A Report submitted in partial fulfilment of
the regulations governing the award of

the Degree of

BSc (Honours) Programme Name

at the University of Northumbria at Newcastle

Project Report

Your Project Title

Your Name

2016/ 2017

General Computing/Software Engineering Project

Declaration

I declare the following:

1. that the material contained in this dissertation is the end result of my own work and that due acknowledgement has been given in the bibliography and references to ALL sources be they printed, electronic or personal.
2. the Word Count of this Dissertation is ⟨len⟩
   (result of shell command `texcount -total -inc Dissertation.tex`)
3. that unless this dissertation has been confirmed as confidential, I agree to an entire electronic copy or sections of the dissertation to being placed on the eLearning Portal (Blackboard), if deemed appropriate, to allow future students the opportunity to see examples of past dissertations. I understand that if displayed on eLearning Portal it would be made available for no longer than five years and that students would be able to print off copies or download.
4. I agree to my dissertation being submitted to a plagiarism detection service, where it will be stored in a database and compared against work submitted from this or any other School or from other institutions using the service.

   In the event of the service detecting a high degree of similarity between content within the service this will be reported back to my supervisor and second marker, who may decide to undertake further investigation that may ultimately lead to disciplinary actions, should instances of plagiarism be detected.
5. I have read the Northumbria University/Engineering and Environment Policy Statement on Ethics in Research and Consultancy and I confirm that ethical issues have been considered, evaluated and appropriately addressed in this research.

SIGNED: . . . . . .

Acknowledgements

Abstract

A summary of the entire project. From background to conclusions. I recon on about half a page as the upper end of the summary.

This is an example structure for the Terms-of-Reference and the Dissertation. Along with some notes.

You can start by forking the repository on github `https://github.com/dr-alun-moon/cs-dissertation`. Then you have a working copy of this document as a starting point.

Contents

Chapter 1

Introduction

This document is split into several files. This makes the editing easier, if the file management a little harder.

The two principle means of including sub parts into a main document are the LaTeX commands `\input` and `\include`

1.1 Splitting the Input

The input command simply reads that file in, at that point in the main file. The result is as if the inputed file was pasted in at the point of the command.

The include command is a little more complex. The main point to note is that it forces a new page, then reads in the file. It is good foe content that needs to start a fresh page, such as chapters. As a rule of thumb I put each chapter in an included file.

1.1.1 Some tricks.

The Dissertation uses the book class, that gives the Chapter as the top sectional element, followed by section, subsection, paragraph. The Terms-of-reference uses the article class, which starts at the section level.

By writing the ToR as a driver document that handles all the setting up, and then inputing the tor content from a separate file. When it comes to including the tor as an appendix in the dissertation, you can start a chapter then just input the tor content file.

```
\appendix
\chapter{Terms of reference}
\input{tor}
```

1.2 Magic comments

Since LaTeX needs to have a certain amount of setup (known as the preamble), this is the file that needs to be build. From a Makefile or command-line, this is simple enough.

Several editors allow you to trigger the build from within their environment. Here the file you are editing is a subpart and not the mater document that needs to be passed to LaTeX. The editor needs to know which `.tex` file is the master document. Some editors recognise a magic comment placed at the top of a sub-file.

```
%!TeX root=Dissertation.tex
```

This informs the editor that `Dissertation.tex` is the file to build to rebuild the document. I can save changes and just press the key combination set to trigger a LaTeX build, and the editor knows how to do the rest.

Part I

Analysis

Chapter 2

Tools

2.1 LaTeX

I recommend the TeXlive distribution, which can be downloaded from `https://www.tug.org/texlive/`. There are installation packages for Windows, Linux, and Macs here. For Linux machines there is often a copy of texlive in the package repository.

2.2 Editors

Any text editor will do. I wrote my MSc dissertation in `vi`[1] [2]!

In practice, many modern editors have a rich set of tools to help with the editing process. From syntax-highlighting, auto-completion, to rebuilding.

2.3 Atom

The Atom editor is my current editor of choice[3]. As it is a very extensible editor, there are a number of add-ons that help with latex. I currently have:

language-latex `https://atom.io/packages/language-latex` which provides syntax highlighting.

latex `https://atom.io/packages/latex` which provides means of compiling latex documents from within the editor.

pdf-view `https://atom.io/packages/pdf-view` which is a PDF viewer. (I'm not sure about this one, I can't tell if it is too much of a strain on the editor)

[1] `https://en.wikipedia.org/wiki/Vi`
[2] Hardcore Unix use
[3] Having used emacs since the late 1980s for latex

```
apt search texlive
```

Figure 2.1: Searching for texlive on Debian/Ubuntu systems

2.4 PDF viewers

Latex generates pdf files out of the box. There are some good PDF viewers about. The ones below integrate nicely with the Atom latex tools.

2.4.1 Skim – OSX

The viewer for Macs `http://skim-app.sourceforge.net/`.

2.4.2 Sumatra PDF – Windows

A good viewer for Windows based machines
`https://www.sumatrapdfreader.org/free-pdf-reader.html`.

A note about Adobe Acrobat on Windows machines. On windows machines Acrobat locks the file when it opens it, this means that pdflatex and tools cannot write to the file to rebuild it.

2.5 SyncTeX integration

In the settings for the latex plugin for Atom, and the various PDF viewers, you'll see settings for something called SyncTeX. This does two useful things.

Firstly it allows the editor to move the document to the position the cursor is at in the text. In Skim and Sumatra the viewer displays a coloured dot corresponding to the position of the cursor.

Second and more useful is allows the PDF viewer to control the cursor position int he editor. Click on a location in the PDF, and the cursor in the editor is moved to the corresponding file and location in the sources.

2.6 GitHub

Treat the documents as code. Put the ToR and Dissertation under version control. It also makes moving your work between university and home easy. Just commit all the changes, push the repository back to the server, then pull the repository at the other end. It also means you have a copy backed up.

Sign up for the Student Developer Pack See `https://education.github.com/` and `https://education.github.com/pack/`. They will give you unlimited private repositories (normally $7/month) while you are a student.

2.7 Perl

Some of the *very* useful tools in texlive, need a Perl installation to work. In Linux and Macs, perl should be there automatically. For windows although texlive does install a minimal Perl set, it isn't enough for the really usefull tools (see 2.8.1 and 2.8.2).

On Windows machines install Perl before TeXlive. Make sure that the `Perl.exe` is in the system PATH before installing TeXlive. My Windows installation of Perl is Strawberry Perl `http://strawberryperl.com/`.

2.8 TeXlive utilities

There are a couple of very utilities that come with TeXlive, they do need a complete Perl installation (see 2.7).

2.8.1 latexmk

Latex needs several passes through to collate and use cross references. It also needs a pass by BibTeX to compile the reference list, and latex passes to put the citations in.

`latexmk` is a make like program that understands latex, it can parse the log files generated and re-run the appropriate tools until a build is complete.

```
latexmk -pdf TermsOfReference
```

2.8.2 texcount

`texcount` parses the docment, following any inputed or included files, and counts the words used. Being TeX aware it knows how to ignore the markup.

```
texcount -total -inc Dissertation.tex
```

2.8.3 texdoc

Latex is *very* well documented. To find the documentation for a package, which are loaded in the preamble. Use `texdoc` with the package name.

```
texdoc minted
texdoc tcolorbox
texdoc tikz
```

2.9 Listings

There are several packages that layout code listings, complete with syntax highlighting. Their advantage is that they can read in and highlight a source-file in the appropriate language.

2.9.1 minted

I use the `minted` package, it does require the use of an external program. You will need to have Pygments `http://pygments.org/` installed, which depends on having a Python installation `https://www.python.org/`.

```
\inputminted{c}{hello.c}
```

```c
#include <stdio.h>


int main ( int argc, char *argv[] )
{
    printf("hello");
}
```

In running latex you'll need to enable shell-escape.

```
pdflatex -shell-escape Dissertation
latexmk -latexoptions=-shell-escape Dissertation
```

Or see the settings for the latex Atom package

2.9.2 listings

Listings is an older package. It doesn't produce coloured output, but it is written in tex, so it has no external dependencies.

```
\lstset{language=c}
\lstinputlisting{hello.c}
```

```c
#include <stdio.h>


int main ( int argc, char *argv[] )
{
    printf("hello");
}
```

Part II

Synthesis

Chapter 3

Some TeXnical Details

3.1 The Gantt Chart

The Gantt chart in the Terms-of-Reference is drawn with another latex package. It uses an `input` command to pull it into the tor (and dissertation).

Most of the file `Gantt.tex` is the setup of the Gant chart, in order to make it fit in one page.

```
% % --Tasks go here
% % put in a title, a start date, end date...
% \ganttbar{TOR}{11/9/17}{13/10/17}
% \ganttbar[inline]{\emph{revise}}{15/10/17}{10/11/17}\\
% \ganttbar{Analysis}{18/9/17}{24/11/17}\\
% \ganttbar{Design}{31/10/17}{17/1/18}
% \ganttmilestone[ms=right]{Build complete (Obj \ref{write-code})}{18/1/18}\\
% \ganttbar{Exploration}{22/9/17}{15/11/17}
%
```

Part III

Evaluation

Bibliography

Henning Schulzrinne. Writing systems and networking articles. URL `http://www.cs.columbia.edu/~hgs/etc/writing-style.html`.

Brian Kerngihan and Denis Ritche. *The C Programming Language*. Prentice Hall, second edition, 1988.

Nicola L. C. Talbot. *Using LaTeX to Write a PhD Thesis*, volume 2 of *Dickimaw LaTeX Series*. Dickimaw Books, Norfolk, UK, 2013. ISBN 978-1-909440-02-9. URL `http://www.dickimaw-books.com/latex/thesis/`.

Part IV

Appendices

Appendix A

Terms of Reference

A.1 Background

This document demonstrates the structure of a proposed Terms of Reference document written in LaTeX.

This should describe the "context" of the proposed project and answer the question, "Why this project?", both from your own perspective as a student undertaking a final year computing project, and that of any client. It should show what makes this proposal a worthwhile computing final year project. *It must make clear both the application area or area of investigation and the computing aspects of your work.*

LaTeX and its companion BibTeX are a good pair of tools for writing your documentation.

Don't forget to reference background material. BibTeX makes this simple. You can have one or more `.bib` files, these are easy to populate from academic search tools, such as Google-scholar, and the Library's search facilities. A bibliography file looks something like figure A.1. In the text use the label in a citation command `\citep{k+r}` the toolset puts the right form of citation [Kerngihan and Ritche, 1988, pages 2–4] into the text.

Where you want the bibliography to go use the `\bibliography{c,unix}` command, with a comma separated list of `.bib` files (without the extension).

```
@book{k+r,
        Author = {Brian Kerngihan and Denis Ritche},
        Edition = {Second},
        Publisher = {Prentice Hall},
        Title = {The C Programming Language},
        Year = 1988
}
```

Figure A.1: Sample BibTeX content

A.2 Proposed Work

. The project must exhibit a level of difficulty appropriate to final year honours BSc work, and be of a size that can be attempted in the time available; this section should define the topic and project work in enough detail for the markers to be sure that it is suitable. The more detail and discussion you produce at this stage, the stronger the foundation for the actual project work.

You should emphasise the computing aspects you expect to be involved in, including those specifically relevant to your programme. Remember that you are undertaking the project as part of a BSc programme in a computing-related discipline, and avoid being side-tracked into areas that are not relevant to your course.

A.3 Aims and Objectives

There should only be one or two aims

A.3.1 Aims

To show how LaTeX and tools can be used to write a dissertation

A.3.2 Objectives

Your objective list is a series of measurable objectives, can you tick each one off as *done*? I usually expect between 8 and 12 objectives

The **enumerate** environment is useful here for generating a numbered list. You can put `\label{}` commands in with a keyword `\label{understand-problem}` and then refer to the label with a `\ref{understand-problem}` command, it puts the number of the objective in the text

```
See objective \ref{understand-problem}
```

See objective 1

1. **Classify the problem domain** this is where you develop an understanding of the nature of the problem/project
2. **Identify Techniques to solve** What Algorithms are you to use, how is a database structured,
3. **Select tools to use** What languages, software, hardware; are you using?
4. **Design the system to be build** Its requirements, the **test plan**, the architecture (Layer model/Model-View-Controller)
5. **Build the system** I'd include testing here, as the result is a *working wywtem*

A.4 Skills

This is where you can cover the skills you have relevant to the project and the new skills you are going to acquire during the project.

1. Programming in C, see module KFxxx
2. Hardware Design

A.5 Resources

This is an important section, it lists the hardware and software you are going to need for the project.

A.5.1 Hardware

For Hardware this is more critical, as we need to identify any hardware we have, or that you are going to buy. We do have an ordering mechanism in the Department, but time and budget are critical constraints here.

A.5.2 Software

In the case of software, there isn't usually an issue, unless you're needing huge amounts of run-time (we don't have a super-computer handy).

A.6 Structure and Contents of the Report

Here you set out the likely chapters you will have in your report. Usually each objective lends itself to one or more chapters. You can refer back to the objectives set.

A.6.1 Report Structure

Introduction Sets out the background and motivation for the project. Summarises the work done, the results, the conclusions, and the recommendations for future work. It is a one chapter summary of the *entire* project.

Defining the problem Objective 1 requires a precise definition of the problem you are solving. Don't forget to reference good source material [Henning Schulzrinne] and [Talbot, 2013]. See section A.2.

Possible Solutions Discuss the possible solutions, compare the alternatives, and select the one to use for the implementation.

A.6.2 List of Appendices

What Appendices you will include. A copy of the TOR should be the first, followed by the Ethics form and the Risk Assessment.

Others might include design documentation, code listings, tables of results (if too large to include in the main text).

A.7 Marking Scheme

The marking scheme sets out what criteria we are going to use for the project.

Project Type General Computing or Software Engineering projects

Project Report State which chapters constitute the *Analysis*, the *Synthesis*, and the *Evaluation*. This help me when marking to know when to stop reading one section and put a mark down for it.

Product List the deliverables that make up the *Product*. Code, design, requirements specifications, test plans, etc.

For the *Fitness for Purpose* and *Build Quality* list the critera used to asses the product by

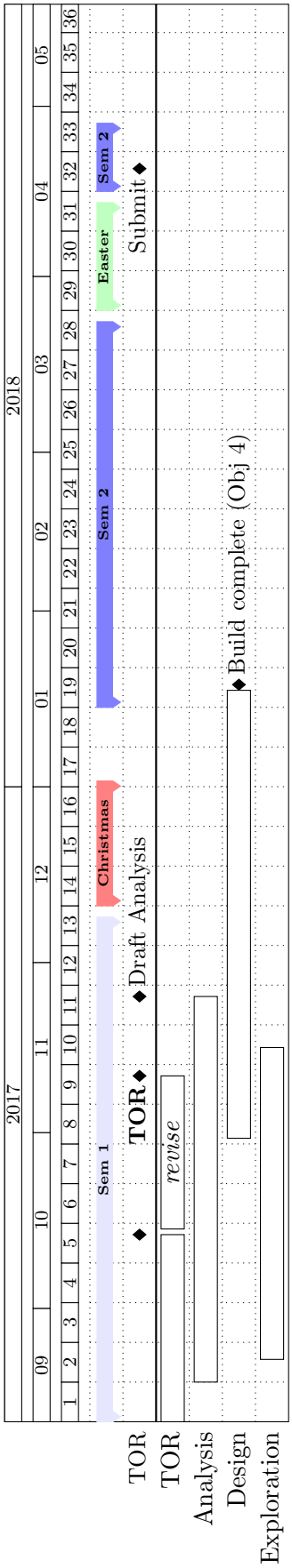
Fitness for Purpose

- meet requirements identified
- other appropriate measures

Build Quality

- Requirements specification and analysis
- Design Specification
- Code quality
- Test plan and Results

A.8 Project Plan



A.9 Ethics Form

If you scan the Ethics form on one of the multifunction printers, you can get a pdf copy. This can then be included with the LaTeX command

```
\includegraphics{ethics.pdf}
```

Assuming of course you have saved the form as `ethics.pdf`

A.10 Risk Assessment Form

Likewise you can scan and include the Risk Assessment Form

```
\includegraphics{risk-assesment.pdf}
```