UNIVERSITY OF WATERLOO
Faculty of Engineering

DESIGN AND EVOLUTION OF A LaTeX DOCUMENT CLASS FOR WORK REPORTS

Waterloo Maple Inc.
Waterloo, ON

Prepared by
Simon Law
ID #99168374
sfllaw@engmail.uwaterloo.ca
3B Computer Engineering

11 May 2003

Simon Law
149 Kingsdale Ave.,
Toronto, ON M2N 3W8

11 May 2003

Dr. A. Vannelli, Chair
E&CE Department,
University of Waterloo,
Waterloo, ON N2L 3G1

Dear Dr. A. Vannelli:

Re: Submission of my work term report.

I have just completed my fifth work term, following my 3B term. Please find enclosed my fourth work term report entitled: "Design and Evolution of a LaTeX document class for work reports" for the Build Team at Waterloo Maple Inc. My departmental manager was Victor Luk and our group's primary task was maintaining and developing the Maple build system.

This report focuses on uw-wkrpt, the unofficial work report documentation class that I wrote. It explores why I wrote it, how I designed it, and how it has evolved over the past two years. It is written for fellow co-op students who work at Waterloo Maple Inc., as well as students at other companies. The report is also written for programmers who wish to edit uw-wkrpt, so that they may have some insight in the challenges I overcame.

I have had no direct assistance from anyone. I do wish to thank Leslie Lamport and Donald E. Knuth for inventing and implementing such marvellous typesetting tools. I thank Frank Mittlebach and David Carlise for their innovative doc and docstrip packages. I thank Karl Berry, Robin Fairbarns, David Arseneau and Stepan Kasal for their critiquing of my early work in TeX. I would also like to give special thanks to fellow Waterloo students: Ryan Leslie, Andrew Stannard, David Meyers, James Morrison, Andrew Miklas, Jang Han Goo, and Mike Jarrett for their thoughtful comments and helpful suggestions. They have helped make uw-wkrpt easier to use, better made, and more correct.

I hereby confirm that I have received no further help other than what is mentioned above in writing this report. I also confirm that this report has not been previously submitted for academic credit at this or any other academic institution.

Yours sincerely,

_____

Simon Law, 99168374

Encl.

Contributions

At Waterloo Maple Inc., I worked in the Build Team headed by Victor Luk, in a branch of the Research and Development department under Vice-President Paul Mansfield. My co-worker and manager was Scott Searle.

The Build Team was responsible for ensuring that the nightly development builds ran smoothly and reliably. This consisted of maintaining a set of scripts that ran through the build process; operating and caring for a farm of machines that ran nightly builds; and to make incremental improvements to the system. Our team was also responsible for creating installers for our products.

Over the course of four months, I was the person in charge of the build scripts. This implied that I would fix bugs and implement new behaviour as required. I developed a more efficient way to copy finished binaries, fixed the system for distributing binaries to University labs, documented the previously undocumented system, and cleaned up a lot of ancient code. The build scripts are tied directly to Maple's development model: without a reliable nightly build with an associated test run, developers would be working with old binaries that exhibited problems which have been fixed. Productivity always suffered whenever a build run was interrupted.

In the process of maintaining these scripts, I performed minor system administration on our build farm; supplementing the efforts of our UNIX administrator. This led to the creation of `mrsh`, a "multiple remote shell" program that can be used to run the same command line on all our build machines.

In addition, I built an end-to-end build system for another development group. Using Apache Ant, I designed and implemented a system that could

compile, test, and deploy Java applications based on the last good native binaries generated by our build farms. This was a vast improvement over the previous system of manually fetching the binaries from a central repository. The improvement in feedback was felt immediately. The Java developers did not have to wait overnight to see if their code worked, the system would notify them within half-an-hour if anything failed to compile.

Finally, I had some small input in the installation system for the upcoming Maple release. Scott Searle bounced ideas off me, and we solved interesting technical problems.

Early in the term, I mentioned the topic of this work report to Victor Luk, and was encouraged to pursue it. I hope that the uw-wkrpt document class is employed by co-op students at Waterloo Maple, since they will benefit from the automated typesetting that is provided by this document class, and LaTeX in general. Waterloo Maple will benefit, since they now have a document class to provide to future co-op students, thereby reducing the time they spend on formatting reports. As well, this will encourage students to learn and use LaTeX: the typesetting system used by their primary marketing demographic.

Summary

Over my four year stay at the University of Waterloo, I have written three work reports. Each of these work reports must follow a set of work report guidelines which are provided by the faculty or department, in my case, the Electrical & Computer Engineering guidelines. These guidelines are designed to aid work term report markers in reading and grading each report. As such, it is important that students are able to conform to these guidelines accurately and consistently.

After my first work term report, I realised that a lot of effort went into ensuring that my report followed the guidelines exactly. It occurred to me that a computer program could do the job in a fraction of the time. As a Computer Engineering student, I realised that I could implement such a program to remove the tedium from work reports.

By the time I wrote my second work report, I already had a preliminary design of this program. The primary design goals were to create a program that would know how to adhere strictly to a set of guidelines, to work as an author expects, to be simple, and to typeset text beautifully. I realised that a good language to use would be the LaTeX macro language, a semantic mark-up language that could express the structure of a paper. Since LaTeX is implemented on top of the powerful TeX typesetting language, it would result in æsthetically pleasing reports.

Over the course of two years, the program was developed and refined into a system that can understand four different work term report guidelines, and can typeset them all. This report was created using it, allowing me to concentrate

on writing well instead of manually formatting.

Much time and effort has been spent to get the program to its present state. It can be extended to format work reports according to the any guidelines, and it formats them is a well-documented, modular manner. Although there is much to improve, there are over a dozen students who have successfully employed it to write their work reports.

I encourage all students to try this program: to save time, effort, and frustration. I am committed to maintaining it; adding functionality so that it will be easier to use and so that it will support more work report styles. I highly recommend that the University adopt this program officially, so that more students will be exposed to its capabilities.

Conclusions

It has taken much effort to write a program that can format work reports according to various work report guidelines. However, it is safe to say that this effort has paid off. The programs is well-documented and modular; appearing to follow the guidelines properly. In addition, there are at least a dozen happy users; more will surely follow.

However, the success is still limited. Although the program is technically excellent at formatting work reports, there are certain things that must be done to increase the usability and popularity of this program. A graphical user-interface would make the system accessible to authors who do not wish to learn a new programming language. An automatic installation system would also help, as the program is not easy to install and configure.

It is necessary to convince the Co-operative Education & Career Services (CECS) department that this program would be useful to students. If it were officially sponsored by CECS, then it would be easier to garner acceptance from students. No doubt that this hurdle will be far more difficult, as it is a social, not technical, issue.

Finally, one must consider what would happen if widespread adoption of this program were to occur. It would have to be adapted to fit all the possible work report styles at the University. In addition, there will probably need to be support for the students, some of whom are not technically-minded. This is a problem which cannot be easily solved.

Recommendations

I recommend that any student with the ability to program, or the gumption to learn, should use this program to format work reports. The program I wrote works on many platforms, including Microsoft Windows, so software incompatibilities should not be a substantial barrier.

As an employer of University of Waterloo co-op students, I highly recommend that Waterloo Maple encourage them to use this program for their work reports. Since the students are all technically competent, it should be quite simple to format a beautiful document without much fuss, saving time and trouble.

Finally, I recommend that the University consider adopting this program officially. As the author of the program, I feel it is highly important to concentrate on good writing, which indicates thought and consideration. Being freed from the need to typeset documents by hand can encourage this. I am committed to improving the program so that all students can enjoy its benefits. This means that third parties need spend neither time, money, nor manpower to improve it.

Table of Contents

List of Figures

List of Tables

1 Introduction

In the early years of mainstream computing, computers were being employed for a novel purpose—writing documents. By the mid 1960s, the Massachusetts Institute of Technology was using the `runoff` program, which would format text for print-outs. By the early 1970s, J. F. Ossanna rewrote it to justify[1] UNIX operating system, and called his version `roff`. Mr. Ossanna continued to develop the system, eventually creating `nroff`. When Bell Laboratories acquired a phototypesetter, he created `troff` to drive it. With this new software, entire books could be printed without any need for lead type, this process is called **computer typesetting**.

In 1977 at Stanford University, Prof. Donald Knuth was reviewing the galley proofs of his *magnum opus*, "The Art of Computer Programming." He had prepared the second edition of Volume 2, but the text looked horrible. His publisher had switched to a computer typesetting system, and it was too primitive. The fonts were ugly, and the mathematical typesetting was especially poor. Prof. Knuth sat down to solve the problem once and for all, and wrote TEX and METAFONT over the course of ten years [2].

In 1983, Leslie Lamport at Digital Equipment Corp. realised that TEX was too low-level for actual writing. He developed a series of macros in the TEX programming language that would provide a **semantic layer** over the document. So instead of instructing the computer to "leave some vertical space, increase the font size, switch to a bold style font, and write 'Introduction'," the author merely has to instruct the computer to "begin a section entitled

[1] Ken Thompson and Dennis Ritchie needed a reason continue developing UNIX, so they proposed that `roff` be used by the AT&T patents division for all their typesetting.

'Introduction'." He released this series of macros as LaTeX and published a manual describing his new system [3].

Today, TeX and LaTeX are widely regarded, and used, within the academic community. Prestigious journals in Mathematics, Science and Engineering use them to typeset their publications. In fact, the University of Waterloo supports LaTeX as a standard document format for submitting thesis papers [4]. It seems reasonable then, for co-operative education students to use LaTeX for their work reports. However, although the Co-operative Education and Career Services (CECS) department has a style [5], it has never been implemented as a **document class**: a programmatic style that instructs LaTeX on how to format a document. Moreover, many faculties or departments embrace their own work report styles, which may be similar to the one published by CECS.

It seems quite apt to use a computer to automate the formatting and layout of work reports. Many students choose to use a word processor (*e.g.* Microsoft Word, Corel WordPerfect, or Lotus Word Pro.) Others use TeX or LaTeX and apply manual corrections to make their reports conform to the requisite guidelines. In 2002, I wrote a document class called uw-ece-workreport to use in my second work report. In the latest incarnation, it has been renamed uw-wkrpt to reflect its more general nature. As far as I know, it is the most correct implementation of work report guidelines for CECS-style reports [5], for Faculty of Mathematics reports [6], Electrical & Computer Engineering reports [1], and Software Engineering reports [7]. Its design, implementation, and evolution are described in this paper.

2 Requirements

After my first work report, I felt that formatting my work report manually seemed to be a large expenditure of effort. I read through the provided specifications, and subjected my report to great scrutiny. This took much effort, and I realised that there must be a better way. As a computer engineering student, I would write a computer program to do this for me.

During my third work term, in the fall of 2001, I was about to write my second work report. I remembered my idea of using a computer program, and decided to survey some of my fellow Electrical & Computer Engineering (E&CE) classmates. From their informal responses, I drew up a list of four major requirements.

The primary requirement, the most fundamental aspect, of this document class is to take care of all formatting guidelines. Initially, these formatting guidelines were the "E&CE work term report guidelines" [1] supplemented by the "Co-operative education student reference manual" [5].

Another important requirement is to embrace the principle of least surprise. That is, extra whitespace should be automatically corrected instead of forcing the author to do it. This should allow the author to concentrate on writing a good report, instead of having to worry over conventions and details.

A corollary of this requirement is the need to behave as expected. When there is already a conventional, or idiomatic, way of performing a task, the program should be faithful to this method. For example, if the program were implemented in Microsoft Word, it would use the "styles" feature denote section headings.

The third requirement is simplicity. It should not be difficult to start a new sub-section. It should not be difficult to create a conforming title page. After learning some stylistic conventions, it should be simple to write the report. This implies that the document class should be usable by the author, and that unnecessary complications should be hidden. It also implies that it should be difficult to violate the work report guidelines, since the most natural way of typesetting some text would be the correct way.

The final requirement is beautiful typesetting. It is a goal to achieve a document that is pleasant to look at, and formatted well. It should adhere to typesetting conventions unless otherwise contradicted by the work report guidelines.

3 Design

Design is quite important for any application. Thoughtful consideration turns out a better program, and a better product. Sadly enough, I was under time pressure and had to develop my document class in parallel with my second work report. This could have led to some crippling architecture flaws, but fortunately it seems that the system has evaded them.

3.1 LaTeX as a platform

Although I was quite familiar with the WordPerfect word processor, I became quite convinced that it would not fare well as a basis for the macro system after an initial prototype. Although the user interface was optimised for entering text, it was also optimised for adjusting the formatting of the text. This rubbed against the grain of my requirements, which would be to remove the need for hand-formatting.

I had been introduced to LaTeX by Prof. Victor Quintana, and came to the conclusion that it may be appropriate. It is a **mark-up language**, such that text has commands embedded within which commands the computer to perform a certain tasks. For instance, the words "mark-up language" were boldfaced with the following command: `\textbf{mark-up language}`. The `\textbf` command indicated that boldface text should be used, and the braces contained the text to be boldfaced. LaTeX is a complete programming language for manipulating documents, so it seemed like a good choice for implementing a set of formatting rules, such as work report guidelines.

The decision to use LaTeX was one of the best, and worst design choices

I made. It was a great decision since the platform on which it was build was well-defined and stable. TEX has been stable since 1995, METAFONT has been stable since 1998 and LATEX has been stable since 2001. LATEX is a well-respected system used throughout academia, and there are a plethora of resources available. As well, the syntax is relatively clean, and it facilitates documents that expressed the semantics as opposed to the raw typesetting.

However, it has one major disadvantage: LATEX is not a word processor. Word processors are transparent to use, when the author types text, it appears much like it would in the final output. A text-based programming language is used to instruct LATEX, which means that the author does not receive immediate feedback on the text input. In addition, syntax errors are inevitably made in LATEX input, something that does not occur with word processors.

Still, the trade-off for rapid development over ease-of-use has been rather profitable. See Section 7 for feedback from users. As well, it seems fairly simple to add a graphical interface using LYX, a graphical front-end for the LATEX system (see Figure 1.)

3.2 Adhering to guidelines

There is an incredible amount of detail that is in the work term report guidelines. In fact, many faculties and programs have a simple checklist which they employ to check over work reports. Without these lists, it would be easy to forget a technical requirement. From my conversations with fellow classmates, it is not uncommon for reports to be resubmitted due to simple errors, caused by a lapse in memory, or a simple mis-understanding.

Most requirements are rather simple for a computer to implement. As an

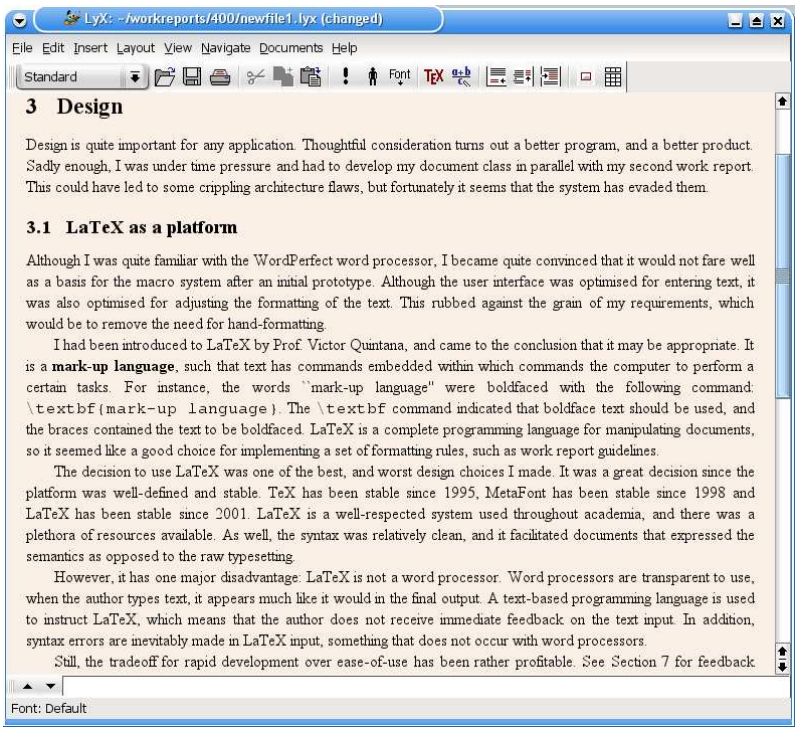

Figure 1: LyX, a graphical front-end.

example, examine the following requirement taken from the "E&CE work term report guidelines".

> "Title Page
>
> > It must contain the following information: university and faculty names, title of report, name and location of employer, student's name, student's id number, student's engmail userid, previous academic term, completion date of report, and *confidential-1* if a confidential report." [1]

This requirement is not difficult to satisfy. All that is necessary is to ensure that the author has typed in all the requisite information, and to lay it out as required. As you can see from Figure 2, a title page is laid out according to the standard methods. The text is represented by boxes. For example, the

name of the University is stored in the variable `\@school`, which is positioned in the top-centre of the page.

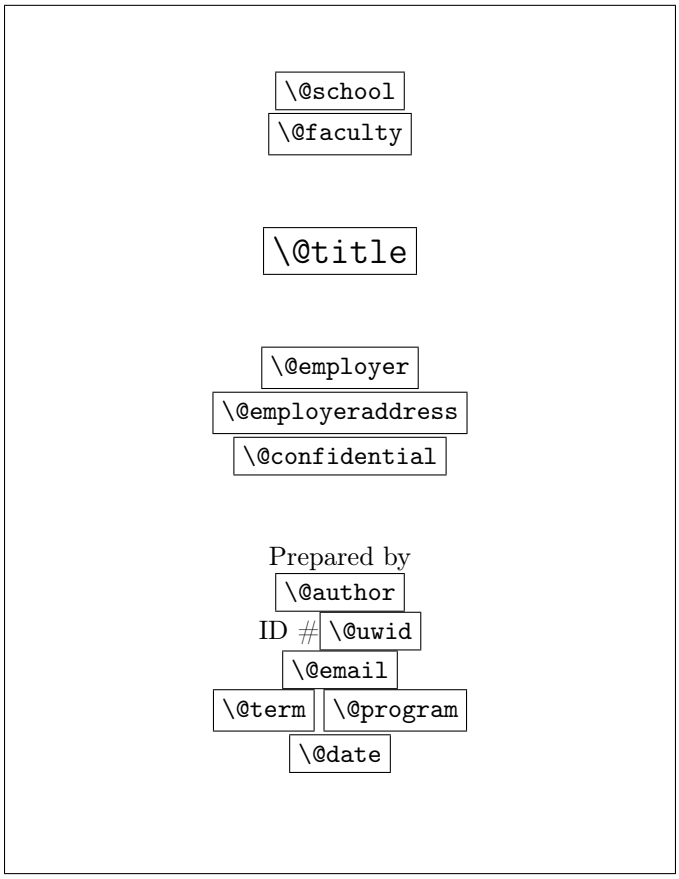

Figure 2: The design of a title page.

Likewise, the requirement that there is "... consistent figure numbering in the report body, ..." [1] is difficult for a human to track, since he may move a figure and forget to update every figure number in the report; but this task is trivial for a computer.

3.3 Principle of least surprise

The LaTeX typesetting system has certain conventions that its users are used to. For example, to typeset the a title page, the author uses the `\maketitle`

command. This uses the information provided by the `\author`, `\title`, and `\date` commands.

When implementing the document class, I could have either defined a new command to replace `\maketitle` or I could redefine `\maketitle`.

Defining new commands seems to be advantageous. Since it has a different name, there is no expectation that it behaves like the old command. It also reduces the possibility of another package with a conflicting overrides. For instance, the titling package can be loaded to produce custom title pages, but it replaces `\maketitle` with its own command. By using another command name, say `\uwwkrptmaketitle`, loading the titling package would not do any harm.

On the other hand, emulating the conventional syntax means that authors can use commands from the familiar LaTeX syntax. In the case of `\maketitle`, the document class would be doing what the author expects if it creates a title page.

In order to minimise the chance that the author will load a foreign package that overloads `\maketitle` again, the document class needs to provide a title page that meets all of the author's needs. Nominally, this requires that the generate title page conform to work report guidelines.

In addition, this design reduces the learning curve to use the document class. For an experienced LaTeX author does not have to learn new commands or behaviour; and an author new to LaTeX can rely on published tutorials. The transparency introduced by overloading standard LaTeX commands seems to be an acceptable trade-off.

3.4 Simplicity

The document class must be simple to use. Alas, this may be a failing in the current implementation. Although it is quite simple to perform certain tasks, other tasks are far more difficult than they ought to be.

For instance, it is quite simple to start a new section that conforms to the guidelines. It must start on a new page, have the correct section numbering, and appear in the Table of Contents. To start a section entitled "Introduction," the author merely types `\section{Introduction}`.

However, inserting a picture is more difficult. At the beginning of a document, the author must issue the `\usepackage{graphics}` command.[2] Within the document, a picture must be wrapped within a **float**, which is an object that can "float" across pages in the document. Its position is not fixed, and will move to the most æsthetic position possible. Then, the picture must be centred horizontally. Finally, the picture must be imported, and scaled to fit. This results in the following code:

```
\begin{figure}
  \centering \resizebox{3in}{!}{\includegraphics{picture}}
  \caption{The caption}
  \label{fig:example}
\end{figure}
```

It is evident to see that this is not easy to remember, due to the unnecessarily complex syntax. Alas, the extreme power and flexibility of the LaTeX system becomes visible to the author as excess verbiage.

My design ignored this particular aspect of the system in order to reduce development time. Authors who are familiar with LaTeX will already know

[2] This is not necessarily true, since the author could write `\usepackage{graphicx}` for an extended version of the graphics package, but that may not be installed on all systems.

the standard syntax for figure placement, and use it competently. As well, it seems that there may be a better way to specify the complexities of figure and table placement by using a graphical front-end to the system. This is discussed further in Section 6. As a temporary measure, I included sample documents that were typeset according to the various styles supported by the document class. These samples could be used as templates for authors who were unfamiliar with the LaTeX language.

3.5 Beautiful typesetting

Knuth's TeX system is renowned for its ability to automatically typeset beautiful text. Its line breaking algorithm is still far better than that of popular word processors; the same applies to its page breaking algorithm. With the addition of micro-typographic extensions in Hàn Thế Thành's pdfTeX [8], the quality of typesetting is vastly improved.

However, there are always some caveats. Since the work term report guidelines are designed such that markers have space to write comments, they require authors to "Use one-and-one-half or double spacing throughout." [5] This is highly discouraged in the typesetting industry, as it disrupts the flow of text.

Another typographical error is the size of the margins: "Leave a margin of at least 3.8 cm. (1.5 in.) on both the left and right sides of the page to allow for binding and for the evaluator's comments." [5] Other faculties require a specific length. This impinges on the standard line length of 66 characters.

Nevertheless, following the guidelines strictly is more important than optimal typesetting. This seems like an acceptable trade-off, since many faculties will reject a report that does not follow their guidelines.

4 Version 1.0

The first public release of the document class occurred shortly before submitting my third work report. In Fall 2001, I wrote my second work report in LaTeX. Fortunately, I had pursued a reasonably clean design, and had a report that conformed to the E&CE guidelines [1]. By the middle of the next work term, I realised that I could re-use my previous efforts for my next work report. Approximately one month was spent removing code that was specific to my work report and writing a sample work report to serve as a template. After which, I placed it on the Internet as uw-ece-workreport on 3 August 2002.

This version already implemented the majority of the requirements, and could produce satisfactorily formatted reports. However, I had little experience with LaTeX, and the code I produced was not idiomatic and rather convoluted.

Although the documentation for the initial document class was quite sparse, the sample document was sufficient for others to use it. David Meyers and Ryan Leslie both provided constructive feedback on how the document class could be improved.

4.1 Version 1.1

In early January 2003, I was surprised to receive an e-mail from Andrew Stannard, who was using uw-ece-workreport to write his work report. He supplied a list of formatting errors produced by the document class which had, thankfully, escaped previous technical markers. After fixing these minor errors, and adding more examples to the sample work report, version 1.1 was published on 11 January 2003.

5 Version 2.0

Work on the next version began in early March. After reviewing the existing code base, it was evident that several changes had to be made:

- The source code was convoluted, so strategic rewrites should be made.
- The document class was poorly documented, which could be corrected with the assistance of the docstrip package.
- It was E&CE specific, and should be generalised to work for all faculties.

By 27 April 2003, version 2.0 was available to the public. Now renamed to uw-wkrpt to reflect its more general nature, it offered the ability to typeset work reports in four different styles, and was well documented according to LaTeX conventions. It also included four sample work reports, one for each style.

5.1 Version 2.x

Due to feedback from James Morrison and Jang Han Goo, several spelling and formatting mistakes were caught in new code which was added to support Mathematics and Software Engineering students. These were quickly corrected and version 2.1 became available on 30 April 2003.

Version 2.2 was released on 8 May 2003 which was quickly followed by version 2.3 on 9 May, versions 2.4 and 2.5 on 10 May. These were urgent bug fixes for Mathematics and Software Engineering students. Andrew Miklas was essential in isolating these faults. Version 2.5 is included in Appendix A, and is licensed under a General Public License (see Appendix B.)

6 Version 3.0

Although this is my last work report, I still have several goals for the next major release. These will address some of the design trade-offs that were originally made.

To address the simplicity and usability issues, I plan to implement a graphical front-end to the document class, in order to hide its obscure syntax. This should make the class far more accessible, even to people who are not computer savvy. By extending the LyX software package (see Figure 1,) and informing it of any additional syntax, it should be simple to provide an easy-to-use interface for people who are familiar with word processing.

In order to increase the beauty of the generated documents, the next version should take advantage of newer LaTeX and TeX features. These should be able to produce text layout that is far superior to tools that students are normally accustomed to.

Finally, I plan to include support for more faculties (*i.e.* Civil and Environmental Engineering; Mechanical Engineering; Science; *etc.*) This should provide more students with the opportunity to concentrate on conveying ideas, instead of spending time on formatting. It is my goal for this document class adopted officially, so that all students may benefit.

7 Feedback

After the first release of the document class, I have began to receives e-mails which commented upon it. Many authors sent enthusiastic letters:

> "I've used your class today to write my work report, and I just wanted to thank you for creating such a marvellous thing!
>
> Too bad it's my last work report. ;)"
>
> — Jang Han Goo, 3A Computer Science

Others provided constructive critiscm, or feature suggestions to improve the next version of the document class. These were all appreciated and many ideas were incorporated.

As is evident from Table 1, the number of authors using the document class is increasing. I expect this to grow as more students discover it. From the conversations that I have had with students, many of them discovered the document class by word of mouth; a good sign that the program is satisfying its users. There appears to be a great jump in the most recent term, which signifies that it is becoming better known.

Table 1: Students known to be using uw-wkrpt.

Work term		**Users**
2001	Fall	1
2002	Winter	0
	Spring	4
	Fall	3
2003	Winter	10

References

[1] W. M. Loucks PEng, G. H. Freeman, and J. A. Barby PEng, "E&CE work term report guidelines." http://www.ece.uwaterloo.ca/~wtrc/WrkTrmRpt.html (current August 2002).

[2] D. E. Knuth, *Computers & Typesetting*. Reading, MA: Addison-Wesley, 2000.

[3] L. Lamport and D. Bibby (Illustrator), *LaTeX: a document preparation system*. Reading, MA: Addison-Wesley, second ed., 1994.

[4] S. Carr, "LaTeX for theses and large documents." http://ist.uwaterloo.ca/cs/latex/ (current May 2003).

[5] University of Waterloo, Co-operative Education & Career Services, "Co-operative education student reference manual." http://www.cecs.uwaterloo.ca/manual/ (current April 2003).

[6] University of Waterloo, Math undergrad office, "Faculty of mathematics work report guidelines.." http://www.math.uwaterloo.ca/navigation/Current/workreport/index.html (current April 2003).

[7] M. Armstrong, J. Atlee, W. M. Loucks PEng, G. H. Freeman, and J. A. Barby PEng, "Software engineering work term report guidelines." http://www.softeng.uwaterloo.ca/Current/work_report_guidelines.htm (current April 2003).

[8] H. T. Thành, *Micro-typographic extensions to the TeX typesetting system.* PhD thesis, Massaryk University Brno: Faculty of Informatics, October 2000.

Appendix A Source code

This appendix consists of the documentation and source code for the uw-wkrpt document class. This typeset presentation is possible because the document class was written as a **literate program**, using the docstrip package. A literate program combines both source code and documentation so that humans are better able to comprehend its structure.

The documentation embedded within gives a sense of how the document class has changed over time, and what specifications the class tries to follow. I hope it makes for interesting reading.

The source code is freely available from http://www.eng.uwaterloo.ca/~sflllaw/programs/uw-wkrpt/ under the GNU General Public License (see Appendix B.)

N.B. this document is not typeset according to the "E&CE work term report guidelines" since it is a separate document, following its own consistent style.

The uw-wkrpt document class*

Simon Law†

20 May 2003

Contents

1 Introduction

At the University of Waterloo,[1] thousands of undergraduate students participate in the co-operative education program: a partnership between the University and businesses world-wide to provide first-hand experience for students.

As part of this program, students write work reports—both to enrich their own literary skills, and also to provide employers with research that is professional, analytical, and useful.

2 Justification

The Co-operative Education and Career Services (CECS) department mandates certain formatting and stylistic conventions. In addition, each department may

*Version v2.5, last revised 2003/05/10

†sfllaw@engmail.uwaterloo.ca

[1] http://www.uwaterloo.ca/

impose their own conventions above and beyond the general CECS conventions. Work reports are checked to ensure that they conform.

Human beings are fallable, however, and are liable to misinterpret the required conventions. Formatting a document according to fixed rules is something a computer should be apt at doing. Indeed, this LaTeX document class implements the formatting so that is done automatically.

The inquiring mind may wonder, "why choose LaTeX?" We must consider that it is not a common application with which undergraduate students are familiar. Indeed, a word processor is more comfortable to most students. However, implementing the requirements of each style in a word processor is far from simple. Templates and styles are available to the student, but they are neither transparent to use nor easy to implement. LaTeX is a simple, macro based language that can format text without great user effort. It can parse plain text with some sparse semantic tags to provide a decent report. Since it is coupled with the world-renowned TeX typesetting engine, the resulting report is æsthetically pleasing and typeset tastefully.

A simple LaTeX document can be constructed easily [1], with the knowledge of just a few commands. In the example on the following page, it is plain to see that the majority of the document is entered as plain text.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

```
1  \documentclass{article}
2  \begin{document}
3
4  Lorem ipsum dolor sit amet, consectetur
5  adipisicing elit, sed do eiusmod tempor
6  incididunt ut labore et dolore magna aliqua.
7  Ut enim ad minim veniam, quis nostrud
8  exercitation ullamco laboris nisi ut
9  aliquip ex ea commodo consequat.
10
11 \end{document}
```

Each macro is prefixed by a backslash, and is followed by an alphabetic identifier. Parameters to these macros are encased within curly braces. On line 1, we declare that this document uses the article document class, which determines certain formatting options. The rest of the document is created with a `\begin{document}` command, and finished with an `\end{document}`.

Although the layout of the final text is not immediately obvious from the input that is keyed in, the input language is rather legible. One can argue that using a plain text interface allows the author to concentrate on the content of his message, and not the formatting. Since the computer does most of the formatting work, it is only necessary to proofread the document and tweak minor details.

3 A simple document

Sample documents that show the recommended layout are available. You can use these samples as a basis for your own report by removing the generic text, and replacing it with your own. As well, they provide examples for how to typeset common forms. These documents are stored as `uw-wkrpt-`*`faculty`*`.tex`, where *`faculty`* is one of:

`ece` for Electrical and Computer Engineering (E&CE) students, this implements the "E&CE work term report guidelines" [4];

`math` for Mathematics (Math) students, this implements the "Faculty of mathematics work report guidelines" [3];

`se` for Software Engineering (SE) students, this implements the "Software engineering work report guidelines" [5]; or

`cecs` for those students without special guidelines, see section 3.1

3.1 The document class

uw-wkrpt

Every document needs to have a document class, so it must be specified. The simplest work report format is the one required by CECS and specified by Chapter 9 of the "Co-operative education student reference manual." (CESRM) [2]. These guidelines are used by the majority of programs and can be used like so:

```
\documentclass{uw-wkrpt}
```

However, some programs have their own special requirements. Although there are a number of such programs, I have only implemented the guidelines for E&CE [4], Math [3], and SE [5]. To specify these special requirements, we provide an optional argument to the `\documentclass` command. For example, a Math student would use:

```
\documentclass[math]{uw-wkrpt}
```

Notice how `[math]` is enclosed by square brackets. Other valid options are `[ece]` and `[se]`.

As well, the letter of submittal may be formatted in either modified block, or block format. The default is modified block. This can be controlled with the `[modifiedletter]` and `[blockletter]` options.

Note that no text may appear before the `\documentclass` command.

3.2 The preamble

Between the `\documentclass{uw-wkrpt}` command and the `\begin{document}` command is the section known as the preamble. No text may occur here,[2] but commands to set initial values and options are declared at this point.

[2] In fact, if plain text does get put in the preamble LaTeX will complain with the error: `! LaTeX Error: Missing \begin{document}.`

3.2.1 Mandatory values

The following commands define initial values that must be set. These values are used to typeset the title page (see Section 3.3,) and the letter of submittal (see Section 3.3.1.)

`\title` The `\title{`⟨*text*⟩`}` command defines the work report's title. This will be capitalised on the title page, and included in the letter of submittal.

This command is analogous to the standard LaTeX 2_ε command.

`\author` The `\author{`⟨*text*⟩`}` command defines the author's name.

This command is analogous to the standard LaTeX 2_ε command.

`\uwid` The `\uwid{`⟨*text*⟩`}` command defines the author's student identification number.

`\address` The `\address{`⟨*text*⟩`}` command defines the author's home address. Since an address can span multiple lines, each line is separated with a `\\*` command.

```
\address{200 University Ave. W.,\\*
         Waterloo, ON\ \ N2L 3G1}
```

Also note the use of `\␣\␣` to force a double-space between the province and the postal code.

`\employer` The `\employer{`⟨*text*⟩`}` command defines the employer's name. Typically, this will be the company's business name.

`\employeraddress` The `\employeraddress{`⟨*text*⟩`}` command defines the employer's short address, which should merely be the name of the city and province. For example, if the employer is located in Montréal, Québec:

```
\employeraddress{Montr\'eal, QC}
```

or in New York, New York, USA:

```
\employeraddress{New York, NY}
```

or in London, England:

```
\employeraddress{London, UK}
```

`\school` The `\school{`⟨*text*⟩`}` command defines the name of the school the author attends. This should be

```
\school{University of Waterloo}
```

for most students.

`\faculty` The `\faculty{`⟨*text*⟩`}` command defines the faculty or program the author is in.

`\email` The `\email{`⟨*text*⟩`}` command defines the author's e-mail address.

`\term` The `\term{`⟨*text*⟩`}` command defines the previous academic term the author was enrolled in. For instance, if the author has only finished one school term, (*i.e.* she is in stream four), then she would use

```
\term{1A}
```

because she last attended school in her 1A term.

`\program` The `\program{`⟨*text*⟩`}` command defines the author's current program. A student in Computer Science would write

```
\program{Computer Science}
```

`\chair` The `\chair{`⟨*text*⟩`}` command defines the very important person to whom your letter of submittal is submitted. From Section 9.9.1 of the CESRM [2]:

> If this is your first report (except if you are in Arts, Math, AHS, Geography or Science) address your letter to Mr. B. Lumsden, Director, Co-operative Education & Career Services. If it is not your first report or if you are in Arts, Math, Geography or Science, direct your letter to the Department Chair. If you are in AHS, your letter should be addressed to the Associate Dean of your faculty.[3]

`\chairaddress` The `\chairaddress{`⟨*text*⟩`}` command defines the address to which you will send your report. Like the `\address` command, you should break lines appropriately.

3.2.2 Optional values

Some commands are completely optional and do not have to be included in the preamble.

`\date` The `\date{`⟨*text*⟩`}` command defines an arbitrary date for the title page and the letter of submittal. By default, today's date is used.

This is analogous to the standard LaTeX 2_ε command.

`\confidential` The `\confidential{`⟨*text*⟩`}` command defines the confidentiality of the report. Most reports do not require this command. Refer to Section 9.7 of the CESRM [2] for more information. As an example, if the report is rated as "Confidential-1":

```
\confidential{Confidential-1}
```

Please be aware that there are certain restrictions for confidential reports. You must speak with your field co-ordinator or faculty before undertaking a confidential report. Most confidential reports are not marked until the following term. If you work for certain corporations, your work report cannot be confidential. If you are in certain faculties, your work report cannot be confidential. For more information, see section 9.7 of the CESRM [2] and Section 5 of the E&CE [4] and SE [5] guidelines.

There are several levels of confidentiality:

Not confidential These reports can be reviewed and evaluated by one or more markers.

Confidential-1 These reports must be stored safely, and may only be evaluated by one marker. No duplicates may be made.

[3] This statement was quoted on 24 April 2003.

Confidential-2 One particular aspect of the report may be subject to a non-disclosure agreement. This must be negotiated between the employer and a particular marker.

Confidential-3 Confidential data contained in the report has been altered to permit disclosure.

Confidential-4 The report cannot leave the employer and must be evaluated by a fellow employee.

Confidential reports are not eligible for an Outstanding grade. For a detailed discussion of the levels of confidentiality, see "Confidential work term reports" [6].

3.2.3 Accessing values

\theauthor
\thetitle
\theuwid
\theaddress
\theemployer
\theemployeraddress
\theschool
\thefaculty
\theemail
\theterm
\theprogram
\thechair
\thechairaddress
\thedate
\theconfidential

Each of these commands reproduce the text defined by the respective command defined in the previous sections. Although these macros can be used anywhere in the report, they are used primarily in the **letter** environment, see Section 3.3.1.

Here is a more comprehensive example:

```
1  \documentclass{uw-wkrpt}
2  \title{My first work report}
3  \author{J. Doe}
4  % ...more definitions ...
5  \begin{document}
6
7  Hello, my name is \theauthor, and the title
8  of my report is ``\thetitle.''
9
10 \end{document}
```

Hello, my name is J. Doe, and the title of my report is "My first work report."

3.3 The document

document

Any text within the `\begin{document}` and `\end{document}` commands are said to be within the **document** environment. This text will be typeset into the final output, and any text after the environment will be ignored.

\maketitle

To create the title page, the `\maketitle` command is used. This command should be invoked before any other text. All the necessary information is contained upon this page. A clear cover should be used to let this show through.

This command is analogous to the standard LaTeX 2_ε command.

3.3.1 Preliminary pages

\frontmatter

The `\frontmatter` command is used to tell LaTeX that the next sections should be typeset as preliminary pages. This typically involves lower-case roman page numbers.

This command is analogous to the standard LaTeX 2_ε command in the book document class.

`letter` The **letter** environment does most of the difficult work involved in writing the letter of submittal. When `\begin{letter}` is invoked, the headings and salutations are laid out. On the next line, the body of the message should be entered. The environment is closed with the `\end{letter}` command, which generates the boilerplate disclaimer required by the guidelines, and generates the signature block.

The **letter** environment is able to get the information required to generate the address blocks, the date, the salutation and the signature because this information was defined in the preamble, see section 3.2.

The body of the report is required to contain certain information. According to Section 9.9.1 of the CESRM [2], this includes:

- report title (use `\thetitle`)
- report number (first, second, etc.)
- employer (use `\theemployer`)
- previous academic term (use `\theterm`)
- supervisor(s)
- department(s) worked for
- main activity of employer and department
- purpose of report
- acknowledgements and explanation of assistance
- statement of confidentiality, if required

Section 3.3 of the Math [3] guidelines also require that you include:

- your role in the company
- brief description of your duties

As well, you must also left-justify your letter. Although the Math department allows for memorandums of submittal, I do not support their creation.

Section 2 of the E&CE [4] and SE [5] guidelines also require that you:

- state who the report was written for

This environment is analogous to the standard LaTeX 2_ε environment in the letter document class.

`\section` The `\section[`⟨*short*⟩`]{`⟨*text*⟩`}` command is set to suppress any section numbering in the preliminary pages. The ⟨*text*⟩ argument specifies the section heading, and ⟨*short*⟩ specifies the optional short heading for inclusion in the "Table of Contents".

Unlike `\section*{`⟨*text*⟩`}`, these sections are mentioned in the Table of Contents. While Section 9.9.1 of the CESRM [2] states that preliminary pages do not appear in the Table of Contents, this is not an issue since their are nor proper `\section`s in a CESRM report. If there is a "Summary" section, it appears in the Body. For a Math report [3], the "Summary" is the only section in the preliminary pages, and it should be listed in the Table of Contents. For E&CE [4] and SE [5] reports, all preliminary sections are listed.

Section 9.9.3 of the CESRM recommends that section numbers appear only in the body of the report, see Section 3.3.2. This recommendation becomes a requirement in other programs.

As well, each `\section` is printed on a separate page. This is implied by the CESRM, and required for other programs. Since it does not hurt to put them on separate pages, it is always done.

This command is analogous to the standard LaTeX 2_ε command.

`\tableofcontents` `\listoffigures` `\listoftables`

These commands generate a "Table of Contents", "List of Figures" and "List of Tables" respectively. Each table is on a separate page, and contains the appropriate list.

Following Section 9.9.1 of the CESRM [2], the Table of Contents lists all sections, and subsections of a report. Each entry is connected by dotted tab leading to the page number, which is right-aligned.

The "List of Figures" and "List of Tables" are not considered sections, and are in included in the "Table of Contents." For E&CE [4] and SE [5] reports, however, they are considered sections and are listed.

These commands are analogous to the standard LaTeX 2_ε commands.

3.3.2 The body

`\mainmatter` The `\mainmatter` command is used to indicate the body of the report. This turns section numbering back on, and causes an arabic page number to appear on each page.

This command is analogous to the standard LaTeX 2_ε command.

`\section` `\subsection` `\subsubsection`

The sectioning commands here will now provide numbered sections, labelled with the appropriate heading.

1 Primary

1.1 Primier

1.1.1 Primo

2 Secondary

2.1 Deuxième

2.1.1 Secundo

```
\begin{document}

\section{Primary}
\subsection{Primier}
\subsubsection{Primo}
\section{Secondary}
\subsection{Deuxi\‘eme}
\subsubsection{Secundo}

\end{document}
```

These commands are analogous to the standard LaTeX 2_ε commands.

figure
table

The **figure** and **table** environments are used to create a "float" which encapsulates a graphic or a **tabular** environment, respectively.

By defaults, floats try to place themselves at the top of the current page, however, Section 9.9.3 of the CESRM [2] suggests that figures and tables appear only after they are referenced in the text. Other programs require this behaviour. Therefore, a float will now try to place itself immediately after the `\begin{figure}` or `\begin{table}` command. If this is not possible, the float tries to place itself at the end of the current page. If this is still not possible, it will center itself on a dedicated page.

Figures must have their captions below, and tables must have their captions on top. Section 9.9.3 of the CESRM [2]. shows some examples.

These environments are analogous to the standard LaTeX 2_ε environments.

3.3.3 Back matter

\appendix

The `\appendix` command indicates that section numbers should now be reset, and in uppercase letters. That is to say that the first `\section` command will be listed as appendix "A".

Although appendices appear in the back matter, this command should be issues before the `\backmatter` command.

This command is analogous to the standard LaTeX 2_ε command.

\backmatter

The `\backmatter` command is used to indicate the back of the report. This turns section numbering off once more.

This command is analogous to the standard LaTeX 2_ε command.

\bibliography

The `\bibliography` command is used to insert the bibliography, or "References" section. This should come after the `\backmatter` command, and refer to a BibTeX database.

This command is analogous to the standard LaTeX 2_ε command.

4 Implementation

4.1 Parsing options

Since this is a document class, the first thing to do is parse out the options that were passed in. To specify which program this work report is written for, the author passes either `math`, `ece` or `se` options. By default, we use the CESRM guidelines [2].

So, the options are declared, and boolean flags of the form `uwwkrpt@`⟨*program*⟩ are declared.

```
\newif\ifuwwkrpt@math \uwwkrpt@mathfalse
\DeclareOption{math}{%
  \uwwkrpt@mathtrue
  \write10{([math] Mathematics report)}}
\newif\ifuwwkrpt@ece \uwwkrpt@ecefalse
\DeclareOption{ece}{%
  \uwwkrpt@ecetrue
```

```
  \write10{([ece] Electrical and Computer Engineering report)}}
\newif\ifuwwkrpt@se \uwwkrpt@sefalse
\DeclareOption{se}{%
  \uwwkrpt@setrue
  \write10{([se] Software Engineering report)}}
```

Work reports must always be set in 12 pt. type. Warn the author if he specifies smaller type, and use 12 pt. nevertheless.

```
\DeclareOption{10pt}{\ClassWarning{uw-wkrpt}{%
  You requested a 10pt font but reports must be 12pt}}
\DeclareOption{11pt}{\ClassWarning{uw-wkrpt}{%
  You requested a 11pt font but reports must be 12pt}}
```

Finally, we declare a `blockletter` option that formats the letter of submittal in block format. The default letter format is modified block, which correspondes to `modifiedletter`.

```
\newcommand{\@blockletter}{}
\DeclareOption{modifiedletter}{%
  \newcommand{\@blockletter}{}}
\DeclareOption{blockletter}{%
  \newcommand{\@blockletter}{\setlength{\parindent}{0pt}}}
```

All of the options specific to this class are declared. The rest of the options will be passed to the standard LaTeX 2$_\varepsilon$ article document class, the options processed, and the article class loaded.

```
\DeclareOption*{\PassOptionsToClass {\CurrentOption}{article}}
\ProcessOptions
\LoadClass[titlepage,12pt]{article}
```

To parse the required arguments, the ifthen package is loaded. This way, the standard LaTeX facilities can be used instead of the TeX `\if` primitives.

```
\RequirePackage{ifthen}
```

4.2 Page margins

The standard North American paper size is U.S. letter, sized 8.5 by 11 inches. This is the default.

The left and right margins will be set to 1.5 inches wide; the top and bottom margins will be set to 1.0 inches wide. This is required by Section 9.8.5 of the CESRM [2].

We piggy-back on the standard fullpage package, but use one of the internal variables `\FP@margin`, so we need to declare this length if it does not exist.

```
\RequirePackage{fullpage}
\ifx\FP@margin\undefined
  \newlength{\FP@margin}
\fi
\setlength{\FP@margin}{1.5in}
\setlength{\textwidth}{\paperwidth}
\addtolength{\textwidth}{-2\FP@margin}
\setlength{\oddsidemargin}{\FP@margin}
```

```
\addtolength{\oddsidemargin}{-1in}
\setlength{\evensidemargin}{\oddsidemargin}
```

4.3 Spacing

Spacing is rather important in this document, as there are several requirements for line spacing.

To facilitate changing from single-spaced to one-and-half-spaced or double-spaced throughout the document, the setspace package is loaded. See Section 9.8.5 of the CESRM [2].

For Mathematics students, their reports must be double-spaced (See Section 3.1 of the Math guidelines [3].) For Software Engineering students, their reports must be one-and-half-spaced (See Section 2 of the SE guidelines [5].)

```
\RequirePackage{setspace}
\newcommand{\uwwkrpt@spacing}{\doublespacing}
\ifthenelse{\boolean{uwwkrpt@se}}
  {\renewcommand{\uwwkrpt@spacing}{\onehalfspacing}}{}
```

Each paragraph must be followed by a blank line, see Section 9.8.5 of the CESRM [2]. Instead of introducing a completely blank line, which is hideous due to spacing issues, we space each paragraph apart by an ex-height.[4]

```
\newlength{\uwwkrpt@parskip}
\setlength{\uwwkrpt@parskip}{1ex}
\setlength{\parskip}{\uwwkrpt@parskip}
```

4.4 Miscellaneous packages

The url package is also loaded, since it breaks URLs[5] and URIs[6] across lines. However, the default typewriter font is unappealing in plain text, so it has been switched to sans-serif.

```
\RequirePackage{url}
\urlstyle{sf}
```

4.5 Manditory and optional values

We override the standard LaTeX commands `\title`, `\author`, and `\date`.

`\title` The title must be defined, and is therefore enforced. See Section 3.2.1.

```
\renewcommand{\title}[1]{%
  \renewcommand{\@title}{#1}%
  \renewcommand{\@@title}{#1}}
\newcommand{\@@title}{\ClassError{uw-wkrpt}%
  {No \noexpand\title given}{}}
```

[4] This is the height of the lower-case letter 'x'.
[5] Uniform Resource Locators
[6] Uniforce Resource Identifiers

\author The author must be defined, and is therefore enforced. See Section 3.2.1.

```
\renewcommand{\author}[1]{%
  \renewcommand{\@author}{#1}%
  \renewcommand{\@@author}{#1}}
\newcommand{\@@author}{\ClassError{uw-wkrpt}%
  {No \noexpand\author given}{}}
```

\date The date defaults to today's date. This is still an optional command, see Section 3.2.2.

```
\renewcommand{\date}[1]{%
  \renewcommand{\@date}{#1}%
  \renewcommand{\@@date}{#1}}
\newcommand{\@@date}{\today}
```

\uwid
\address
\employer
\employeraddress
\school
\faculty
\email
\term
\program
\chair
\chairaddress

New variables which are defined. These, like the ones above, are used to construct the title page. As well, they can be used to construct the letter of submittal.

The following are manditory values, see Section 3.2.1.

```
\newcommand{\uwid}[1]{\renewcommand{\@uwid}{#1}}
  \newcommand{\@uwid}{\ClassError{uw-wkrpt}%
    {No \noexpand\uwid given}{}}
\newcommand{\address}[1]{\renewcommand{\@address}{#1}}
  \newcommand{\@address}{\ClassError{uw-wkrpt}%
    {No \noexpand\address given}{}}
\newcommand{\employer}[1]{\renewcommand{\@employer}{#1}}
  \newcommand{\@employer}{\ClassError{uw-wkrpt}%
    {No \noexpand\employer given}{}}
\newcommand{\employeraddress}[1]{\renewcommand{\@employeraddress}{#1}}
  \newcommand{\@employeraddress}{\ClassError{uw-wkrpt}%
    {No \noexpand\employeraddress given}{}}
\newcommand{\school}[1]{\renewcommand{\@school}{#1}}
  \newcommand{\@school}{\ClassError{uw-wkrpt}%
    {No \noexpand\school given}{}}
\newcommand{\faculty}[1]{\renewcommand{\@faculty}{#1}}
  \newcommand{\@faculty}{\ClassError{uw-wkrpt}%
    {No \noexpand\faculty given}{}}
\newcommand{\email}[1]{\renewcommand{\@email}{#1}}
  \newcommand{\@email}{\ClassError{uw-wkrpt}%
    {No \noexpand\email given}{}}
\newcommand{\term}[1]{\renewcommand{\@term}{\textsc{\lowercase{#1}}}}
  \newcommand{\@term}{\ClassError{uw-wkrpt}%
    {No \noexpand\term given}{}}
\newcommand{\program}[1]{\renewcommand{\@program}{#1}}
  \newcommand{\@program}{\ClassError{uw-wkrpt}%
    {No \noexpand\program given}{}}
\newcommand{\chair}[1]{\renewcommand{\@chair}{#1}}
  \newcommand{\@chair}{\ClassError{uw-wkrpt}%
    {No \noexpand\chair given}{}}
\newcommand{\chairaddress}[1]{\renewcommand{\@chairaddress}{#1}}
  \newcommand{\@chairaddress}{\ClassError{uw-wkrpt}%
```

```
    {No \noexpand\chairaddress given}{}}
```

\confidential

\confidential is an optional value, see Section 3.2.2. If it is empty, it will be ignored. Since most reports are non-confidential, this is the default value.

```
\newcommand{\confidential}[1]{\renewcommand{\@confidential}{#1}}
  \newcommand{\@confidential}{}
```

\thetitle
\theauthor
\thedate
\theuwid
\theaddress
\theemployer
\theemployeraddress
\theschool
\thefaculty
\theemail
\theterm
\theprogram
\thechair
\thechairaddress

The following commands are defined to access these values of these new variables in case the author wishes to refer to them within the document.

```
\newcommand{\thetitle}{\@@title}
\newcommand{\theauthor}{\@@author}
\newcommand{\theuwid}{\@uwid}
\newcommand{\theaddress}{\@address}
\newcommand{\theemployer}{\@employer}
\newcommand{\theemployeraddress}{\@employeraddress}
\newcommand{\theschool}{\@school}
\newcommand{\thefaculty}{\@faculty}
\newcommand{\theemail}{\@email}
\newcommand{\theterm}{\@term}
\newcommand{\theprogram}{\@program}
\newcommand{\thechair}{\@chair}
\newcommand{\thechairaddress}{\@chairaddress}
\newcommand{\thedate}{\@@date}
\newcommand{\theconfidential}{\@confidential}
```

4.6 Title page

We require the textcase package to provide the \MakeTextUppercase{⟨*text*⟩} command.

```
\RequirePackage{textcase}
```

\maketitle

The title page must be laid out in a certain format, for an example see Figure 1 of Section 9.9.1 of the CESRM [2].

```
\renewcommand{\maketitle}{%
  \begin{titlepage}
  \begin{singlespacing}
  \let\footnotesize\small
  \let\footnoterule\relax
  \let \footnote \thanks
  \begin{center}
    {\large \MakeTextUppercase{\@school} \par \@faculty}%
  \end{center}
  \null\vfill%
  \begin{center}%
    {\huge \MakeTextUppercase{\@title} \par}%
  \end{center}\par
  \null\vfill%
  \begin{center}%
    {\large \@employer\\ \@employeraddress\par \textit{\@confidential}}%
```

```
  \end{center}\par
  \null\vfill%
  \begin{center}%
    {\large
      Prepared by\\
      \begin{tabular}[t]{c}%
        \@author\\
        ID \#\@uwid\\
        \@email\\
        \@term{} \@program
      \end{tabular}\par}%
    {\large \@date \par}%       % Set date in \large size.
  \end{center}
  \@thanks
  \end{singlespacing}
  \end{titlepage}%
```

After defining the title page, commands we no longer need are let go.

```
  \setcounter{footnote}{0}%
  \global\let\thanks\@gobble
  \global\let\maketitle\relax
  \global\let\@thanks\@empty
  \global\let\@author\@empty
  \global\let\@date\@empty
  \global\let\@title\@empty
  \global\let\title\relax
  \global\let\author\relax
  \global\let\date\relax
  \global\let\and\relax
}
```

As well, this page should have no page numbering.

4.7 Letter of submittal

`letter` The **letter** environment simplifies the process of writing a letter of submittal.

```
\newenvironment{letter}{%
```

We turn off page numbering for the letter. We use `\everyvbox` to suppress the page numbering on every page. This is much better than trying to save the page numbering and then restore it.

```
  \everyvbox={\thispagestyle{empty}}%
```

Using `\@setletterpagenum`, we decide on the logical page number for the letter of submittal. For the declaration of this macro, see Section 4.9.

```
  \@setletterpagenum%
```

Due to reasons I cannot comprehend, Section 3.3 of the Math guidelines [3] requires that the letter of submittal be left-justified.

```
  \ifthenelse{\boolean{uwwkrpt@math}}
    {\raggedright}{}
```

Using `\@blockletter`, we decide whether we should be using modified block layout, or full block. For the declaration of this macro, see Section 4.1.

```
\@blockletter%
```

Then, the letter is set to single-spaced, since it is not part of the report; but rather an insert.

```
\singlespacing%
```

The header block is created. First, the author and the author's address; then the current date; then the receiver of the report and his address; and finally the salutation.

```
\noindent\@@author\\\@address\par\noindent%
\@@date \par\noindent%
\@chair, Chair\\*\@chairaddress\par\noindent%
Dear \@chair:%
```

Create a subject line, much like formal business letters of old.

```
\begin{center}\textbf{Re: Submission of my work term report.}\end{center}}
```

The author types her letter, and mentions all the things she is required to mention. See Section 3.3.1 for a full list.

When she is done, she ends the environment. This triggers the disclaimer boilerplate, required by Section 9.9.1 of the CESRM [2].

```
{\par I hereby confirm that I have received no further help other
than what is mentioned above in writing this report.
```

The E&CE [4] and SE [5] guidelines require an additional boilerplate message. I will include this boilerplate in the Math report, even though it does not require it.

```
\ifthenelse{\boolean{uwwkrpt@ece}
            \or \boolean{uwwkrpt@math}
            \or \boolean{uwwkrpt@se}}
  {I also confirm that this report has not been previously submitted
  for academic credit at this or any other academic institution.}
```

In other programs, the following boilerplate is required. See Section 9.9.1 of the CESRM [2].

```
  {This report was written entirely by me and has not received
  any previous academic credit at this or any other institution.}
```

The Faculty of Mathematics has a special request, since they ask employers to perform technical marking. It is included after the legal boilerplate. See Figure 4 of the Math guidelines [3].

```
\ifthenelse{\boolean{uwwkrpt@math}}{%
  \par The Faculty of Mathematics requests that you evaluate this report
  for command of topic and technical content/analysis.  Following your
  assessment, the report, together with your evaluation, will be submitted
  to the Math Undergrad Office for evaluation on campus by qualified
  work report markers.  The combined marks will determine whether the
  report will receive credit and whether it will be considered for an
  award.
```

```
    \par I would like to thank you for your assistance in preparing this
    document.}{}%
```

With the legal requirements completed, the signature block can be generated. The signature line is only 3 inches long and 0.3 in tall, but that should be sufficient for most purposes. To satisfy Section 9.1 of the CESRM [2], the student's name and ID are listed below the signature line.

```
  \par\noindent
  \begin{minipage}{\textwidth}
  \setlength{\parskip}{\uwwkrpt@parskip}
  \vspace*{\uwwkrpt@parskip}
  Yours sincerely,\\*%
  \rule{0in}{0.3in}\\*{\hrule \@width 3in}%
  \noindent\@@author, \@uwid
  \par\noindent
  Encl.%
  \end{minipage}
```

Now that the letter is done, we set the correct page number for pages that follow the letter, and then restore double-spacing.

```
  \@setpostletterpagenum\uwwkrpt@spacing%
```

All the excess variables that were used can now be let go.

```
  \global\let\@@author\@empty
  \global\let\@@title\@empty
  \global\let\@@date\@empty
  \global\let\uwid\relax
  \global\let\@uwid\@empty
  \global\let\email\relax
  \global\let\@email\@empty
  \global\let\employer\relax
  \global\let\@employer\@empty
  \global\let\employeraddress\relax
  \global\let\@employeraddress\@empty
  \global\let\address\relax
  \global\let\@address\@empty
  \global\let\chair\relax
  \global\let\@chair\@empty
  \global\let\chairaddress\relax
  \global\let\@chairaddress\@empty
  \global\let\school\relax
  \global\let\@school\@empty
  \global\let\faculty\relax
  \global\let\@faculty\@empty
  \global\let\term\relax
  \global\let\@term\@empty
  \global\let\program\relax
  \global\let\@program\@empty
  \global\let\confidential\relax
  \global\let\@confidential\@empty
}
```

4.8 Formatting sections

We shall emulate the `\frontmatter`, `\mainmatter`, and `\backmatter` commands from the `book` document class. Front matter pages are numbered with roman numerals; main and back matter pages are numbered with arabic numerals. See Section 9.8.5 of the CESRM [2].

In the front and back matter, the `\section` command does not generate headers with section numbers, but does enter them into the Table of Contents.

`\frontmatter` A fresh page is started, the sections are unnumbered, and the page numbers are lower-case roman numberals.

```
\newcommand{\frontmatter}{%
  \clearpage
  \@notmainsect%
  \pagenumbering{roman}%
  \uwwkrpt@spacing%
}
```

`\mainmatter` A fresh page is started, the sections are numbered, and the page numbers are arabic numerals.

```
\newcommand{\mainmatter}{%
  \clearpage
  \@mainsect%
  \pagenumbering{arabic}%
```

Section 3.1 of the Math guidelines [3] state that numbered sections should not have page breaks between them. This is why we restore `\section` to its original form.

```
  \ifthenelse{\boolean{uwwkrpt@math}}{\let\section\section@rig}
```

`\dotzero` `\@secdotzerostart` `\@secdotzeroend` The Math guidelines [3] imply that numbered `\section`s must be of the form "1.0", not the default "1".

```
  \global\def\dotzero{}
  \global\def\@secdotzerostart##1{}
  \global\def\@secdotzeroend##1{}
  \ifthenelse{\boolean{uwwkrpt@math}}{%
    \renewcommand{\@secdotzerostart}[1]{%
      \let\quad@rig\quad
      \ifthenelse{\equal{##1}{section}}{%
        \renewcommand{\quad}{.0\quad@rig}%
        \renewcommand{\dotzero}{.0}}{\renewcommand{\dotzero}{}}
    }
    \renewcommand{\@secdotzeroend}[1]
      {\ifthenelse{\equal{##1}{section}}{\let\quad\quad@rig}}
  }{}%
}
```

\appendix The page numbers turn Roman here. The only change is that for Math, we eliminate the ".0" trailer, as "Section A.0" looks silly.

```
\let\appendix@rig\appendix
\renewcommand{\appendix}{%
  \@mainsect%
  \ifthenelse{\boolean{uwwkrpt@math}}{%
    \renewcommand{\@secdotzerostart}[1]{\renewcommand{\dotzero}{}}
    \renewcommand{\@secdotzeroend}[1]{}
    }{}%
  \appendix@rig%
}
```

\backmatter A fresh page is started, and the sections are unnumbered.

```
\newcommand{\backmatter}{%
  \clearpage
  \@notmainsect%
  \ifthenelse{\boolean{uwwkrpt@math}}%
    {\renewcommand{\section}{\clearpage\section@rig}}{}%
}
```

summary Much like the **abstract** environment in standard LaTeX, the **summary** environment should be used for typesetting summaries. However, all it does is suppress the section numbers. This environment should only be used by people following the CESRM [2] guidelines. Other programs place their "Executive Summary", "Summary", or "Abstract" in different places.

```
\newenvironment{summary}
  {\@notmainsect}
  {\@mainsect}
```

\@notmainsect This is the macro that turns off section numbers. It does this by redefining certain functions to be much simpler, which implies that it no longer compensates from the prefixed number.

This macro was inspired by the standard LaTeX book class.

```
\newcommand{\@notmainsect}{%
  \def\@sect##1##2##3##4##5##6[##7]##8{%
    \@tempskipa ##5\relax
    \ifdim \@tempskipa>\z@
      \begingroup
        ##6{%
          \@hangfrom{\hskip ##3}%
            \interlinepenalty \@M ##8\@@par}%
      \endgroup
      \csname ##1mark\endcsname{##7}%
      \addcontentsline{toc}{##1}{##7}%
    \else
      \def\@svsechd{%
        ##6{\hskip ##3\relax
        \@svsec ##8}%
```

```
        \csname ##1mark\endcsname{##7}%
        \addcontentsline{toc}{##1}{##7}}%
    \fi
    \@xsect{##5}}%
}
```

`\@mainsect` This is the macro that turns on section numbers. It redefines certain functions to be exactly like their standard forms.

This macro was inspired by the standard LaTeX book class.

```
\newcommand{\@mainsect}{%
  \def\@sect##1##2##3##4##5##6[##7]##8{%
    \ifnum ##2>\c@secnumdepth
      \let\@svsec\@empty
    \else
      \refstepcounter{##1}%
      \@secdotzerostart{##1}
      \protected@edef\@svsec{\@seccntformat{##1}\relax}%
      \@secdotzeroend{##1}
    \fi
    \@tempskipa ##5\relax
    \ifdim \@tempskipa>\z@
      \begingroup
        ##6{%
          \@hangfrom{\hskip ##3\relax\@svsec}%
            \interlinepenalty \@M ##8\@@par}%
      \endgroup
      \csname ##1mark\endcsname{##7}%
      \addcontentsline{toc}{##1}{%
        \ifnum ##2>\c@secnumdepth \else
          \protect\numberline{\csname the##1\endcsname\dotzero}
        \fi
        ##7}%
    \else
      \def\@svsechd{%
        ##6{\hskip ##3\relax
        \@svsec ##8}%
        \csname ##1mark\endcsname{##7}%
        \addcontentsline{toc}{##1}{%
          \ifnum ##2>\c@secnumdepth \else
            \protect\numberline{\csname the##1\endcsname\dotzero}
          \fi
          ##7}}%
    \fi
    \@xsect{##5}}%
}
```

`\section` Every section must start on a separate page. Overloading the `\section` command ensures this. Although the CESRM [2] implies this, Math [3], and Engineering demand this, see section 3.1 of the Math guidelines [3]; and section 2 of the

E&CE [4] and SE [5] guidelines.

```
\let\section@rig\section
\renewcommand{\section}{\clearpage\section@rig}
```

4.9 Tables and Lists

These functions are used to decide what page numbers are used for the front matter section.

`\@setletterpagenum` `\@setpostletterpagenum`

Section 9.9.1, Figure 3, of the CESRM shows an example "Table of Contents" that begins on page i. We will take this as the correct example.

```
\newcommand{\@setletterpagenum}{}
\newcommand{\@setpostletterpagenum}{\setcounter{page}{0}}
```

Section 9.8.5 of the CESRM [2] states that the "Table of Contents" must start on page ii, contradicting Section 9.9.1 above. However, this does not take into account letters of submittal that are longer than one page. The following code is commented out as it makes no sense to follow this requirement.

```
%\newcommand{\@setletterpagenum}{}
%\newcommand{\@setpostletterpagenum}{\setcounter{page}{1}}
```

Section 3.1 of the Math guidelines [3] state that the title page is page i and the "Table of Contents" is page ii, because the letter of submittal is an insert, and not part of the report.

```
\ifthenelse{\boolean{uwwkrpt@math}}{%
  \renewcommand{\@setletterpagenum}{\setcounter{page}{1}}
  \renewcommand{\@setpostletterpagenum}{}
}{}
```

Section 2 of the E&CE [4] and SE [5] guidelines require that the submittal letter be page ii. The "Table of Contents" then follow in logical order, after any preliminary sections.

```
\ifthenelse{\boolean{uwwkrpt@ece} \or \boolean{uwwkrpt@se}}{%
  \renewcommand{\@setletterpagenum}{\setcounter{page}{2}}
  \renewcommand{\@setpostletterpagenum}{}
}{}
```

4.9.1 Table of contents

In LaTeX, the default name for a "Table of Contents" section is "Contents". This is changed to follow Section 9.9.1 of the CESRM [2].

```
\renewcommand{\contentsname}{Table of Contents}
```

`\tableofcontents`

The table should be single-spaced, as `\parskip` should give adequate spacing between items.

```
\let\tableofcontents@rig\tableofcontents
\renewcommand{\tableofcontents}{%
  \clearpage
  \begin{singlespacing}
```

```
  \setlength{\parskip}{0pt}
  \tableofcontents@rig\par
  \end{singlespacing}
}
```

Between the heading of each section, and the right-justified page numbers, there should be dotted tab leaders to lead the eye across the table. By default, sections did not have this behaviour, although subsections did. The following code makes it apply to all.

```
\renewcommand*\l@section[2]{%
    \ifnum \c@tocdepth >\m@ne
      \addpenalty{-\@highpenalty}%
      \vskip 1.0em \@plus\p@
      \setlength\@tempdima{1.5em}%
      \begingroup
        \parindent \z@ \rightskip \@pnumwidth
        \parfillskip -\@pnumwidth
        \leavevmode \bfseries
        \advance\leftskip\@tempdima
        \hskip -\leftskip
        #1\nobreak\
          \leaders\hbox{$\m@th
          \mkern \@dotsep mu\hbox{.}\mkern \@dotsep
          mu$}\hfil\nobreak\hb@xt@\@pnumwidth{\hss #2}\par
        \penalty\@highpenalty
      \endgroup
    \fi%
}
```

4.9.2 Lists of stuff

`\listoffigures@intoc` `\listoftables@intoc`

The "List of Figures" and "List of Tables" should not be mentioned in the Tables of Contents, according to Section 9.9.1 of the CESRM [2]. This is the default behaviour for LaTeX.

```
\newcommand{\listoffigures@intoc}{\relax}
\newcommand{\listoftables@intoc}{\relax}
```

However, Section 2 of the E&CE [4] and the SE [5] guidelines state that they must be listed. So we add the "List of Figures" and "List of Tables" to the "Table of Contents".

```
\ifthenelse{\boolean{uwwkrpt@ece} \or \boolean{uwwkrpt@se}}{%
  \renewcommand{\listoffigures@intoc}{%
    \addcontentsline{toc}{section}{List of Figures}}
  \renewcommand{\listoftables@intoc}{%
    \addcontentsline{toc}{section}{List of Tables}}
}{}
```

`\listoffigures`

We ensure that the "List of Figures" is on a separate page and single-spaced. The spacing provided by `\parskip` is sufficient. Also included is `\listoffigures@intoc`

to respect the settings above.

```
\let\listoffigures@rig\listoffigures
\renewcommand{\listoffigures}{%
  \clearpage
  \begin{singlespacing}
  \listoffigures@rig \listoffigures@intoc%
  \end{singlespacing}
}
```

`\listoftables` The "List of Tables" should behave exactly as the "List of Figures".

```
\let\listoftables@rig\listoftables
\renewcommand{\listoftables}{%
  \clearpage
  \begin{singlespacing}
  \listoftables@rig \listoftables@intoc%
  \end{singlespacing}
}
```

4.10 Tables and figures

Save the original table and figure environments so that they can be overridden. Notice that the `\endtable` command is an implementation dependant part of LaTeX.

```
\let\table@rig\table
\let\endtable@rig\endtable
\let\figure@rig\figure
\let\endfigure@rig\endfigure
```

`figure` `table` According to Section 9.9.3 of the CESRM [2], Figures and tables must be on their own page after they have been referenced in the text. The only way to guarantee this is to change the default ⟨*loc*⟩ argument in `\begin{table}[`⟨*loc*⟩`]` to `[p]`.

```
\renewenvironment{figure}[1][p]{\begin{figure@rig}[#1]}{\end{figure@rig}}
\renewenvironment{table}[1][p]{\begin{table@rig}[#1]}{\end{table@rig}}
```

According to Section 3.4 of the Math guidelines [3]; and Section 2 of the E&CE [4] and the SE [5] guidelines, figures and tables must appear after they are referenced in the text. The only way to guarantee this is to change the default ⟨*loc*⟩ argument to `[htbp]`. See Section 3.3.2 for more information.

```
\ifthenelse{\boolean{uwwkrpt@ece}
            \or \boolean{uwwkrpt@math}
            \or \boolean{uwwkrpt@se}}{%
  \renewenvironment{figure}[1][htbp]{\begin{figure@rig}[#1]}{\end{figure@rig}}
  \renewenvironment{table}[1][htbp]{\begin{table@rig}[#1]}{\end{table@rig}}
}{}
```

4.11 References

Every paper needs a "References" section. Section 9.9.5 of the CESRM [2] does not set any bibliography style. Section 2 of the E&CE [4] and the SE [5] guidelines

require the use of the IEEE Computer Society style [7]. The Math guidelines [3] appear to specify a style similar to the IEEE's.

The IEEE Transactions bibliography style is almost identical to that of the IEEE Computer Society style. An implementation of this ships with almost every LaTeX installation, so we will call that instead. The only difference is that `@ELECTRONIC{}` does not exist as a type of citation, but this can be emulated with `@MISC{}`.

```
\bibliographystyle{ieeetr}
```

In the future, I may consider adding support for the more recent IEEE Transactions style, but only after it ships with the major TeX distributions. As well, I would consider using any styles that the IEEE Computer Society implement.

`\bibliography` Add the References section to the Table of Contents. As well, make it single spaced.

```
\let\bib@rig\bibliography
\renewcommand{\bibliography}[1]{%
  \clearpage
  \begin{singlespacing}
  \bibliography@intoc \bib@rig{#1}\par
  \end{singlespacing}
}
```

`\bibliography@intoc` According to Section 9.9.1 of the CESRM [2], the References section is actually a section that comes before the Appendices.

```
\newcommand{\refn@me}{References}
\newcommand{\bibliography@intoc}{%
  \renewcommand{\refname}{%
    \addtocounter{section}{1}%
    \arabic{section}\hspace{2.5ex}\refn@me%
    \addcontentsline{toc}{section}{%
      \numberline{\arabic{section}}{\refn@me}}%
}
```

However the Math [3], E&CE [4], and SE [5] guidelines state that the `\bibliography{`⟨*file*⟩`}` should come after `\backmatter`, since it should not have a section number.

```
\ifthenelse{\boolean{uwwkrpt@ece}
            \or \boolean{uwwkrpt@math}
            \or \boolean{uwwkrpt@se}}{%
  \renewcommand{\bibliography@intoc}{%
      \addcontentsline{toc}{section}{\refn@me}}%
}{}
```

4.12 Legacy code

The following code is to retain compatibility with the old uw-ece-workreport document class. It merely provides a stub that calls uw-wkrpt with the `[ece]` option.

This document class will be depreciated in uw-wkrpt 3.0.

```
\ClassWarning{uw-ece-workreport}{%
  The 'uw-ece-workreport' class is now ^^J%
  deprecated.  Use '\string\usepackage[ece]{uw-wkrpt}' instead}
\DeclareOption*{\PassOptionsToClass {\CurrentOption}{uw-wkrpt}}
\ProcessOptions
\LoadClass[ece]{uw-wkrpt}
\newcommand{\UWECEWorkReportVersion}{2.0}
```

References

[1] L. Lamport and D. Dibby (Illustrator), *LaTeX: a document preparation system.* Reading, MA: Addison-Wesley, second ed., 1994.

[2] University of Waterloo, Co-operative education & career services, "Co-operative education student reference manual." http://www.cecs.uwaterloo.ca/manual/ (current 24 Apr. 2003.)

[3] University of Waterloo, Math undergrad office, "Faculty of mathematics work report guidelines." http://www.math.uwaterloo.ca/navigation/Current/workreport/index.html (current 26 Apr. 2003.)

[4] W. M. Loucks PEng, G. H. Freeman, and J.A. Bary PEng, "E&CE work term report guidelines." http://www.ece.uwaterloo.ca/~wtrc/WrkTrmRpt.html (current 24 Apr. 2003.)

[5] M. Armstrong, J. Atlee, W. M. Loucks PEng, G. H. Freeman, and J.A. Bary PEng, "Software engineering work report guidelines" http://www.softeng.uwaterloo.ca/Current/work_report_guidelines.htm (current 24 Apr. 2003.)

[6] W. M. Loucks PEng, "Confidential work term reports." http://www.pads.uwaterloo.ca/Wayne.Loucks/Service/confidential/page1.html (current 26 Apr. 2003.)

[7] IEEE Computer Society Press, "CS Style Guide: References" http://www.computer.org/author/style/refer.htm (current 1 Nov. 2001.)

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

A

B

C

D

E

F

H

I

L

M

O

Change History

v2.4

Appendix B GNU General Public License

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced

by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions

of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED

INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and a brief idea of what it does.
Copyright (C) year  name of author

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston,
MA  02111-1307  USA
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) year  name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
This is free software, and you are welcome to redistribute it
under certain conditions; type ‘show c’ for details.
```

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items–whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.