# OPEN-SOURCE SOFTWARE SYSTEM FOR MULTI-AUTHOR DOCUMENTS*

L. Emery†, ANL, Argonne, IL 60439, USA

*Abstract*

An efficient means was developed to manage multi-author documents using software components not usually run together that are both freely available and free of cost: Concurrent Version Software (CVS), LaTeX typesetting software, and the Unix "make" utility. Together they solve the main problem with multi-author documents: losing track of the "latest" version, tracking author contributions, and a strict enforcement of document format. APS has used this system for two large documents with about a dozen authors each: a 2007 white paper (150 pages) on a ERL proposal and a chapter (230 pages) of the APS Upgrade CDR. We stress the use of LaTeX because the plain-text format is amenable to version comparisons, the macro-based system allows last-minute global format changes, and finally LaTeX just looks better than Microsoft Word. Several contributions from APS to this conference actually use this system.

## INTRODUCTION

Most of the accelerator literature consists of multi-author documents. They could be large texts for facility construction proposals, or simply conference proceedings with two or more authors that (actually) share the writing work.

Many editing and content changes are made in the process of writing, thus producing many past versions of the document, though not all worth retaining. Popular What-You-See-Is-What-You-Get (WYSIWYG) software such as Microsoft Word and OpenOffice have ways of tracking changes by individual authors. The changes from the original can be viewed in color-coded fragments, though this may be confusing as well. When an author sends a new version to the rest of the authors some of them may already have made changes to the previous version, and thus need to reconstruct a new version themselves before continuing the editing. Finally piecing parts together after all authors have contributed may be a confusing task. The authors may resort to renaming of files in order to understand the order in which the changes have been made.

In large multi-chapter documents, sections or chapters may be the responsibility of individual authors, which is less prone to confusion. The remaining risk is that in writing a small part of the document, a particular author may not respect the agreed-upon format, which comprises an extensive set of parameters: page layout, paragraph styles, fonts, font size, section header formats, table and figure styles and captions, equations styles, citation style, bibliography style, etc. In addition typing a backspace while editing a WYSIWYG document could change the document in an unseen way, which has ramifications later.

Large scientific documents also have many figures, equations, tables, and references. When a new one of these are inserted in the document, then all the following instances must be renumbered manually, which is painstaking work and prone to error. Twenty years of WYSIWYG word processing development has not addressed this.

In addition, the paragraphs formed by WYSIWYG software line-breaking algorithms are not as pleasing to the eye of cultured readers compared to those that are professionally typeset.

Thus we propose using a versioning software to resolve the editing synchronization between authors and organize the versions, using LaTeX for the aesthetics and the automatic reference numbering, and the `make` Unix utility to compile the LaTeX files, which is sometimes onerous, but necessarily so. Programs LaTeX, `CVS` and `make` require the use of a command line, which should not be difficult for scientists and engineers, who are paid to learn things.

With this system we have written two large documents with about a dozen authors, one a 2007 white paper (150 pages) on a ERL proposal, and the other a 230-page chapter of the APS Upgrade Conceptual Design Report (CDR).

In addition these programs are open source, meaning that source code is available to examine freely. These program are also free of charge. Beginners have at their disposal a vast amount of help through the bulletin boards that many expert users frequent. In fact after completing the APS Upgrade CDR, we found a Wikibooks on LaTeX with a section dealing with collaborative writing [1], which was equivalent to what is written here.

## USING LaTeX

LaTeX is a formatting language that was created to typeset aesthetically-pleasing math papers and to handle numerical references to figures, tables and citations. It has been embraced by authors in many scientific fields, and instruction manuals of various levels abound (experts should use [2]). Many Ph.D.s have used LaTeX to write their thesis. The input file is plain text with commands that tell LaTeX in a high-level way how to format text, which can be readily seen in the JACoW proceedings template files [3].

LaTeX by default has good aesthetics, which is a good enough reason to adopt it over Microsoft Word and OpenOffice. Usually, the formatting doesn't need to be adjusted by the author. Often a style file is provided by a

* Work supported by the U.S. Department of Energy,Office of Science, Office of Basic Energy Sciences, under Contract No. DE-AC02-06CH11357.

† lemery@aps.anl.gov

journal editor, which means the author can concentrate on content.

The APS Upgrade CDR was assembled into a book by joining a large chapter composed in LaTeX with others composed in Word. This mixture is possible since the final chapter pdf files can be joined with pdf tools such as `pdftk` [4]. Given that LaTeX and Word must live together, and that the style format management of LaTeX is more flexible than Word's, the Word formatting style is imposed on the LaTeX chapter. Luckily for us, many unorthodox styles were encountered by outside expert LaTeX users and solutions were made available on the searchable Comprehensive TeX Archive Network [5].

A companion program BibTeX handles the typesetting of bibliographies. A plain text file provides the information, and a style file arranges the information in the final document. Bibliography formatting differ drastically between journals, so it is important to keep the content separate from the formatting (obviously the same goes for LaTeX ). In our case a bibliography style format required by the Word-centric editors did not conform to any established ones (BibTeX users over the years have developed dozens of BibTeX style files conforming to several journals). Thus we had to customize, using instructions given in [2], a closely matching BibTeX style file, which was then available for future customizations. This has proven invaluable as the bibliography style became more specific over the course of the project. Such changes in Word documents would be very time-consuming to make.

In collaborations it is important for authors to adopt consistent policies in editing input text files and naming of certain things. When typing paragraphs, it is best to have sentences wrap around at some column number and put a line break at the end. This improved viewing differences between versions with `tkdiff` [6]. When using BibTeX it is recommended to use a consistent policy in creating keys, such as Author+year+publication.

Large documents usually have a structure of chapters, sections, subsections, and so on. It is natural therefore to split the contents into many files, one for each section or subsection. One should use the LaTeX `\input` command to read in files at appropriate places from the higher-level section. The files should be arranged in a directory tree, mirroring the document structure, one level for chapters and another for sections (two levels is enough). The image files for figures should be in a lower level directory called `figures`, say. Depending on whether we run LaTeX or `pdflatex`, the image files must be in either EPS or PDF formats, respectively. Some image-producing software may not know how to make a EPS file. In this case pdf files are preferred, since they retain the high resolution of vector graphics available in EPS. An additional advantage of `pdflatex` is that it accepts most common bitmap formats, although these of course tend to have poor quality compared to vector graphics.

Formatting in LaTeX is done through commands that precede or enclose the text. There are many commands to choose from to produce a basic and very readable document. Special commands can be defined through the use of style files (packages) that are declared at the beginning of the file, the preamble. This area of the file can potentially be crowded by many package declaration and special variable settings. In our application some 20 or so packages are declared along with dozens of variable assignments. To remove the clutter, we move all of this into a new style file and call this package as a single line in the document.

Some particular LaTeX packages that were useful for our large documents were: `latexsym` for special symbols, `hyperref` for hyperlinks to figures and tables, `fancyhdr` for the sometimes busy/unusual headers and footers required, `chappg` for page numbers composed of chapter and page within chapter, `longtable` for tables that are longer than one page, `multirow` for joining cells in a table across rows, `chapterfolder` for reading of chapters and sections in a directory structure, `secsty` for modifying section headers style, `subfig` for multiple subfigures in a figure, `color` for color in table cells, `threeparttable` for adding tablenotes to text in tables, `rotating` for rotating large figures or tables so that they fit alone in a single page, and finally `caption` for arbitrary styles for captions in figures and tables.

Some of the packages were not sufficient to match format invented by the Word-centric editors, so they were modified, renamed and placed in the `CVS` repository: `biblio` for bibliography header modification, `chngcntr` for special numbering of equations, `cite` for grouping of citation numbers, `bibunits` for making a separate bibliography for each chapter or section. One odd style requirement was to indent the first paragraph of every section, which required package `indentfirst`. Last but not least the Greek letter micron needed to conform to Microsoft's upright micron using the package `pifont`.

The above list may seem like a complicated way to meet format requirements. However the packages are easy to find in [5], and they are for the most part compatible with each other. If they are not, then some editing and renaming of style files or workarounds are necessary.

## USING CVS

A version control system maintains versions of a software project or data. The software files and data would generally reside on a server (the repository). A user would obtain a complete copy of the project with a client-like checkout action. A user would then modify the local copy of a particular file, and then issue a "commit" command to create a new version of the file in the repository. Now, when other users would checkout or update their copies from the repository, the new version of the above file appears in their local working areas. In general even if two authors would edit the same file at the same time, the sections of the files that are modified would be hopefully independent of each other and not cause a problem. Otherwise one of the authors would have to repeat his edits on the other's current

version.

At APS we generally use Concurrent Versions System (CVS) as our versioning system for EPICS and other accelerator software. Because of our familiarity with CVS we adopted it for versioning large multi-author documents.

For those uncomfortable with command lines, tkcvs [7] is a GUI interface to CVS.

In versioning systems one can make comparisons between two versions using, say, the tkdiff utility.

Tagging is a way to save the version numbers of all files in the system. This is useful when one complete a major step in paper preparation, that is all the files are synchronized at the same stage of preparation. When sending a document to reviewers or referee, one could assign a tag to the current version, and continue to make improvements for future version. Also if one has to "downgrade" a paper to fit a conference proceedings 3-page limit, one can still tag a longer version that can be archived privately.

To summarize the workflow: 1) initial checkout (to get a copy), 2) update copies, 3) edit copy, 4) review work, i.e., compile to check syntax and results, and 5) commit the changes (give a descriptive comment).

All source files for compiling LaTeX must be put under CVS including sources that are used for making EPS figures, for example, an OpenOffice drawing file. Derived files, such as PS and PDF files don't need to be in the repository. The LaTeX and BibTeX style files and scripts that use them certainly must be in the repository.

The revision system Apache Subversion (svn) [8] is a newer revision system that is meant to be a replacement to CVS. Presently CVS does everything that we require for publishing a multi-author document: 1) network accessibility, 2) saving of all versions, 3) comparison of different versions, and 4) tagging of milestones.

For last-minute editing of a document or proceedings paper, having an editor as a collaborator would be very beneficial, since the iteration process is more efficient than passing email documents pack and forth. (For editors and contributors who use Windows-based computers, use of a virtual machine running Linux is an excellent way to get access to free software and work efficiently with scientists and engineers using the system described here.)

## USING MAKEFILE

The make system is used in software engineering to compile a system of code according to dependency rules defined by the Makefile file written by the user. Usually this file defines a series of compilation steps required to produce one or more final executables. The goal is to hide from the user the complexity of running commands on the command line. In the case of a LaTeX document, the dependency rules run the latex or pdflatex and the bibtex executables to obtain the final product, the pdf file.

The file Makefile is located in the directory of the document. Typing "make" in that directory will launch the necessary compilation. The Makefile also includes definitions of macros that can be used to set the search path of style files for latex and bibtex commands. Makefile can also direct the execution of scripts that run bibtex on each chapter bibliographies. Note that the Makefile must also be stored in the CVS software repository. Finally the make command can be programmed as a special command in the tkcvs GUI.

For a multi-chapter or multi-section document it is often useful and time-saving to compile only a lower level portion of the document. Thus each directory level has its ownMakefile. This entailed writing a LaTeX source file in each directory with the same preamble as the file in the top directory. (For compilation of whole document the "included" chapter or section files don't have preambles.)

## OTHER ISSUES

Editors may want to compare two revisions of the full document to make sure editing instructions have been followed. Two pdf files may be compared using AcrobatPro selecting the command Advanced →Compare documents. One could also export the file in text format and run the tkdiff command on the plain text. The best approach, as indicated above, is for the editor to use the CVS system directly, coupled with tkcvs.

As mentioned above, several contributions to this conference have made use of LaTeX CVS, and make. A repository structure was set up specifically for conference papers and will be used by many accelerator scientists at APS from here on. In contrast to previous years, when we shared drafts via email and often had confusion and conflicts with multiple authors changing documents at the same time, coauthoring of papers was very smooth using this system.

One issue with submission of LaTeX conference papers and journal articles is that most do not accept BibTeXfiles, but require the bibliography to be embedded in the document. Also, there are often naming conventions for the files, including figures. We have written a simple script that parses the LaTeX document and BibTeX output to create a new set of files that are ready for submission.

## REFERENCES

[1] A. Henningsen. http://en.wikibooks.org/wiki/LaTeXCollaborative_Writing_of_LaTeX_Documents.

[2] F. Mittelbach et al. The LaTeX Companion, 2nd ed. Addison-Wesley, 2004.

[3] Templates for Papers. http://accelconf.web.cern.ch/accelconf/jacow/templates/templates.htm.

[4] http://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/.

[5] http://www.ctan.org/search.html#byDescription.

[6] http://sourceforge.net/projects/tkdiff/.

[7] http://www.twobarleycorns.net/tkcvs.html.

[8] http://subversion.apache.org/.