GELLMU: A Bridge for Authors from LaTeX to XML

William F. Hammond
Department of Mathematics & Statistics, University at Albany
hammond@math.albany.edu
http://www.albany.edu/~hammond/

Abstract

GELLMU, which stands for "Generalized Extensible LaTeX-Like Markup", is a system for using LaTeX-like markup, though not LaTeX itself, to write consciously for a markup language in the SGML category or in its popular XML subcategory. The *basic* level of GELLMU offers a way to use LaTeX-Like notation together with a LaTeX-Like *newcommand* (with arguments) macro facility to write web pages. The *advanced* level of GELLMU enables one additionally to incorporate certain LaTeX-Like features, such as the use of a blank line for a new paragraph, in writing for an SGML language. The didactic GELLMU production system provides an "article" XML language, with some resemblance to LaTeX itself, that is a rigorous domain for translation to other formats.

Author Level Markup

Inasmuch as the World Wide Web is becoming an important library resource, one wants one's publications to be accessible online, and one wants web-crawling robots to be able to catalogue them properly.

Despite the popularity of Adobe's Portable Document Format (PDF) the distribution of PDF reading software is not as widespread as the distribution of web browsing software, and web-crawling robots often do not scan the contents of PDF documents.

What is available for the LaTeX author toward this end? More specifically, consider the following situations:

Online publication archives Specifically, I would like to cite the TeX/LaTeX-based archive begun at Los Alamos in the early 1990's by Paul Ginsparg, now known as "arXiv", which is now a participant in the Open Archives Initiative. While in its early time the term *e-print* was understood to mean "electronic pre-print", it has more recently become a repository for established journals including now, for example, the highly regarded *Annals of Mathematics*, which was founded in 1884 by Ormond Stone of the University of Virginia, and Ginsparg now tells us that the term *e-print* denotes "self-archiving by the author" under a new overall academic publication[1] design.

Course handouts How can a college teacher prepare course handouts for both paper and online distribution? If the teacher writes LaTeX, some manual intervention is likely to be needed in order to obtain correct HTML. If the teacher writes HTML, then the paper distribution[2] will be limited by what can be expressed in HTML, which is not as rich a markup as LaTeX.

TUG articles Before preparing a TUG 2001 article an author is asked to read *Preparation of documents for multiple modes of delivery* by Ross Moore, which is available on the web only as a two-column printed page image in Adobe's Portable Document Format. From this article one might conclude that carefully prepared LaTeX may be suitable for translation to HTML although no HTML version of the article seems to be available.

GNU documentation While working for TUG on the TeX Directory System (TDS) guidelines in January 1998, Ulrik Vieth produced a particular LaTeX document (CTAN:tex-archive/tds/draft-standard/tds-0.9995) and a tailored program for translating that document into *Texinfo*, The Language of the GNU Documentation System.

[1] Paul Ginsparg, "Electronic Clones vs. the Global Research Archive", http://arXiv.org/blurb/pg00bmc.html.
[2] If the HTML is correctly written, then robust translation to LaTeX is possible.

Texinfo is a TEX-based system that pre-dates HTML. Its original purpose was to provide both print and (early online hypertext) *Info* versions of GNU software project documentation. When HTML came along, it was possible to provide fairly reliable transation from *Texinfo* to HTML because *Texinfo* is a well-structured markup. In fact, *Texinfo* is very nearly equivalent to an SGML language, and, Daniele Giacomini in August 2000 came up with an effort in that direction: *Sgmltexi*.

Although programs are available for translating carefully structured LATEX into HTML and sometimes for XML extensions of HTML, this method of generating online content for the basic level of the web sometimes requires manual intervention. A more direct approach to the world of SGML offers better prospects for long-term access to new web formats without sacrificing access to the quality of print typesetting that is available through LATEX.

The basics of *basic* GELLMU

In looking over Vieth's set-up for the TDS document in the late spring of 1998, I arrived at the idea of using LATEX-like notation for conscious writing in document languages under SGML and I have written a program in the GNU Emacs Lisp language, the GELLMU syntactic translator, for converting this LATEX-like markup to SGML markup.

The advantage of SGML markup is that each markup language (formally document type) under the SGML umbrella constitutes a structured domain for the application of automatic processors that are easy to create under any of a number structured processing frameworks. There are frameworks accessible in standard computing languages, and there is also a recent framework *xmltex* by David Carlisle for writing TEX typesetting routines for XML document types.

The root idea in using LATEX-like markup for the conscious writing of markup under SGML is the simple syntactic correspondence between markup such as

```
some \em{emphasized} text
```

on the one hand, and the markup

```
some <em>emphasized</em> text
```

on the other.

Most LATEX commands are analogous to SGML elements. Moreover, the attribute list associated with an SGML element can be made to correspond with a LATEX command option. For example,

```
\a[href="http://foo.dom/"
]{The Foo Domain}
```

matches

```
<a href="http://foo.dom/"
>The Foo Domain</a> .
```

Enhancement with *newcommand*

The idea of using LATEX-like syntax for conscious writing under an SGML document type gains power when one realizes that although the notion of SGML entity provides, among other things, simple macro expansions, there is no provision under SGML for macros that take arguments. Moreover, there is no obvious method of extending SGML systems to accommodate macros with arguments apart from the idea of extending a document type[3].

GELLMU provides a LATEX-like meta-command[4] called *newcommand* that may be invoked with arguments. For example, if one writes

```
\newcommand{\afoo}[2]{%
\a[href="http://www.foo.org/#1 "]{#2}}
```

then a subsequent invocation

```
\afoo{tex-archive/tds/}{TDS at Foo}
```

[3] While document type extensions require enough work that they cannot be spontaneous, they provide a sound way to avoid the tangles that can arise working with TEX or LATEX when attempting the simulataneous use of conflicting macro packages.

[4] In GELLMU while a *command* corresponds to an SGML element, a *meta-command* is something having the same syntax as a command that does not correspond to an SGML element and instead receives resolution into other SGML markup under the syntactic translator.

will yield (without line breaks):[5]

```
<a href=
"http://www.foo.org/tex-archive/tds/"
>TDS at Foo</a>
```

This *newcommand* markup differs from that of LaTeX in that it is classical macro substitution rather than vocabulary expansion. Since the syntax of a *newcommand* invocation is very similar to that of an SGML element, the use of *newcommand* can, apart from its on-the-fly convenience, be a help in the development of SGML document type extensions. A new name in a test document can be moved from being that of a macro to being that of an element simply with the removal of a *newcommand* definition.

SGML vs. XML

Basic GELLMU as enhanced by its macro facility is as far as one can sensibly go toward conscious writing under a language in the restricted subfamily of SGML document types known as XML.

From one viewpoint the differences between SGML and XML are not very important since most correct documents under the larger category can, if correct, be automatically translated into equivalent documents under XML. For example, classical HTML that passes validation can be translated into the newer XML form of HTML using either James Clark's classical SGML library *SP* or Dave Raggett's program *tidy*.

However, the rules for XML were designed to make things easy for processors rather than for humans, and for that reason an author writing toward an ultimate XML document type usually is well-advised to write for a version of the document type under more author-friendly SGML rules.

For example, if in an SGML language a forced linebreak is represented by the defined-empty element *brk*, then the markup `<brk>` is sufficient, whereas under the more restrictive XML version of the same language, either the markup `<brk></brk>` or its abbreviated form `<brk/>`[6] must be used. For GELLMU this means that the markup `\brk` and the markup `\brk{}` are interchangeable under SGML except for the case of `\brk` abutting a following character without intervening whitespace but not equivalent under XML. GELLMU provides the form `\brk;` to represent the abbreviated form `<brk/>` of an element that is defined as empty.

Advanced GELLMU

Basic GELLMU deals with markup languages more or less at the level of syntax without getting to the level of grammar.

Advanced GELLMU may be used to roll language-independent grammatical concepts into the picture.

The first of these is LaTeX-like multiple argment/option syntax. For example under advanced GELLMU the markup

```
\frac{a x + b}{c x + d}
```

is converted in syntactic translation to

```
<frac><ag0>a x + b</ag0><ag0>c x + d</ag0>
```

That is, a chain of '{', '}' pairs and '[', ']' pairs following a command without intervening white space between the command name and the first delimiter nor between a close delimiter and the next open delimiter in the chain, constitutes an SGML element whose content begins with a sequence of generic positional arguments (tag name *ag0*) and options (tag name *op0*). Without knowledge of the document type it cannot be determined if a name used with multiple argument/option syntax has only *ag0*, *op0* content. The syntactic translator provides a list variable consisting of names that have only this type of content and that, therefore should be given close tags after the sequence of arguments and options. Absent that, the author must provide a close tag unless an SGML parser can infer it, and even in that case, if the element can appear in the mixed content model of another element such as, for example, a paragraph, then the parser's automatic placement of a close tag could lead to the unwanted collapse of a word boundary similar to that which occurs in LaTeX when an author's careless markup

```
\TeX benchmark
```

gets typeset as "TeXbenchmark" instead of as "TeX benchmark".

[5] The syntactic translator maintains line number alignment between its input and its SGML output so that line numbers used by SGML parsers in flagging errors match those in source markup.

[6] Moreover, some confusion may arise from the fact that under the SGML syntax (formally SGML declaration) specified for HTML neither of these XML forms of markup would not be permitted.

If multiple argument/option syntax is used, then there is ambiguity on the nature of the first pair of chained delimiters if it is the pair ‘[’, ‘]’ — whether it represents an *op0* or an attribute list. Therefore, in this case it is required that it is an attribute list if the first character after its ‘[’ opening delimiter is a colon (‘:’).

In basic GELLMU the following four of the ten LaTeX-special characters are special:

\ { } % .

Additionally, the character ‘#’ is special when used in the definition of a *newcommand*, the characters ‘[’ and ‘]’ are special when used for LaTeX-like option syntax, and the character ‘&’ is special when followed immediately by a letter since then it is the introducer for SGML entity invocations.

Advanced GELLMU provides for the possibility giving traditional LaTeX meaning to ‘&’ when not followed immediately by a letter and also to the other four LaTeX-special characters, which are

_ ^ $ ~ .

Additionally, it provides for the possibility of giving traditional LaTeX-like meaning to other short forms of markup such as "\(. . . \)" for inline math, "\[. . . \]" for displayed math, "--" for a range-dash, "---" for a punctuation-dash, "\ " for an inter-word space, "\," for a short space, and others including also, if desired, the use of blank lines, as appropriate, for paragraph boundaries.

The didactic production system

The conversion of both *basic* and *advanced* GELLMU source markup to SGML is performed by my program called the *syntactic translator*.

If one wishes to write consciously for a public document type such as HTML or the Text Encoding Initiative's TEI using GELLMU's LaTeX-like syntax, the syntactic translator is the only part of GELLMU that will be of interest.

The optional features of advanced GELLMU described above can only be used when one is writing for a document type that provides markup in which the corresponding concepts have representation. For example, LaTeX-like use of the character ‘~’ for non-breaking space requires a markup that provides non-breaking space.

Moreover, if blank lines are going to be paragraph boundaries, then the syntactic translator will need a list of element names before which a new paragraph does not make sense, and, since there is no separate provision for a list of names after which a paragraph must end, the document type cannot be XML.

The GELLMU didactic production system provides such a document type and also provides tools, which can be used as inter-changeable components, for working with that document type.

The didactic production system consists of the syntactic translator and the following additional components:

1. An SGML document type called “article”.
2. Its XML cousin, also called “article”.
3. A program for translating SGML article to XML article.
4. A program for translating XML article to HTML.
5. A program for translating XML article to LaTeX.

The document type is intended to be comfortable for authors with past experience in LaTeX. The document type and the components are didactic. They are intended to illustrate how such a system can be assembled from inter-changeable components. They are not finished in any sense, and each has shortcomings.

They do serve, I hope, to demonstrate to the community of LaTeX authors that it will find no limitations in this approach to document production.

At the same time it is intended to provide a whole new way of thinking about the subjects of package design and class design.

Its unfinished nature is intended to make it relatively easy for those who are so inclined to move in various ways to finish such a system that fits their needs.

Production of this Document

This document was prepared using the GELLMU didactic production system. Parallel to the online version in my home web one may find:

- GELLMU source
- syntactic translation
- centrally styled XML version
- LaTeX formatting for print
- DVI for print from *latex*
- LaTeX formatting for screen
- PDF for screen from *pdflatex*

The HTML and LaTeX print formattings were made using the formatters in the didactic production system. The screen-optimized LaTeX formatting was made with a small variation of the didactic LaTeX formatting that changes page size, invokes Sebastian Rahtz's well known *hyperref* package, and does a very small dance around some current font encoding issues.

Subsequent to the GELLMU run a copy of its LaTeX output was manually modified for conformance with TUG guidelines. If I were going to submit a number of such TUG articles, it would be worthwhile to make another variant of the LaTeX formatter for TUG.