MlbibTEX and Its New Extensions

Jean-Michel Hufflen

These last years, MlbibTEX's kernel functions have been reused and extended in order to put new programs about bibliographies into action. Examples are the `hal` program, allowing an open archive site to be populated, the `mlbiblatex` program, building bibliographies suitable for the biblatex package, the `mlbibcontext` program, doing the same task for ConTEXt documents. We show how all these programs are organised, and explain how some operations can be refined or extended. For a point of view related to efficiency, the programs `mlbiblatex` and `mlbibcontext` are written using Scheme only, so they are more efficient than analogous programs that would interpret a .bst bibliography style of bibTEX.

I dedicate this article to my late father (1922–2012). When I was a child, he introduced me to the joy of reading. He was himself an avid reader; I surely share this feature with him.

1. Introduction

LATEX [23] is rightly viewed as a wonderful word processor for typesetting written documents. Besides, it is assisted by other programs like bibTEX [24] as bibliography processors which generate 'References' sections (.bbl files), or other graphical tools [4]. As a proof that TEX's community of developers is very dynamic, many programs—including LATEX itself—have evolved and been improved for many years. Other formats based on TEX or engines related to it have come out: e.g., XƎTEX [19], LuaTEX [7]. We can observe analogous dynamism about graphical tools: compare the two editions of *The LATEX Graphics Companion*, [5] and [4].

As we mentioned in [16], bibTEX was unrivalled as the bibliography processor usually associated with LATEX for a long time. Besides, bibTEX is stable for many years. In fact, some slight extensions, built out of bibTEX's source files, have been designed, e.g., bibTEX8 [23, § 13.1.1] and bibTEXu [29, § 4.3] (see [16] for more details). The difficulty of writing a new bibliography processor from scratch is mainly related to bibliography database files. Many LATEX users have a *huge* number of .bib files, according to the format used by bibTEX. So a new bibliography processor designed to work in conjunction with LATEX should be able to deal with this format. At first glance, it is not very complicated, entries' metadata are 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 = ”Villa Vortex”
```

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 what braces[1]:

```
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 [10] for more details—is accurate for simple cases, but may be

[1] Personally, we always recommend users to adopt this convention, simpler, from our point of view.

```
@BOOK{holmstrom2011,
        AUTHOR = {Darwin Holmstrom},
        TITLE = {Toxic Terrain},
        SERIES = {Don Pendleton's The
                  Executioner},
        NUMBER = 390,
        PUBLISHER = {Gold Eagle},
        TOTALPAGES = 192,
        YEAR = 2011,
        MONTH = may}
```

Figure 1: Example using bibTEX's format.

surprising in such a case:

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

(if you remove the braces surrounding 'Le Clerc de la Herverie', that causes 'Herverie' to be viewed as the last name, 'Jean Le Clerc' as the first name, and 'de la' as a particle). In addition, many users get used to insert LATEX commands inside values of bibTEX fields:

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

what would be difficult to interpret for a converter into a language used to put Web pages into action. Moreover, such a declaration:

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

yields a title's specification which would be correctly interpreted by LATEX, but ConTEXt [6] 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[2]-like syntax, or '`(KEY value)`' if Lisp[3]-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 a few programs run the risk of analysing the contents of fields' values.

```
<book id="holmstrom2011" from="mb.bib">
  <author>
   <name>
    <personname>
     <first>Darwin</first>
     <last>Holmstrom</last>
    </personname>
   </name>
  </author>
  <title>Toxic Terrain</title>
  <publisher>Gold Eagle</publisher>
  <number>390</number>
  <series>
   Don Pendleton's The Executioner
  </series>
  <totalpages>192</totalpages>
  <year>2011</year>
  <month><may/></month>
</book>
```

(The `from` attribute of the book element is set to the base name of the .bib file originally containing this entry.)
Figure 2: Fig. 1's example given using XML syntax.

Let us recall that we have developed MlbibTEX[4] [9], as a 'better' bibTEX with particular focus on multilingual features. 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. Later, we were asked for a program populating an open-archive site from the entries of .bib files [14,15].Although this program needed conventions more precise than usually about .bib files, we succeeded in developing it quickly. More precisely, they have many fragments in common, and the different parts were easily assembled. We decided to do again this kind of experiment... and succeeded again. First we explain how MlbibTEX can be extended. Second we recall some advantages of using MlbibTEX's kernel. Then we sketch the variants of MlbibTEX out.

[2] e**X**tensible **M**arkup **L**angage.

[3] **LIS**t **P**rocessor.

[4] **M**ulti**L**ingual bibTEX.

2. MlbibTEX's extensibility

When MlbibTEX's parser processes a .bib file, we can consider that it builds an XML tree of this file. More precisely, this program written using Scheme [18] builds expressions according to the SXML[5] format [20]. For example, Fig. 1's entry is translated to the XML tree given in Fig. 2. We can see that the author's name has been split into these components. Likewise, LATEX commands—e.g., \em or \emph—are recognised and replaced by XML tags.

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[6]. By default, the fields subject to additional check are:

- the standard fields AUTHOR, EDITOR, MONTH, PAGES, and YEAR;
- the field DAY, used by numerous styles[7];
- the fields GENDER and TOTALPAGES, used by the bibliography styles associated with the jurabib package [23, § 12.5.1];
- two special fields used by MlbibTEX: LANGUAGE [9] and LASTSORTKEY [12].

The second extension of MlbibTEX—as abovementioned, the hal program, populating an open-archive site from the entries of .bib files [14]—needs additional check about the ADDRESS field of an entry being type @INPROCEEDINGS: we have to extract the country of the corresponding conference, and optionally the town. In addition, the name of such a country is to be checked, because we have to give its ISO[8] code. So we have decided to accept declarations like:

```
ADDRESS = {Breskens, The Netherlands}
```

or 'ADDRESS = {The Netherlands}'. If the country is not given—e.g., in 'ADDRESS = {New-York}' or:

```
ADDRESS = {Paris, Texas}
```

—an error has to be reported[9]. So we implemented a switch mechanism that allowed us to perform 'classical' check about this ADDRESS field when 'original' MlbibTEX was running, and 'complete' check when this program related to open archives was used[10]. Symmetrically, disabling some check procedures would be possible within other variants. When MlbibTEX's functions work in interpreted mode, such switch can be controlled by means of Scheme functions.

Later, we noticed the *modus operandi* of the biblatex package [21]: .bbl files only contain *structures*, and formatting 'References' sections is entirely deferred to LATEX. That is why there is no need of a \bibliographystyle command. If bibTEX is used, there is only one suitable bibliography style written using bibTEX's language.

[5] **S**cheme implementation of **XML**.

[6] This is just a warning message; the corresponding information is not lost. This *modus operandi* may be viewed as an advantage: for example, if you inadvertently type 'EDITORS = ...' instead of 'EDITOR = ...' inside an entry of type @INPROCEEDINGS, 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 does 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 sorts years as strings, so the value associated with a YEAR field must be an integer.

[7] For example, the styles 'apa...', used by the American Psychology Association.

[8] **I**nternational **S**tandardisation **O**rganisation.

[9] We also accept declarations like: ADDRESS = {Washington, District of Columbia, United States} that is, a string of three comma-separated components. The first is supposed to be the town, the last the city.

[10] Technically, it is not very difficult since we consider that Scheme—as a functional programming language—allows functions to be handled like any other value. MlbibTEX's parser uses *association lists* whose elements look like (key . f) where f is the function to be called to parse the value associated with key. To perform such a switch, just change the function associated with key.

Another bibliography processor, biber [1], has come out: it builds only .bbl files suitable for biblatex. Let us consider the example of a LaTeX document using this biblatex package given in Fig. 3. The corresponding .bbl file looks like Fig. 4, and the bibliography will be formatted w.r.t. the author-date style [23, § 12.3], because of the bibstyle option of the biblatex package.

```
\documentclass{article}

\usepackage[bibstyle=authoryear]{biblatex}
\addbibresource{mb.bib}  %  The suffix is
needed.

\begin{document}

Did you read \citetitle*{holmstrom2011}?
This is a
thriller written by \citeauthor{holmstrom2011}.

\printbibliography

\end{document}
```

Figure 3: Using the biblatex package.

```
\entry{holmstrom2011}{book}{}
  \name{author}{1}{}{%
    {{uniquename=0}{Holmstrom}{H.}{Darwin}%
     {D.}{}{}{}{}}%
  }%
  \field{title}{Toxic Terrain}%
  \list{publisher}{1}{{Gold Eagle}}%
  \field{number}{390}%
  \field{series}{Don Pendleton’s The
Executioner}%
  \field{totalpages}{192}%
  \field{year}{2011}%
  \field{month}{05}%
\endentry
```

Figure 4: Reference used by the biblatex package.

The biblatex package’s conceptor introduced new entry types a bibliography processor should be able to process. On the contrary, these new types are unknown in standard bibliography styles. Again, a switch mechanism allows us to recognise these new types only when the parser is running in a kind of ‘mlbiblatex mode’. Another point is related to *dates*: in standard bibliography styles, they are specified by a YEAR field and optionally by a MONTH field. The biblatex package allows dates to be expressed this way, or by means of a DATE field allowing the specification of a *range* of dates [21, § 2.3.8]. The extension of our parser for biblatex has been revised to include these points. Let us mention that the specification of dates are crucial within bibliographies since they are used for the sort operation in most styles. A last point: the syntax of the PAGES field has been refined.

A framework similar to biblatex had been put into action by Taco Hoekwater’s bib module of ConTeXt [8]: see Fig. 5 for a source text using a bibliographical reference. This reference, as it should be produced by a bibliography processor, is given in Fig. 6. The bib module can be used with ConTeXt MkII [2], it has been reimplemented in ConTeXt MkIV by Hans Hagen [3]. In this last case, the switch we installed considers a new @CONTEXTPREAMBLE directive when a .bib file is parsed. This directive aims to replace the ‘traditional’ @PREAMBLE directive, often used to put definitions of new LaTeX commands [23, § 13.2.4]. This @CONTEXTPREAMBLE directive can be used to program some LaTeX commands put throughout .bib files and non-existing in ConTeXt.

```
\usemodule[bib] % Needed for MkII, not for
                % MkIV

\setupbibtex[database=mb]
\setuppublications[numbering=yes]

\starttext

Did you read \cite[holmstrom2011]?

\placepublications

\stoptext
```

Figure 5: Citations and bibliographies in ConTeXt.

```
\startpublication[k=holmstrom2011,
 t=book,a={{Holmstrom}},y=2011,n=2,s=Hol11]
\author[]{Darwin}[D.]{}{Holmstrom}
\pubyear{2011}
\title{Toxic Terrain}
\series{Don Pendleton’s The Executioner}
\volume{390}
\pubname{Gold Eagle}
\month{5}
\stoppublication
```

Figure 6: Reference used by ConTeXt.

3. MlbibTeX's advantages

When the approach of biblatex and ConTeXt is used, a bibliography processor does not have to provide the text of successive references of a bibliography. Since it just produces 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—the idea is to build two accurate bibliography processors out of MlbibTeX’s kernel. These two programs —`mlbiblatex` (resp. `mlbibcontext`) for biblatex (resp. ConTeXt)—are written entirely in Scheme, in order to get more efficiency. Even if we are not interested in multilingual extensions of MlbibTeX during a first step, here are the features of interest for such bibliography processors.

3.1 Order relations

In [11], 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 simple use of the `<english?` function—for English 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*[11] 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. The sixth example shows that our `<english?` 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[12]: the idea is to sort persons regarding last names, first names, birth dates, and possibly other information. As you can see, this feature—sketched in [12, § 4]—makes easier a sort by means of several successive sort keys. More details about these order relations are given in [17].

3.2 Syntactical extensions

MlbibTeX’s syntactical extensions about multilinguism are explained in detail in [9]. Presently, they are not used by the programs `mlbiblatex` and `mlbibcontext`. On the contrary, our extensions for authors’ and editors’ names can be directly usable by these two programs. 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:

[11] A zero-argument function, w.r.t. Scheme’s terminology.

[12] The `arithmetical?` function, used within Fig. 7’s last 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 <.

```
(<english? "ConTeXt" "ConTeXt")                                  ⟹  #f
(<english? "ConTeXt" "ConTeXt" (lambda () #f) < 'uppercase-1st)  ⟹  #f  ; Default values explicited.
(<english? "ConTeXt" "ConTeXt" (lambda () 'ok))                  ⟹  ok  ; Equal strings.
(<english? "ConTeSt" "ConTeXt")                                  ⟹  #t
(<english? "ConTeSt" "ConTeXt" (lambda () 'ok) >)                ⟹  #f  ; Descending order.
(<english? "ConText" "ConTeXt" (lambda () 'ok))                  ⟹  #f
(<english? "ConText" "ConTeXt" (lambda () 'ok) < #f)             ⟹  ok  ; Case-insensitive equality.
(<english? "ConText" "ConTeXt" (lambda () 'ok) < 'lowercase-1st) ⟹  #t  ; Lowercase letters take
                                                                          ; precedence.
(<english? "ConTeXt" "ConTeSt"
           (lambda ()
              (<english? "Mk" "Mk" (lambda () (<arithmetical? 2 4 (lambda () ...))))) ⟹ #f
```

Figure 7: Order relations handled by MlbibTeX.

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

[the four keywords '`first =>`', '`von =>`', '`last =>`', '`junior =>`' are available, the order of appearance being irrelevant]. In addition, 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. 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 => Euro\TeX~2012,
sortingkey => EuroTeX 2012
```

It is well-known that co-authors are connected by means of the '`and`' keyword. MlbibTeX also allows the specification of *collaborators*, by means of the '`with`' keyword; an example is given in this article's bibliography: see the reference [23].

4. MlbibTeX's programs

MlbibTeX's distribution is located at:
http://disc.univ-fcomte.fr/home/~jmhufflen
/texts/superreport/smlbibtex-1.3.tar.gz
The easiest way to install it is to compile the source files by the bigloo [25] Scheme compiler; the installation procedure [17] uses the commands `configure` [28] and `make` [22], well-known within GNU[13] software; more details are given in [17, § 4.2]. The executable programs generated are described hereafter. The complete distribution's version number is given 'classically', that is, by means of sequence of numbers. Versions of particular variants are labelled by geographical names. Those demonstrated at the EuroTeX 2012 conference are 'Breskens versions'.

4.1 mlbibtex

This programs aims to replace bibTeX and is described in [9]; you can use it analogously to 'original' bibTeX. This `mlbibtex` is the 'historical' origin of the present toolbox.

4.2 mlbibtex2xml

This program allows .bib files to be converted into XML files, according to the format internally used by MlbibTeX. You can run it as follows:

```
mlbibtex2xml ([-screen] | [-o output])\
    f0.bib f1.bib ...
```

where f_0`.bib`, f_1`.bib`, ...—the .bib suffix can be omitted—are .bib files. If the `-screen` option is used, the result is displayed at the screen, otherwise it is written into a file. If the `-o` option is used, `output` gives the output file name, oth-

[13] Recursive acronym: **G**nu is **N**ot **U**nix.

erwise, this name defaults to f$_0$-mlbiblio.xml, even if several .bib files are processed. Obviously, results look like Fig. 2.

4.3 ar-style and hal

These two programs are the first two extensions of MlbibTEX. The ar-style program can be used for activity reports' bibliographies, when they have to be conformant to the classification of the French agency AERES[14] [13]. See Section 'MlbibTEX's extensibility' and [14,15] about the hal program.

4.4 mlbiblatex

The mlbiblatex program builds .bbl files suitable for the biblatex package. You can run it as follows:

```
mlbiblatex filename.aux
    key-expr lg-code
```

where:

filename.aux
—the .aux suffix can be omitted—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'[15], '**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[16]: if no sign is recognised, the list of bibliographical items is left unsorted[17];

lg-code
is the code for the language to be used for sorting strings—this information is relevant whenever person names and titles of works are compared—available values are DE for German, EN for English, FR for French, PO for Polish; there is no default value.

Results look like Fig. 4. More detais are given in [17].

4.5 mlbibcontext

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

```
mlbibcontext filename.aux
    key-expr lg-code
```

and filename.aux, key-expr, lg-code have the same meaning. Results look like Fig. 6.

5. Future directions

As we mention above, the interface between the functions of a word processor in charge of processing 'References' sections—the commands of the biblatex package or ConTEXt MkIV—could be improved. For example, the commands mlbiblatex and mlbibcontext only deal with ascending orders. This is just related to the rough interface we designed in order to propose first experimental versions of these programs: as shown in Section 'MlbibTEX's advantages', descending orders are provided by MlbibTEX's kernel. Concerning the biblatex package, we think that an option could be added[18]:

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

other options allowing accurate information to be passed to MlbibTEX.

[14] ***A**gence d'**É**valuation de la **R**echerche et de l'**E**nseignement **S**upérieur*, that is, 'agency evaluating research and university courses'.

[15] ... an item without month information being ranked after an item with such.

[16] 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.

[17] In this case, the bibliography is *unsorted*, that is, the order of items is the order of first citations of these items throughout the document.

[18] Presently, the possible values for the backend option of biblatex are 'bibtex', 'bibtex8', 'bibtexu', 'biber'.

Likewise, ConTEXt MkIV users should be able to choose between bibTEX—or an 'enriched' bibTEX such that bibTEX8 or bibTEXu—or MlbibTEX. In this last case, we have to study how accurate information could be passed to the `mlbibcontext` program.

Some present lack of MlbibTEX: only two encodings are available, for input .bib files as well as output .bbl ones. More precisely, .bib files are supposed to be encoded w.r.t. Latin 1. The characters that are not included in this encoding—e.g., some Polish letters, such that 'ł'—can be reached only by using TEX commands—like '`\l`'[19]. About generated .bbl files, either MlbibTEX detects that the Latin 1 encoding is used by looking into the document's preamble[20], in which case this encoding is used for the .bbl file produced; otherwise, this .bbl file is a pure ASCII[21] file, all the accented letters being specified by means of TEX commands[22]. Such behaviour is due to the Scheme programming language. MlbibTEX has been written using the fifth revision of this language [18], 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[23] and will be Unicode-compliant, so a future version of MlbibTEX should be able to deal with other encodings such that Latin 2, UTF-8, UTF-16, etc.

Last but not at least, we plan to update the programs `mlbiblatex` and `mlbibcontext`, in order for them to be able to deal with MlbibTEX's multilingual features. From our point of view, that should be quite easy for `mlbibcontext`, in the sense that all the languages are available *a priori* within ConTEXt MkIV—you do not have to put all the languages you use throughout a text as options of a module like the babel package [23, Ch. 9]—but might require more work for the texts to be processed by the commands of the biblatex package.

6. Conclusion

We are personally an adept of functional programming in general and Scheme in particular. But MlbibTEX has been able to be adapted to applications other than those initially planned, what is a good quality for a program[24]. In particular, the `mlbiblatex` program succeded in taking as much advantage as possible of biblatex's features[25] with just slight modifications of our kernel. We think that we have been able to reach such adaptability and flexibility because of the use of Scheme, even if these qualities could

[19] For example, the name of the Polish city 'Łódź' should be written down '`{\L }\'{o}d\'{z}`' or '`{\L }ód\'{z}`' within a .bib file, its internal form handled by MlbibTEX is '`{\L }ód\'{z}`', since 'ó' belongs to Latin 1, whereas 'Ł' and 'ź' do not.

[20] bibTEX just read .aux files and never reads a .tex file [23, § 12.1.3], whereas the `mlbibtex` program may look into a document's preamble.

[21] **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange.

[22] Let us recall that ConTEXt MkIV texts are supposed to be encoded w.r.t. UTF-8. Since MlbibTEX cannot deal with this encoding, the output files of the `mlbibcontext` program are presently encoded w.r.t. pure ASCII.

[23] See the Web page http://scheme-reports.org. In fact, MlbibTEX has been implemented using the conventions of R5RS, what stands for '**R**evised^5 **R**eport on the algorithmic language **S**cheme' [18]. Later, a new revision (R6RS) was designed and ratified [26][27], including functions dealing with the whole range of Unicode and different encodings [27, §§ 1 & 2.9]—but for some reasons that we do not give here, most Scheme implementors did not update their programs. So MlbibTEX is still R5RS-compliant. It seems that Scheme's next version (R7RS)—see some drafts at the Web page abovementioned—will be adopted by most Scheme implementors. So we hope that we will be able to get a Unicode-compliant version of MlbibTEX very soon.

[24] More generally, some people already announced the end of Lisp dialects, or the end of TEX & Co... and these programs are still in action.

[25] Especially the notion of field *type*: for example, `@AUTHOR` is a *list of names*, `@TITLE` is a *literal*, according to the biblatex package's terminology. Analogous notions exist within MlbibTEX.

have been reached within other programming paradigms[26]. In addition, our programs can be used with a Scheme interpreter, but better efficiency is reached if programs are compiled. Even if we think that we are not in competition with a bibliography processor like biber, it is certain that a program written using Scheme is more efficient than a program written using Perl[27]. So we have spent much time when we began MlbibTEX's development, but we do not regret anything and were happy to be able to adapt this program to new requirements.

7. References

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

[2] CONTEXTGARDEN, http://wiki.contextgarden.net/Bibliography: *Bibliographies in MkII*. April 2012.

[3] CONTEXTGARDEN, http://wiki.contextgarden.net/Bibliography_mkiv: *Bibliographies in MkIV*. July 2012.

[4] Michel Goossens, Frank Mittelbach, Sebastian Rahtz, Denis B. Roegel and Herbert Voß: *The LATEX Graphics Companion*. 2nd edition. Addison-Wesley Publishing Company, Reading, Massachusetts. January 2009.

[5] Michel Goossens, Sebastian Rahtz and Frank Mittelbach: *The LATEX Graphics Companion. Illustrating Documents with TEX and PostScript*. Addison-Wesley Publishing Company, Reading, Massachusetts. March 1997.

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

[7] Hans Hagen: 'The Luafication of TEX and ConTEXt'. In: *Proc. BachoTEX 2008 Conference*, p. 114–123. April 2008.

[8] Taco Hoekwater: 'The Bibliographic Module for ConTEXt'. In: *EuroTEX 2001*, p. 61–73. Kerkrade (the Netherlands). September 2001.

[9] Jean-Michel Hufflen: 'MlbibTEX's Version 1.3'. TUG*boat*, Vol. 24, no. 2, p. 249–262. July 2003.

[10] Jean-Michel Hufflen: 'Names in bibTEX and MlbibTEX'. TUG*boat*, Vol. 27, no. 2, p. 243–253. TUG 2006 proceedings, Marrakesh, Morocco. November 2006.

[11] Jean-Michel Hufflen: 'Managing Order Relations in MlbibTEX'. TUG*boat*, Vol. 29, no. 1, p. 101–108. EuroBachoTEX 2007 proceedings. 2007.

[12] Jean-Michel Hufflen: 'Revisiting Lexicographic Order Relations on Person Names'. In: *Proc. BachoTEX 2008 Conference*, p. 82–90. April 2008.

[13] Jean-Michel Hufflen: *Classe superreport — Manuel d'utilisation*. March 2010. http://lifc.univ-fcomte.fr/home/~jmhufflen/superreport/superreport-readme.pdf.

[14] Jean-Michel Hufflen: 'Using MlbibTEX to Populate Open Archives'. In: Tomasz Przechlewski, Karl Berry, Gaby Gic-Grusza, Ewa Kolsar and Jerzy B. Ludwichowski, eds., *Typographers and Programmers: Mutual Inspirations. Proc. BachoTEX 2010 Conference*, p. 45–48. April 2010.

[15] Jean-Michel Hufflen: 'From Bibliography Files to Open Archives: the Sequel'. In: Karl Berry, Jerzy B. Ludwichowski and Tomasz Przechlewski, eds., *Proc. EuroBachoTEX 2011 Conference*, p. 61–66. Bachotek, Poland. April 2011.

[16] Jean-Michel Hufflen: 'A Comparative Study of Methods for Bibliographies'. TUG*boat*, Vol. 32, no. 3, p. 289–301. Proc. TUG 2011 conference. October 2011.

[17] Jean-Michel Hufflen: 'MlbibTEX and the biblatex package'. In: Tomasz Przechlewski, Karl Berry and Jerzy B. Ludwichowski, eds., *Twenty Years After. Proc. BachoTEX 2012 Conference*, p. 91–99. Bachotek, Poland. April 2012.

[18] 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, Vol. 11, no. 1, p. 7–105. August 1998.

[19] Jonathan Kew: 'XƎTEX in TEX Live and beyond'. TUG*boat*, Vol. 29, no. 1, p. 146–150. EuroBachoTEX 2007 proceedings. 2007.

[20] Oleg E. Kiselyov: XML *and Scheme*. September 2005. http://okmij.org/ftp/Scheme/xml.html.

[21] Philipp Lehman: *The biblatex Package. Programmable Bibliographies and Citations. Version 1.6*. 29

[26] But we think that more effort would have been needed.

[27] **P**ractial **E**xtraction and **R**eport **L**anguage. A good introduction to this language is [30].

July 2011. ftp://ftp.tex.ac.uk/archive/Archive\%20directory/macros/latex/exptl/biblatex/doc/biblatex.pdf.

[22] Miki Loukides and Andy Oram: *Programming with* GNU *Software*. O'Reilly & Associates, Inc. December 1996.

[23] 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.

[24] Oren Patashnik: *bibTEXing*. February 1988. Part of the bibTEX distribution.

[25] Manuel Serrano: *Bigloo. A Practical Scheme Compiler. User Manual for Version 3.3b*. March 2010.

[26] Michael Sperber, R. Kent Dybvig, Matthew Flatt, and Anton van Straaten, with Richard Kelsey, William Clinger, Jonathan Rees, Robert Bruce Findler and Jacob Matthews: *Revised^6 Report on the Algorithmic Language Scheme*. September 2007. hhtp://www.r6rs.org.

[27] Michael Sperber, R. Kent Dybvig, Matthew Flatt, and Anton van Straaten, with Richard Kelsey, William Clinger and Jonathan Rees: *Revised^6 Report on the Algorithmic Language Scheme—Standard Libraries*. September 2007. hhtp://www.r6rs.org.

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

[29] Herbert Voß: *Bibliografien mit LATEX*. Lehmans Media, Berlin. 2011.

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