these order relations are given in (Hufflen, 2012b).

A sort key may refer to *optional* information, e.g., the month of an entry, not required as an entry's year is. MlBIBTEX addresses such problems by means of functions encompassing default values when we are looking for optional information. For example, the Scheme function returning a month's rank—from 1 to 12—works as follows; the expression:

```
(<month-position T default-value)
```

returns the month's rank if the *`T`* form contains month information, *`default-value`* if this information is not supplied. Within a chronological sort, just bind the second argument—*`default-value`*—to `0` (resp. `13`) if you would like the forms without this information to come before (resp. after) the forms where such value is defined.

23. For example, the styles '`apa...`', used by the American Psychology Association.

24. A zero-argument function, w.r.t. Scheme's terminology.

25. The `arithmetical?` function, used within this example, is analogous to our order relations, in the sense that its third argument is called if the two numbers given as first two arguments are equal. Otherwise it behaves like `<`.

```
@BOOK{cussler2004,
        AUTHOR = {Clive Eric Cussler,
                  abbr => C. with
                  first => Paul,
                  last => Kemprecos
        TITLE = {White Death},
        SERIES = {Numa Files},
        PUBLISHER = {Penguin Books},
        YEAR = 2004}
```

FIGURE 8: Extensions regognised by MlBibTEX.

3.4 Syntactical extensions

MlBibTEX's syntactical extensions about multilingualism have been explained in detail in (Hufflen, 2003). Even if the biblatex package includes some multilingual features and work with the babel package (Mittelbach et al., 2004, Ch. 9), it cannot use these features presently.[26] On the contrary, our extensions for authors' and editors' names can be directly usable by the biblatex package. In addition to BibTEX's conventions, *keywords* may be used to point to the four parts—*First*, *von*, *Last*, *Junior*—of a name, what may be very useful in such a case:

```
first => Jean,
last => Le Clerc de la Herverie
```

(the four keywords '`first =>`', '`von =>`', '`last =>`', '`junior =>`' are available, the order of appearance being irrelevant). The '`abbr =>`' keyword may be used when a first name is not abbreviated according to the standard way—that is, retaining only the first letter, following by a period—as shown in Fig. 8 (removing this keyword would cause the first name to be abbreviated by '`C. E.`') Let us mention that mixed specifications are allowed: the second name is given according to BibTEX's conventions, followed by the redefined abbreviation of the first name. If an organisation's name is used as an author or editor, you can use the keywords '`org =>`' for the name as it must be typeset and '`sortingkey =>`' for the key used for sorting:

```
org => \GuIT~2012,
sortingkey => GuIT 2012
```

It is well-known that co-authors are connected by means of the '␣`and`␣' keyword. As shown in Fig. 8, MlBibTEX also allows the specification of *collaborators*, by means of the '␣`with`␣' keyword.

26. A connection is planned for a near future. That should be quite easy for ConTEXt, in the sense that all the languages are available *a priori*—you do not have to put all the languages you use throughout a text as options of a module like the babel package (Mittelbach et al., 2004, Ch. 9).

4 MlBibTEX's variants

When the approach of biblatex and ConTEXt is used, a bibliography processor has just to provide successive structures whatever the bibliography style is, such a style is put into action by customising the command of LaTEX or ConTEXt producing the final bibliography. So the *program* building these structures does not need to be customised by end-users. We can use techniques of 'actual' programming in order to get more efficiency, rather than interpreting a language as BibTEX does for bibliography styles. When we programmed MlBibTEX's kernel, we succeeded in using techniques related to functional programming in Scheme, we got an efficient program by compiling our functions. We have wanted to extend these advantages to the complete process of generating references for biblatex and ConTEXt. That yields two bibliography processors out of MlBibTEX's kernel, written entirely in Scheme: `mlbiblatex` and `mlbibcontext`.

4.1 The `mlbiblatex` program

The `mlbiblatex` program generates .bbl files suitable for the biblatex package. You can run it as follows—the `.aux` suffix can be omitted—:

`mlbiblatex` *f*`[.aux]` *key-expr lg-code*

where:

f`.aux` is the auxiliary file where the information about bibliographical keys and database files has been stored;

key-expr gives successive sort keys, according to the pattern `(m | n | t | y)*`, where '`m`', '`n`', '`t`', '`y`' respectively stand for '**M**onth',[27] '**N**ame' (person name as an author or editor), '**T**itle', '**Y**ear'; all the other signs are ignored; there is no default order relation:[28] if no sign is recognised, the list of bibliographical items is left unsorted;

lg-code is the language's code to be used for sorting strings when person names and titles of works are compared; available values are `DE` for German, `EN` for English, `FR` for French, `IT` for Italian, `PO` for Polish; there is no default value.

See Fig. 3 for example.

A switch mechanism implemented by means of Scheme functions allows us to recognise the extensions introduced by biblatex only when our parser is running in a kind of '`mlbiblatex` mode'. For

27. ... an item without month information being ranked after an item with such.

28. The default order relation used by both BibTEX and biber would be specified by `ynt`. Let us recall that by default, these two programs do not use any information about month during the sort step.

example, new entry types of biblatex are processed. The same mechanism allows biblatex users to specify dates either by means of a `DATE` field, or by a `YEAR` field and optionally by a `MONTH` field and a `DAY` field, whereas the `DATE` field is not allowed in BibTEX standard styles.

4.2 The `mlbibcontext` program

The `mlbibcontext` program generates .bbl files suitable for ConTEXt. The corresponding command line looks like `mlbiblatex`'s:

```
mlbibcontext f[.aux] key-expr lg-code
```

and *f*`.aux`, *key-expr*, *lg-code* have the same meaning than in § 4.1. See Fig. 5 for an example. By the way, we can notice that the `\emph` command used in Fig. 1 and not recognised by ConTEXt has been replaced by the `\em` command.

Here also, another switch mechanism considers a new `@CONTEXTPREAMBLE` directive when a .bib file is parsed by the `mlbibcontext` program. This directive aims to replace the 'traditional' `@PREAMBLE` directive, often used to put definitions of new LaTeX commands (Mittelbach et al., 2004, § 13.2.4). In particular, the `@CONTEXTPREAMBLE` directive can be used to define in ConTEXt some missing LaTeX commands used inside .bib files.

4.3 The distribution

MlBibTEX's distribution is located at:

`http://disc.univ-fcomte.fr/home/~jmhufflen/texts/superreport/smlbibtex-1.3.tar.gz`

To install it easily, compile the source files using the bigloo (Serrano, 2010) Scheme compiler; the installation procedure (Hufflen, 2012a) uses the commands `configure` (Vaughn et al., 2000) and `make` (Loukides and Oram, 1996) of the GNU[29] toolchain; more details are given in (Hufflen, 2012a, § 4.2). The complete distribution's version number is given 'classically', that is, by means of number sequences. Versions of particular variants are labelled by geographical names; the current versions of `mlbiblatex` and `mlbibcontext` are 'Breskens versions', since they have been demonstrated at first at the EuroTEX 2012 conference, at Breskens, The Netherlands.

5 Discussion and conclusion

An important point to take note of is that the successors of BibTEX have introduced many often incompatible extensions.[30] Moreover, their design principles are different: biblatex has introduced new fields whereas MlBibTEX has extended the syntax of the existing ones. For example, biblatex uses a `COLLABORATOR` field for additional collaborators whereas MlBibTEX links them by means of the keyword '␣`with`␣', as mentioned in § 3.4 (see also Fig. 8). These two techniques have advantages and drawbacks, already discussed in (Hufflen, 2011, § 3). Anyway, we think that the .bib format should evolve to take into consideration how modern tools deal with it.

The programs `mlbiblatex` and `mlbibcontext` currently have rough interfaces that should be enhanced and perfected: for example, command lines only get access to ascending orders, whereas MlBibTEX's kernel can provide descending orders, as mentioned in § 3.3. These programs are work in progress and in a state that allows us to experiment which interface could be a better fit for biblatex and ConTEXt. That could lead to a new backend for the biblatex package:

```
\usepackage[backend=mlbiblatex,...]{%
 biblatex}
```

—and something equivalent for ConTEXt—other options could allow accurate information to be passed to MlBibTEX.

A limitation of MlBibTEX's current version is its capability to handle only two encodings, for input .bib files as well as output .bbl ones. More precisely, .bib files are supposed to be encoded in Latin 1. The characters not included in this encoding can be reached only by using TEX commands. About generated .bbl files, either MlBibTEX detects that the Latin 1 encoding is used by looking into the document's preamble,[31] in which case this encoding is used for the .bbl file produced; otherwise, this .bbl file is a pure ASCII file, all the accented letters being specified by means of TEX commands. Such behaviour is due to Scheme. MlBibTEX has been written using this language's fifth revision (Kelsey et al., 1998), not Unicode-compliant. Most of Scheme interpreters can deal with Latin 1, some—not all—accept other encodings, but in a non-portable way. Besides, we want our functions to be able to work on as many Scheme interpreters as possible. A new revision of Scheme is in progress[32] and will be Unicode-compliant, so a future version of MlBibTEX should be able to deal with other encodings such as Latin 2, UTF[33]-8, UTF-16, etc. When we started to program MlBibTEX, we planned for a future version the use of Unicode, so we organised our functions in order for this update not to require too deep changes. To end up and be honest, we already succeeded in adapting

29. Recursive acronym: **G**NU is **N**ot **U**NIX.

30. For example, some BibTEX bibliography styles use an additional field for the total number of a book's pages. Often this field is named `TOTALPAGES` (see Fig. 1)—e.g., within the jurabib bibliography style—but the tools related to biblatex know this information as `PAGETOTAL`.

31. BibTEX just reads .aux files and never reads a .tex file (Mittelbach et al., 2004, § 12.1.3), whereas `mlbibtex` may look into a document's preamble.

32. See the Web page `http://scheme-reports.org`.

33. **U**nicode **T**ransformation **F**ormat.

MlBibTEX to applications other than those initially planned, so we face the future with confidence.

Acknowledgements

Thanks to Gianluca Pignalberi for the Italian translation of the abstract and keywords. I am also grateful to the anonymous referee whose valuable suggestions allowed me to improve many points.

References

François Charette and Philip Kime. *biber. A Backend Bibliography Processor for biblatex. Version biber 0.9 (biblatex 1.6)*. `http://freefr.dl.sourceforge.net/project/biblatex-biber/biblatex-biber/development/documentation/biber.pdf`, August 2011.

Bibliographies in MkII. CONTEXTGARDEN, `http://wiki.contextgarden.net/Bibliography`, April 2012a.

Bibliographies in MkIV. CONTEXTGARDEN, `http://wiki.contextgarden.net/Bibliography_mkiv`, July 2012b.

Hans Hagen. *ConTEXt, the Manual*. `http://www.pragma-ade.com/general/manuals/cont-enp.pdf`, November 2001.

Taco Hoekwater. The bibliographic module for ConTEXt. In *EuroTEX 2001*, pages 61–73, Kerkrade (the Netherlands), September 2001.

Jean-Michel Hufflen. MlBibTEX's version 1.3. TUG*boat*, 24(2):249–262, July 2003.

Jean-Michel Hufflen. MlBibTEX: a survey. In *Proc.* GUIT *Meeting*, pages 171–179, Pisa,Italy, October 2005.

Jean-Michel Hufflen. Names in BibTEX and MlBibTEX. TUG*boat*, 27(2):243–253, November 2006. TUG 2006 proceedings, Marrakesh, Morocco.

Jean-Michel Hufflen. Managing order relations in MlBibTEX. TUG*boat*, 29(1):101–108, 2007. EuroBachoTEX 2007 proceedings.

Jean-Michel Hufflen. Revisiting lexicographic order relations on person names. In *Proc. BachoTEX 2008 Conference*, pages 82–90, April 2008.

Jean-Michel Hufflen. A comparative study of methods for bibliographies. TUG*boat*, 32(3):289–301, October 2011. Proc. TUG 2011 conference.

Jean-Michel Hufflen. MlBibTEX and the biblatex package. In Tomasz Przechlewski, Karl Berry, and Jerzy B. Ludwichowski, editors, *Twenty Years After. Proc. BachoTEX 2012 Conference*, pages 91–99, Bachotek, Poland, April 2012a.

Jean-Michel Hufflen. Gestion d'ordres lexicographiques multilingues avec xindy et MlBibTEX. À paraître dans les *Cahiers GUTenberg*, 2012b.

Richard Kelsey, William D. Clinger, and Jonathan A. Rees, with 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, 11(1):7–105, August 1998.

Oleg E. Kiselyov. XML *and Scheme*. `http://okmij.org/ftp/Scheme/xml.html`, September 2005.

Philipp Lehman. *The biblatex Package. Programmable Bibliographies and Citations. Version 1.6*. `ftp://ftp.tex.ac.uk/archive/Archive%20directory/macros/latex/exptl/biblatex/doc/biblatex.pdf`, July 2011.

Mike Loukides and Andy Oram. *Programming with* GNU *Software*. O'Reilly & Associates, Inc., December 1996.

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

Lorenzo Pantieri. L'arte di gestire la bibliografia con biblatex. *ArsTEXnica*, 8:48–60, Ottobre 2009.

Oren Patashnik. *BibTEXing*. Part of the BibTEX distribution, February 1988a.

Oren Patashnik. *Designing BibTEX Styles*. Part of the BibTEX distribution, February 1988b.

Manuel Serrano. *Bigloo. A Practical Scheme Compiler. User Manual for Version 3.3b*, March 2010.

Gary V. Vaughn, Ben Ellison, Tom Tromey, and Ian Lance Taylor. GNU *Autoconf, Automake, and Libtool*. Sams, October 2000.

Herbert Voß. *Bibliografien mit LaTEX*. Lehmans Media, Berlin, 2011.

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

▷ Jean-Michel Hufflen
FEMTO-ST (UMR CNRS 6174) &
University of Franche-Comté,
16, route de Gray,
25030 BESANÇON CEDEX
FRANCE
`jmhuffle at femto-st dot fr`