A Citation Style Language (CSL) workshop

Daniel Stender

Abstract

CSL is a free and open XML-based language for the programming of citation styles. With these styles, bibliographical references can be printed out in different ways from several database formats, including BibTeX. The so-far over 7000 CSL styles which are currently available can be used with several popular applications like Zotero, Mendeley, or Pandoc. This article is an introduction into the programming of citation styles with CSL, based on a few example BibTeX bibliographic database records.

1 Introduction

Bibliographical references, as used in scientific publications, are pointers to cited or regarded literature. Regularly, they consist of two standardized components: an in-line citation (the "cite") refers to an entry in the publication's bibliography. Despite the common concepts, there is no uniform outline for references; rather, each scientific discipline and every publishing house has its own traditional set of conventions, which also might change between series.

In electronic typesetting, bibliographical information is often gathered in comprehensive, reusable data files. CSL[1] is a programming language for citation styles, with which differently formatted references can be generated from the same bibliographic databases. CSL (current version: 1.0.1) is XML-based, open and free, and was substantially developed for the all-around reference manager Zotero.[2]

This article demonstrates how a rudimentary example citation style could be implemented with CSL, with reference to the BibTeX data format. The usage of several programs refers to a Debian GNU/Linux based system (like Ubuntu and Linux Mint), but CSL styles could also be easily developed on other operating systems. Some basic knowledge of XML and the BibTeX data format is definitely needed to follow every detail.

2 CSL styles

The citation styles which have already been implemented in CSL (file extension: `.csl`) are collected by the CSL developers in the official style repository,[3] and in the Zotero style repository.[4] The so-far more than 7000 styles are distributed under the *Attribution-ShareAlike 3.0 Unported* license by Creative Commons.[5] For the search of specific citation styles the online CSL style editor[6] is quite useful because it provides, in addition to other features, a search by example.

3 Pandoc-citeproc

Among the several current applications which already know how to use CSL styles for the automatic generation of references, there is the popular universal markup converter Pandoc [1].[7] With short citation keys like `@doe2014 [p. 40-42]` for its extended Markdown lightweight markup, Pandoc can query bibliographical data files and recognize CSL styles to put out variously formatted references for documents either in HTML, LaTeX or ConTeXt markup, along with several others [3]. Pandoc is a command line interface application; thus, the CSL style which is going to be processed and the bibliographical database(s) are given as arguments in the program call. Here's an example (some line breaks are editorial) for Pandoc's LaTeX output of a random BibTeX database record (see below for details), formatted using `chicago-author-date.csl`:[8]

```
$ echo "On this, see @reference2 [p. 127]." \
  | pandoc --to=latex \
      --csl=chicago-author-date.csl \
      --bibliography=references.bib

On this, see Flom (2007, 127).

Flom, Peter. 2007. ‘‘LaTeX for Academics and
Researchers Who (Think They) Don’t Need It.’’
\emph{TUGboat} 28 (1): 126--128.
```

As shown in this example, the bibliography is printed at the end of a document. Incidentally, in the input (shown later), the title is given in lowercase; the titlecasing done here is automatic, a feature of this style [8, chp. 14].

Processors which produce formatted citations out of bibliographic databases according to CSL styles are called CiteProcs.[9] CiteProcs are being developed in several programming languages. The one which is used normally by Pandoc is *pandoc-citeproc*,[10] written in the same functional programming language Haskell as Pandoc itself, and developed closely together with it. This CiteProc (currently: 0.3.0.1) already deals with a number of different database

[1] http://citationstyles.org/
[2] http://www.zotero.org/
[3] http://github.com/citation-style-language/styles/
[4] http://zotero.org/styles/
[5] http://creativecommons.org/licenses/by-sa/3.0/
[6] http://editor.citationstyles.org/about/
[7] http://johnmacfarlane.net/pandoc/
[8] Although given explicitly here, this CSL style is the default in Pandoc.
[9] http://en.wikipedia.org/wiki/CiteProc/
[10] http://github.com/jgm/pandoc-citeproc/

formats, but it's said that it works best with BibTeX resp. BibLaTeX [4] databases so far.[11]

4 Developing CSL styles

CSL styles are XML, and therefore the whole related XML tool chain can also be used with them. For somebody who deals with XML regularly, a specialized editor is useful, but fundamentally CSL styles can be created and modified with any text editor. CSL is described in detail in the specification [10], and the primer which has been written by the developers [9] is a good starting point for beginners.

CSL is standardized as an XML grammar in the schema language RELAX NG,[12] and in principle any CSL style file can checked with an XML validator against the CSL schema to determine if it is correct (valid).[13] Unfortunately, some validators, for example *xmllint* (Debian package: *libxml2-utils*), cannot cope with XML schemes in RELAX NG compact syntax like the one shipped by the CSL developers (file extension: `.rnc`); the scheme has to be converted (e.g. with Trang) into the regular RELAX NG syntax (file extension: `.rng`) before a CSL style can be validated against it:

```
$ git clone https://github.com/\
citation-style-language/schema.git
Cloning into 'schema' [...]
$ trang schema/csl.rnc schema/csl.rng
$ xmllint --noout -relaxng schema/csl.rng \
    chicago-author-date.csl
chicago-author-date.csl validates
```

5 XML declaration and CSL header

The standard XML declaration commonly starts a CSL style file:

```
<?xml version="1.0" encoding="UTF-8"?>
```

Incidentally, although it is often suggested to be included, specifying the UTF-8 encoding like this could be omitted because UTF-8 is the XML default [2, p. 28].

The next thing which is needed is a well-formed CSL header to specify that the XML file is a CSL style. A standard CSL header goes like this:

```
<style xmlns="http://purl.org/net/xbiblio/csl"
       version="1.0" class="in-text">
```

This defines ⟨*style*⟩ to be the root element, and the CSL namespace is set as the default with *xmlns*.[14] That the style is compatible with CSL 1.0 is stated by *version*, while the *class* attribute determines that this style provides cites in the running text by default, rather than as footnotes or end notes (which would be "note").

6 Info block

The next mandatory unit of a CSL style is an ⟨*info*⟩ block, which provides metadata for labeling and identification. A typical info block looks like this:

```
<info>
 <title>An example CSL style</title>
 <id>http://www.danielstender.com/csldemo</id>
 <updated>2014-09-18T23:53:00+02:00</updated>
</info>
```

Even if it is not intended to publish the style, there are at least three mandatory child elements that are meant for this purpose:

⟨*title*⟩ is the title of the CSL style as it is going to be displayed to users,

⟨*id*⟩ contains, like *xmlns* in the CSL header (see above), a random URI, which may be real or fictitious, and which is solely for identification purposes,

⟨*updated*⟩ carries a *xsd:dateTime* compliant time stamp[15] of the last modification.

7 Example BibTeX records

Before getting any deeper into CSL style programming, here are a few sample BibTeX records to be referred to hereafter to demonstrate how CSL works. A typical `@Book` entry type [5, chp. 13.2] goes like this

```
@Book{reference1,
author    = {Kopka, Helmut and Daly, Patrick W.},
title     = {A Guide to LaTeX and Electronic
             Publishing},
publisher = {Addison-Wesley},
year      = 2004,
address   = {Boston},
edition   = {Fourth}}
```

The next one is an `@Article` data set of a (well-known) journal whose issues are counted as volume numbers:

```
@Article{reference2,
author  = {Flom, Peter},
title   = {LaTeX for academics and researchers
           who (think they) don't need it},
journal = {TUGboat},
year    = 2007,
volume  = 28,
number  = 1,
pages   = {126-128}}
```

[11] cf. `Readme.md`.

[12] `http://relaxng.org/`

[13] `http://github.com/citation-style-language/schema/tree/v1.0.1/`

[14] At no time will a connection to this url be established; it's only for the purpose of identification.

[15] `http://books.xmlschemata.org/relaxng/ch19-77049.html`

And another `@Article` data set of a journal which appears monthly and doesn't have any issue numbers:

```
@Article{reference3,
author  = {Sharma, Tushar},
title   = {Why I never close Emacs},
journal = {Open Source For You},
year    = 2014,
pages   = {53-55},
month   = {jan}}
```

The main body of a CSL style consists of several instructions for how exactly the CiteProc is going to typeset citations, thus the cites and the corresponding references in the bibliography of a document, from standardized data records like these.

8 The cite

Another child element of ⟨*style*⟩ is ⟨*macro*⟩. Multiple macros are permitted, and this element serves to deliver data and reusable formatting sets. The following macro provides the author surnames out of the BibTeX data field *author*:

```
<macro name="surname">
  <names variable="author">
    <name form="short" and="symbol"/>
  </names>
</macro>
```

From the CSL variable *author*, which has the same name as the BibTeX data field and initially also contains the full author names, `<name form="short"/>` extracts the surnames, while *symbol* for the attribute *and* specifies the ampersand as delimiter between multiple authors. Macro names (like "surname") are user-defined. Their result could be typeset with the rendering element ⟨*text*⟩ (see below), but macros have necessarily to be written *before* they are referenced (the XML parser needs it this way).

Another macro fetches the year of the publication out of the date variable[16] *issued*:

```
<macro name="year">
  <date variable="issued">
    <date-part name="year"/>
  </date>
</macro>
```

In CSL, the year of the publication from the BibTeX data field *year* isn't available independently; for example, *issued* also contains the BibTeX field *month*, if the data item holds that. Thus, the year of the publication has to be extracted from *issued* like this, before it is available.

With these two macros it's already possible to set up an author–year cite. Also needed is the standard variable *locator*, which provides the page reference out of the citation key in the document source. The cite is described within the element ⟨*citation*⟩, looking, for example, like this:

```
<citation>
  <layout>
    <text macro="surname" suffix=" "/>
    <group prefix="(" suffix=")">
      <text macro="year"/>
      <text variable="locator" prefix=":\,"/>
    </group>
  </layout>
</citation>
```

Along with the ⟨*layout*⟩ element, ⟨*citation*⟩ has another child for sorting, ⟨*sort*⟩, which defines the order of multiple references within one and the same cite. When ⟨*sort*⟩ isn't used, the order of appearance is kept as is the case here.

With a ⟨*citation*⟩ block like this one the bibliographic records above could be cited easily, for example again with Pandoc:

```
$ echo "@reference1 [p. 100], @reference2
[p. 127], @reference3 [54-55]." \
  | pandoc --to=latex --csl=example.csl \
           --bibliography=references.bib

Kopka \& Daly (2004:\,100), Flom (2007:\,127),
Sharma (2014:\,54--55).
```

The parentheses and colon punctuation were set up using the *prefix* and *suffix* attributes of the ⟨*group*⟩ element, above.

9 The bibliography

In CSL, how the full records appear is defined within the ⟨*bibliography*⟩ block. Using the bibliographical records above as an example, we will set up a citation style resulting in the following (with `\frenchspacing` in effect):

[Kopka & Daly 2004] Helmut Kopka, Patrick W. Daly: "A Guide to LaTeX and Electronic Publishing". Fourth edition. Boston: Addison-Wesley 2004.

[Flom 2007] Peter Flom: "LaTeX for academics and researchers who (think they) don't need it". In: *TUGboat* 28,1 (2007), p. 126–128.

[Sharma 2014] Tushar Sharma: "Why I never close Emacs". In: *Open Source For You* 1/2014, p. 53–55.

The individual elements of this style can be implemented in CSL as follows. As was the case with ⟨*citation*⟩, everything which is going to be printed has to be put within the ⟨*layout*⟩ child element of ⟨*bibliography*⟩ (see p. 313 for the complete code).

[16] The CSL variables are divided into four different classes: standard, number, date, and name.

9.1 Label

The first element that is going to be set up is a label which matches the cite. That's for the purpose of easily locating the corresponding record in the bibliography. As we did with ⟨*citation*⟩ above, a label like this can be constructed using the macros *surname* and *year*:

```
<group delimiter=" " prefix="[" suffix="] ">
  <text macro="surname"/>
  <text macro="year"/>
</group>
```

9.2 Authors

As explained above, the CSL variable *author* contains the full author names. This information can easily be output directly with the rendering element ⟨*names*⟩, when it is needed:

```
<names variable="author" delimiter=", "
       suffix=":"/>
```

9.3 Title

The variable *title* is meant to be used with the rendering element ⟨*text*⟩:

```
<text variable="title" prefix="‘‘" suffix="’’."/>
```

Features like the wanted quotation marks are implemented in Unicode.[17] The CiteProc converts them into LaTeX standard quotes, if this is chosen as the output format (see below for the result).

9.4 *container-title*

The variable *container-title* represents the BibTeX data field *journal*:

```
<text variable="container-title" prefix="In: "
      font-style="italic"/>
```

This variable remains empty when entry types like `@Book` are processed, and therefore nothing would be printed out in this case.

9.5 Journal issue

The following macro extracts the month of publication, which comes from the BibTeX field *month*, from the date variable *issued*:

```
<macro name="month">
  <date variable="issued">
    <date-part name="month" form="numeric"/>
  </date>
</macro>
```

In the same process, the month of appearance, which is commonly recorded in BibTeX in the form of "jan", "feb", etc.,[18] gets converted into the corresponding number by setting the attribute *form* to *numeric*.

Using this macro, we can define a second macro to render the desired output of the journal issue either as "*volume*,*number* (*year*)", or as "*month*/*year*" for journals appearing monthly:

```
<macro name="issue">
 <choose>
  <if type="article-journal" variable="volume">
   <text variable="volume"/>
   <text variable="issue" prefix=","/>
   <text macro="year" prefix=" (" suffix=")"/>
  </if>
  <else-if type="article-journal">
   <text macro="month"/>
   <text macro="year" prefix="/"/>
  </else-if>
 </choose>
</macro>
```

The CSL variable *issue* represents the BibTeX field *number*. This macro checks whether the publication type is *article-journal* (a condition which is satisfied by `@Article` entry types of BibTeX), and, depending on whether *volume* data for the record is available, prints the element of the reference in the desired way.

Generally, it's best to keep formatting switches dependent on publication type out of ⟨*bibliography*⟩,[19] but it's fine to have a macro like this:

```
<text macro="issue" suffix=","/>
```

9.6 Page numbers

The page numbers in a BibTeX data set are directly available through the rendering variable *page*. If a record carries any, they can be output with, for example:

```
<text variable="page" prefix="p. "/>
```

9.7 Edition

The same is true for *edition*, which also could be rendered directly, like this:

```
<text variable="edition" suffix=" edition."/>
```

However, although in theory it ought to be homogeneous,[20] when dealing with data files from different origins the literal BibTeX field *edition* often carries

[17] These are the Unicode entities U+201C LEFT DOUBLE QUOTATION MARK and U+201D RIGHT DOUBLE QUOTATION MARK.

[18] "For reasons of consistency the standard three-letter abbreviations (`jan`, `feb`, `mar`, etc.) should be used" [5, p. 765].

[19] "The use of macros can improve style readability, compactness and maintainability. It is recommended to keep the contents of `cs:citation` and `cs:bibliography` compact and agnostic of item types (e.g., books, journal articles, etc.) by depending on macro calls" [10].

[20] "However, the edition field poses a bit of a challenge. The BibTeX standard way of specifying edition numbers is to use ordinal words with capital first letters such as "First", "Second", "Third" and so forth" [7, Q13].

Figure 1: The full code of ⟨*bibliography*⟩

```
  <bibliography>
    <sort>
      <key macro="surname"/>
    </sort>
    <layout suffix=".">
      <group delimiter=" " prefix="[" suffix="] ">
        <text macro="surname"/>
        <text macro="year"/>
      </group>
      <group delimiter=" ">
        <names variable="author" delimiter=", " suffix=":"/>
        <text variable="title" prefix="‘‘" suffix="’’."/>
        <text variable="container-title" prefix="In: "font-style="italic"/>
        <text macro="issue" suffix=","/>
        <text variable="page" prefix="p. "/>
        <text variable="edition" suffix=" edition."/>
        <text macro="published"/>
      </group>
    </layout>
  </bibliography>
</style>
```

record types of different natures, such as “Second”, “second”, “2nd”, “2”, etc. Therefore, if a CSL style needs to be robust, and requires an exact format for edition information, type queries and conversions routines may be needed especially for this field. For this purpose, CSL provides a number of different tests for complex, conditional processing of data fields; for example, *is-numeric* returns a (Boolean) “false” if a variable has the form “Second”, “second”, etc.

9.8 Publication details

The rendering of the publication details of books can be implemented like this:

```
<macro name="published">
<choose>
 <if variable="publisher">
  <group delimiter=" ">
  <text variable="publisher-place" suffix=":"/>
  <text variable="publisher"/>
  <text macro="year"/>
  </group>
 </if>
</choose>
</macro>
```

This macro first checks whether the variable *publisher* is defined (which is not the case with `@Article`), and, if this is true, renders it together with *publisher-place* (which adopts the BibTeX field *address*) and again the macro *year* in the desired way for this citation style. This macro could be employed similarly to the others, with:

```
<text macro="published"/>
```

9.9 Sorting key

With an author–date citation style like this it's useful to install a sort order, or the records are going to appear in the order of occurrence of the corresponding cites, which is typically not wanted. The following sort key puts the entries of the bibliography into the alphabetical order of the author's surnames:

```
<sort>
 <key macro="surname"/>
</sort>
```

10 Result

With these features and the closing ⟨*/style*⟩ root element, the very basic citation style which we intended to implement is completed. Like the others, this CSL style could be used to produce complete formatted citations out of the example BibTeX data.

The LaTeX formatted Pandoc output of the example references looks like this (with some editorial line breaks):

```
{[}Flom 2007{]} Peter Flom: ‘‘LaTeX for
academics and researchers who (think they)
don’t need it’’. In: \emph{TUGboat} 28,1
(2007), p. 126--128.

{[}Kopka \& Daly 2004{]} Helmut Kopka,
```

```
Patrick W. Daly: ‘‘A Guide to LaTeX and
Electronic Publishing’’. Fourth edition.
Boston: Addison-Wesley 2004.

{[}Sharma 2014{]} Tushar Sharma: ‘‘Why I never
close Emacs’’. In: \emph{Open Source For You}
1/2014, p. 53--55.
```

To be sure, what has been set up here is far from robust and is just for demonstration purposes. The experienced bibliography writer knows that even with only the basic publication types which have been discussed, plenty of open questions remain which would go beyond our scope here. A more refined style would need additional features such as book titles set in italics, using the prefix "pp." for page ranges, using "et al." for multiple authors if required by the style, etc. These topics and several others are planned to be the subject of a follow-up article.

In general, CSL offers features for every last detail of bibliographical typesetting; the styles which are actually used in production are much more complex than what has been demonstrated here.

11 Conclusion

CSL provides a sophisticated and versatile tool (e.g. it also supports localization) for the programming of citation styles. It has already become widespread, for good reason.

In my opinion, CSL responds to the natural complexity of the subject "citation" with a very elegant, intuitive and simple XML-based user interface. This distinguishes CSL from the, for example, difficult-to-penetrate stack-based BibTeX language for `.bst` styles [6].

Although a CSL preprocessor for LaTeX, to the best of my knowledge, still remains a desideratum, it is still highly recommended to become familiar with CSL when dealing with bibliographical typesetting. Finally, until a CSL capable replacement for the BibTeX preprocessor becomes available, Pandoc's LaTeX output is useful.

References

[1] Massimiliano Dominici. An overview of pandoc. *TUGboat*, 35(1):44–50, 2014. URL: http://tug.org/TUGboat/tb35-1/tb109dominici.pdf.

[2] Joe Fawcett, Liam R.E. Quin, and Danny Ayers. *Beginning XML*. John Wiley & Sons Inc., Indianapolis, fifth edition, 2012.

[3] Axel Kielhorn. Multi-target publishing. *TUGboat*, 32(3):272–277, 2011. URL: http://tug.org/TUGboat/tb32-3/tb102kielhorn.pdf.

[4] Philipp Lehman, Philip Kime, Audrey Boruvka, and Joseph Wright. The BibLaTeX package: Programmable bibliographies and citations. version 2.9a. 24/06/2014, 2014. URL: http://ctan.org/pkg/biblatex.

[5] Frank Mittelbach, Michel Goossens, et al. *The LaTeX Companion*. Addison-Wesley Series on Tools and Techniques for Computer Typesetting. Addison-Wesley, Boston, second edition, 2004.

[6] Oren Patashnik. Designing BibTeX styles, 1988. URL: http://mirror.ctan.org/biblio/bibtex/base/btxhak.pdf.

[7] Michael Shell and David Hoadley. BibTeX tips and FAQ. version 1.1, 2007. URL: http://mirror.ctan.org/biblio/bibtex/contrib/doc/btxFAQ.pdf.

[8] University of Chicago Press staff, editor. *The Chicago Manual of Style*. University of Chicago Press, Chicago, Ill., sixteenth edition, 2010.

[9] Rintze M. Zelle. Citation style language 1.0: Primer, 2011. URL: http://citationstyles.org/downloads/primer.html.

[10] Rintze M. Zelle, Frank G. Bennet, Jr., and Bruce D'Arcus. Citation style language 1.0.1: Language specification, 2012. URL: http://citationstyles.org/downloads/specification.html.

◇ Daniel Stender
Hamburg, Germany
daniel (at) danielstender.com
http://www.danielstender.com/