Cardiff University

Department of Computer Science

A Short Guide to Writing

Your Final Year Project Report
Or
MSc Dissertation

Abstract

This guide is meant to help you produce a good computer science final year project report or MSc Dissertation. It gives advice on how to gather relevant material, how to organise it into a suitable form and how to then turn it into written prose. It also describes the conventions that should govern the structure of the project report and suggests some descriptive devices that you can use to make it more effective. A summary of the guidelines is given at the end, and an appendix lists the rules governing presentational details including length of the report, print quality, font sizes, etc.

Contents

1 Introduction

This guide is meant to help you produce a good final year project report or MSc dissertation. A good report is one that presents your project work *concisely* and effectively. It should contain various materials relevant to the project; it should be organised into a logical framework; and it should be supported by textual structures that follow well-established conventions consistently.

An important point to remember is that the report should describe *your* work. Large chunks of bookwork describing standard material are unnecessary. You should simply refer to such material where necessary – assume that your reader is a competent computer scientist.

The guidelines here are arranged roughly in the order that you will need them. For undergraduate projects much of the information given here is based upon the ideas presented in the first year course "Communication Skills and Project Management". Therefore we recommend that you re-read that course's notes.

2 Gathering Material

First we will identify the kinds of material you need to collect before you can begin writing in earnest.

Most of the material will consist of your own ideas and experiences gained while carrying out the project. While working on the project you should keep a notebook handy and record all relevant information. Typically such information will include:

- specifications;
- designs;
- results of tests;
- debugging activities;
- notes from meetings or interviews with
  - your supervisor;
  - potential end-users
  - technical experts;

  and so on.

Also, keep a diary of all your project-related activities. This will show the progress made during the life of the project and will provide a record of how you spent your time. In particular, when you are testing and debugging your work, keep a running log of your debugging activities and their results. You will then have a record of the unforeseen difficulties you met and, hopefully, how you resolved them. Summaries of these may well be worth including in the project report (see Section 3.4).

In general you should supplement the material you generate yourself with relevant material from other sources. A good project report will show that you are aware of relevant work that other people have done (see Section 3.2 for details). You should include relevant references to such work in your project report. References to work in periodicals (i.e. magazines and journals) and conference proceedings may be more useful than references to textbooks, as periodicals and conferences are usually more specialised and up to date. References to technical manuals can also be included.

3 Arranging Material and Structuring the Project Report

Once you have started to gather material you can begin to arrange it in a form which can then be refined into the final project report.

All good technical project reports whatever their subject, follow certain well-established conventions and have a similar overall shape. They all consist of a main body surrounded by other textual structures that support it in various ways. Some of these are mandatory, others are optional.

Figure 3.1 shows an example of the layout we describe for a project which implements a piece of software. You should vary the titles of the sections if they are inappropriate for your project -- consult your supervisor about this. For the moment we will concentrate on the main body of the report and leave the supporting structures until later. We recommend that you do the same when writing your report.

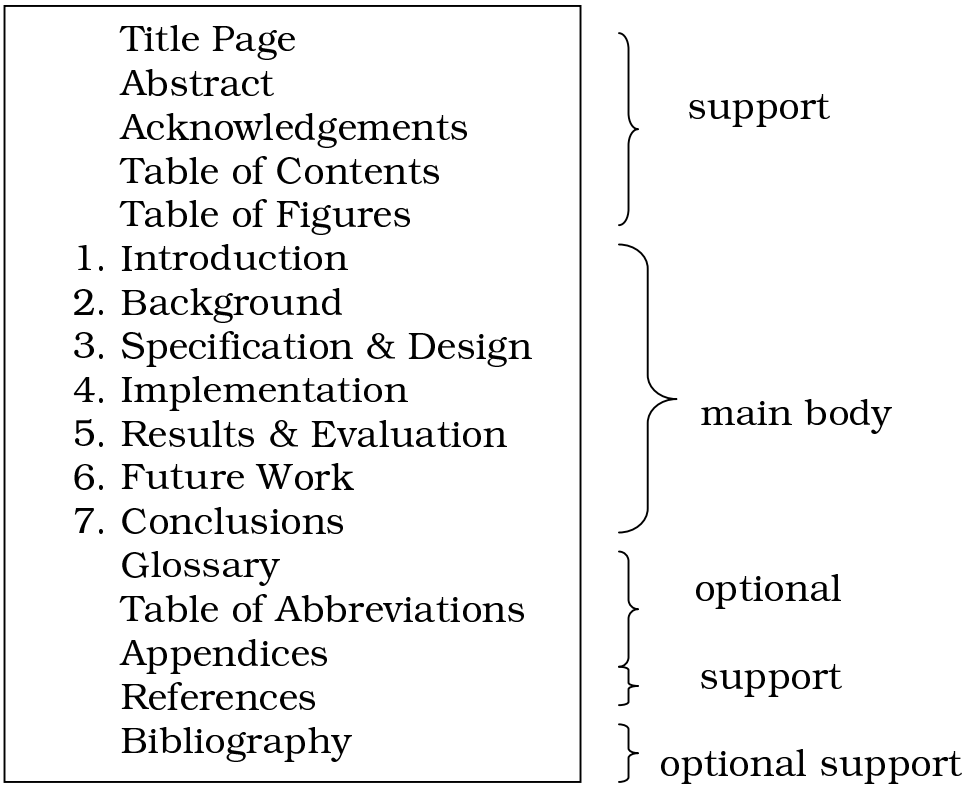

Figure 3.1: Project Report Structure

Project reports describing projects whose aim has been to develop a particular software system tend to have a main body with a characteristic structure. As illustrated above, the body is typically divided into seven major sections.

We will look at each of these in more detail shortly. You can use this characteristic structure as a kind of template for partitioning the material. It is a good idea at this stage to plan roughly how long each part should be, to make sure that the length and

overall balance are about right. You can then construct each part to produce a first draft of the main body.

3.1 The “Introduction”

A good introduction should tell the reader what the project is about without assuming special knowledge and without introducing any specific material that might obscure the overview. It should anticipate and combine main points described in more detail in the rest of the project report. Normally it should include such things as:

- the aim(s) or goal(s) of the project;
- the scope of the project;
- the approach used in carrying out the project; and
- assumptions on which the work is based.

3.2 The “Background”

The purpose of the Background section is to provide the typical reader with information that they cannot be expected to know but which they will need to know in order to fully understand and appreciate the rest of the report (see Section 4.1 for details of who a typical reader might be). This section may describe such things as:

- the wider context of the project;
- the anticipated benefits of the system;
- likely users of the system;
- any theory associated with the project;
- the software development method(s) used;
- any special diagramming conventions used;
- existing software (or hardware) that is relevant to the system;
  and so on.

The wider context of the project includes such things as its non-computing aspects. So, for example, if you are producing software for a specific organisation then you should describe aspects of that organisation’s business that are relevant to the project.
Relevant existing software systems that you should mention could be ones that, for example

- are similar to the one you are building;
- support your project;
- your project aims to extend.

You need only describe things that will be unfamiliar to the potential reader. Your project will almost certainly use all kinds of existing software such as language compilers, subroutine libraries, etc but you can assume that the reader will be fully acquainted with, for example, general purpose programming languages such as Java, Pascal, C, Fortran, Basic, etc,. Also, the better known specialised packages such as ORACLE, INGRES, Lotus 123, etc. You should mention the particular variety and possibly version number, e.g. Java 2 SDK v1.2.2, but you need say nothing more than that.

If your project depends on any esoteric software such as specialised subroutine packages or any of the more obscure 4th generation languages, you should describe it briefly and discuss whatever features are relevant to your project. Often this can be done by comparing it to some well-established piece of software, for example

```
The Descartes language is like a restricted
version of Pascal but with the following extra
features…
```

Again, long descriptions of details are to be avoided and references to other material should be given instead.

Other background information could consist of the sequence of events leading up to the present project or the results of earlier investigations. You could also discuss such things as any cost or time constraints imposed on the project.

3.3 The "Specification & Design"

The purpose of the Specification and Design section is to give the reader a clear picture of the system you have created and why you created it in the way that you did. A specification should tell the reader what the software system is *required* to do. The design then gives the top-level details of how the software system meets the requirement.

Describing what a software system does (specification) and how it does (design) effectively usually means describing it from more than one viewpoint. Each viewpoint will convey some information about the systems that other viewpoints omit. (You would use the same technique when describing any complicated construction such as a building, an aircraft, a novel or a painting.) Possible viewpoints might be:

- the user interface;
- the dynamic behaviour of the system;
- how data flows through the system;
- what data types are implemented in the system;
- what algorithms are implemented in the system;
- the static architecture of the system, i.e. how the code is partitioned into modules, etc.

A common approach is to first describe the static architecture, identify modules and groups of closely connected modules, and then to apply other views to each of these groups. Fine details, specifically details of code, should be left out.

We strongly recommend that you make extensive use of diagrams, such as entity-relationship diagrams or state charts, or other pictorial techniques (see Section 5.4 for more detailed advice on this).

As well as describing the system, it is important that you *justify* its design, for example, by discussing the implications of different design choices and then giving

reasons for making the choices you did. Typically these implications will relate to the aims of the project and to aspects of it discussed in the Background section.

The design of the system will almost certainly have evolved while you were developing it. Obviously you should describe its final state but often there are good reasons for describing intermediate states too; for example if you want to discuss the details of the design method used. If you do this, take special care to make sure the reader does not get confused between different stages of the design.

3.4 The "Implementation"

The Implementation section is similar to the Specification and Design section in that it describes the system, but it does so at a finer level of detail, down to the code level. It can also describe any problems that may have arisen during implementation.

Do *not* attempt to describe all the code in the system, and do *not* include large pieces of code in this section. Complete source code listings should be put in an appendix (see Section 6.6). Instead pick out and describe just the pieces of code which, for example:

- are especially critical to the operation of the system;
- you feel might be of particular interest to the reader for some reason;
- illustrate a non-standard or innovative way of implementing an algorithm, data structure, etc..

You should also mention any unforeseen problems you encountered when implementing the system and how and to what extent you overcame them. Common problems are:

- difficulties involving existing software, because of e.g.
  - its complexity,
  - lack of documentation;
- lack of suitable supporting software;
- over ambitious project aims.

A seemingly disproportionate amount of project time can be taken up in dealing with such problems. The Implementation section gives you the opportunity to show where that time has gone.

3.5 The "Results and Evaluation"

In this section you should describe to what extent you achieved your goals.

You should describe how you demonstrated that the system works as intended (or not, as the case may be). Include comprehensible summaries of the results of all critical tests that were made. You might not have had the time to carry out any full rigorous tests – you may not even got as far as producing a testable system. However, you

should try to indicate how confident you are about whatever code you have produced, and also suggest what tests would be required to gain further confidence.

You must also critically evaluate your system in the light of these tests, describing its strengths and weaknesses. Ideas for improving it can be carried over into the Future Work section. Remember: no project is perfect!

This section also gives you an opportunity to present a critical appraisal of the project as a whole. This could include for example whether the methodology you have chosen and the programming language used were appropriate.

3.6 The “Future Work”

It is quite likely that by the end of your project you will not have achieved all that you planned at the start; and in any case, your ideas will have grown during the course of the project beyond what you could hope to do within the available time. The Future Work section is for expressing your unrealised ideas. It is a way of recording that ‘I have thought about this’, and it is also a way of stating what you would like to have done if only you had not run out of time[1]. A good Future Work section should provide a starting point for someone else to continue the work which you have begun.

3.7 The “Conclusions”

The Conclusions section should be a summary of the aims of project and a restatement of its main results, i.e. what has been learnt and what it has achieved. An effective set of conclusions should not introduce new material. Instead it should briefly draw out, summarise, combine and reiterate the main points that have been made in the body of the project report and present opinions based on them.

The Conclusions section marks the end of the project report proper. Be honest and objective in your conclusions.

3.8 The “References" and "Bibliography”

In Section 2 we said that you should relate your work to that of other people. Other work explicitly cited should be listed in the Reference section and referred to in the text using some kind of key. It is important that you give proper credit to all work that is not strictly your own and that you do not violate copyright restrictions.

References in the Reference section should be listed in order of author’s surname(s) alphabetically, and should give sufficient and accurate publication details. For example,

[Key1] E.J. Chikofsky and J.H. Cross. Reverse Engineering and Design Recovery: A Taxonomy. IEEE Software, 7(1):13-17, 1990.

[1] Remember to take into account Hofstadter’s Law:
‘Everything takes longer than you think, even when you take into account Hofstadter’s Law.’

[Key2] C.J. Date, An Introduction to Database Systems, 7th edition, Addison-Wesley, 2000,ISBN 0-201-38590-2.

are acceptable bibliographic entries.

There are various conventions for creating bibliographic keys. For example, you can quote the name of the author and the year of publication, e.g.

```
A more detailed description can be found in
[Borland 89].
```

There are several other variations. For example, some authors prefer to use only the first three or four letters of the name, e.g. [Borl89] or just to number the references sequentially, e.g. [3].

It can be helpful to the reader if, for books and other long publications, you specify the page number too, e.g. [Borland 89 p22], or just number the references.

Whatever convention you choose BE CONSISTENT.

Information Services provide two leaflets - "Citing references" and "Citing electronic references" which describe in detail accepted ways of presenting references. These may be viewed at http://www.cf.ac.uk/uwcc/infos/resource/usrguide/

It may be desirable to provide a separate list of references and a bibliography. In this case references are those documents/sources cited within the text. The bibliography lists documents which have informed the text or are otherwise relevant but have not been explicitly cited.

4 Writing the Project Report

Once you have gathered and organised enough material you can turn it into written prose. To write effectively requires sustained concentration over long periods of time. Even with the incremental authoring possibilities that word processing offers, writing is best done in long uninterrupted sessions. Most people find it difficult and exhausting.

There are rules you can follow which may make the task easier and which will certainly improve the quality of your writing, but unfortunately there are rather a lot of these and in a guide of this size we can only offer a few pieces of general advice:

- keep your potential readership in mind;
- identify commonality;
- use sections and subsections;
- follow stylistic conventions.

The project report's structure does not necessarily dictate the order in which you write it. If you want you can start by writing the Introduction, then the Background section, and so on, but this is up to you. Some people start by writing the Introduction first which gives direction to writing the other sections, but others prefer to leave writing the Introduction until last. However, we recommend that you start with the middle sections, then write the Introduction (guiding the reader to what they will find in the report), then the Conclusions (bringing the report together at the end), and finally the Abstract (summing up the entire report).

4.1 Potential Readership

Always keep your potential readers in mind and repeatedly review what you have written, putting yourself in their place. Look at the draft, sentence by sentence, and ask yourself: 'Will this make sense to the readers given their existing knowledge and what I have told them up to now?' You can consider the potential readership as:

- your academic supervisor;
- your project moderator/internal examiner;
- the external examiner (usually a computing professor from another university),

and quite possibly

- future computing students.

So, as said earlier, do not explain things which are common knowledge to such readers.

Also, if your project report is of sufficient quality, your supervisor may consider submitting part of it to a journal for publication as a paper, in which case it may eventually be read by a substantial number of computing professionals.

4.2 Identifying Commonality

You can often both clarify text and reduce its bulk if you can identify generality or commonality among the ideas you are expressing. You can then revise the text so that the common factors are described first followed by details of how specific individual ideas differ from them.

4.3 Sections and Subsections

The main body of the project report should be divided up into sections, along the lines suggested in Section 3 or otherwise, as appropriate. Each section should, if necessary, be divided up into subsections, and so on recursively. Such nesting can be used to suggest some kind of hierarchical relationship between sections. This can become obscure though if the nesting gets to more than about three levels deep.

Each major section should begin on a new page. All sections and subsections should be numbered and headed. Numbering should be like this: 3.10.7 – for subsubsection 7 in subsection 10, in section 3.

It is important that you start each section and subsection with a summary of the rest of the material in it: i.e. inform the reader of what you are about to tell them. This has the effect of "softening up" the reader so that when they move on to the body of the section they feel confident about the direction in which you are taking them. They are reassured at regular intervals when they encounter ideas that you have told them to expect. Without the overview the overall effect is like a mystery tour of ideas, with each new idea coming as a surprise. It is sometimes difficult to appreciate the need for this when you are the author because you are already intimately familiar with the whole route that the report takes.

4.4 Stylistic Conventions

There are all kinds of stylistic conventions relating to technical writing that you should try to follow. For example:

- avoid shortened forms such as "don't" for "do not";
- avoid colloquialisms and slang words;
- use British English and write in complete sentences;
- divide your writing up into paragraphs.

To some extent you can use your own judgement about what conventions to follow.

Whatever you do though, you must be *consistent*. Writing where the language style or typography (e.g. font or character size) change arbitrarily looks amateurish and can be very distracting for the reader. Other places where consistency should be maintained include:

- bullets points;
- use of hyphens;

- use of capitalisation;
- technical terms;
- abbreviations;
- use of symbols.

5 Using Descriptive Devices

In this section we will mention some well-established descriptive devices which you can use in your project report to improve its quality.

5.1 Cross-references

Cross-references are just references to other parts of the same document. For example,

```
This module contains procedures for operating
on variables of type WINDOW (see Section 2.2).
```

Section numbers will change if sections are added or deleted, so it is a good idea to wait until the report is almost complete before putting in any cross-references, if your word-processing software does not automatically number sections for you.

Backward references to sections earlier in the project report can make explicit connections between parts of the document that may not be connected obviously. Forward references can be used, for example, to reassure the reader that you are not going to leave them stranded after you have introduced a new idea without explaining it. For example,

```
This procedure uses the Volestrangler algorithm
(to be described in Section 4.3).
```

Note that too many forward references are an indication that the report could be organised better.

5.2 Footnotes

Many word processors have facilities for handling footnotes. By all means use them, in particular when you want to make a comment which is not strictly relevant or which would upset the flow of ideas in the text.

5.3 Lists

Traditionally, collections of items are listed within the text using the adverbs ‘firstly’, ‘secondly’, etc. Often though it is clearer to tabulate these items, particularly if there are many of them. The simplest way of doing this is to use a “bullet” list[2]. Various examples of bullet lists appear throughout this guide. Sometimes there is a need to nest one list inside another. To distinguish the two lists, the inner one can be indented and have a different symbol. Lists with more than one degree of nesting tend to appear confusing and therefore we do not generally recommend them.

[2] “Bullet” is the name for the ● symbol, although other similar symbols are also available.

Listed items can also be keyed using numbers, letters, or other labels. Bibliography entries are an example of keyed items (see Section 3.8). However, keys should only be used when necessary.

5.4 Figures

A project report that uses figures (i.e. diagrams or other pictorial techniques such as tables) to illustrate ideas will probably be easier to digest than one that does not. We therefore recommend that you use figures wherever appropriate[3].

Be careful though. When drawing diagrams try to keep to a standard graphical notation that has been introduced on a computer science course, or that you have seen published widely, and use it consistently. Computer Science, unlike most other professions, has few established conventions governing the use of diagrams and this means that diagrams can sometimes make ideas more obscure rather than clarifying them.

If you feel you have to invent your own notation, remember that the best ones are usually the most *economical*, i.e. they use only a few different kinds of symbols. Also, you MUST explain the precise meaning of your symbols in a key. A very common mistake is to use arrows to illustrate some kind of relationship between items without declaring what that relationship is.

Graphics editors (i.e. picture processors) can be extremely useful, particularly if you have a great deal of drawing to do or if there is a lot if commonality among the drawings (because cut and paste operations can then be used with great effect).

All figures should be labelled and captioned, for example,

```
Figure 3.10 Sub-System Architecture
```

for the tenth figure in Section 3. The label can then be used to refer to the diagram within the text (all diagrams must be explicitly referred to somewhere within the text).

5.5 Literal Text

It is important when writing about software systems to distinguish in the text between the ordinary natural language you are using and the program code or other literal text. If you are using a word processor which offers both proportionally spaced and fixed width character fonts then there is a straightforward way of doing this. Program code and other literal text can be written in a fixed width font such as Courier while the natural language text can be written with a proportionally spaced font such as Time. For example,

> The procedure `draw_circle( p:POINT, r:REAL)` draws a circle of radius `r` at point `p` on the screen.

[3] If you have a graphical rather than a textual/verbal kind of mentality, a good way to write text is to express your ideas in diagrams first and then describe these textually.

Other similar kinds of text, UNIX commands for example, can be treated in the same way. Using fixed width font means that the code appears in the document much as it would do on most computer screens. If you only have fixed width characters available on your word processor then put program code etc. into *italics* or **bold** text. Note that using more than a few different character fonts, styles or sizes can make text look very untidy.

6 Supporting Structures

In Section 3 we said that a project report consisted of a main body plus other textual structures that surround and support the body. There are well established conventions governing the purpose and format of these supporting structures which we will describe now. The structures are, in order of appearance in the project report:

- the title page;
- the abstract;
- the acknowledgements;
- a table of contents;
- a table of figures.

Then comes the main body of the project report, and this is followed possibly by:

- a glossary;
- a list of abbreviations;
- one or more appendices;

and finally

- the references and bibliography.

Each of the elements listed above should begin on a new page. All pages should be numbered, with page 1 being the first page of the Introduction. The pages preceding the Introduction should be given Roman Numerals (i, ii, iii, iv, etc).

6.1 The Title Page

The title page should be the first page of the report (or displayed on the front if the report is bound) and should normally include:

- the title of the project report;
- the name of the author;
- the name of the project supervisor;
- the qualification for which the project report is a part;
- the name of the department and institution (e.g. Department of Computer Science, Cardiff University);
- the date of completion of the project report.

The title itself should be short, yet should aim to describe the contents of the project report as accurately as possible. MSc students are advised to consult current university submission regulations to determine the appropriate form of the title page and page of declarations required in a MSc dissertation.

6.2 The Abstract

This is a summary of the report. It must be less than 300 words long. It should give enough information to allow a potential reader to decide whether or not the report will be of interest to them. It should briefly describe the main ideas of the report, including the aims and conclusions. It should be both self-contained and self-explanatory, and it should not say anything not mentioned in the rest of the report (for this reason it is usually written last).

6.3 Acknowledgements

This optional section should be used to record indebtedness for the use of facilities or help from particular sources. You should mention any organisations who have helped you while you have been carrying out the project.

6.4 The Table of Contents and Table of Figures

The table of contents gives the reader a view of the detailed structure of the report, by giving section and subsection headings and associated pages.

If your project report contains many figures or it refers to the same figure many times you should consider listing them along with their page numbers in a table of figures.

6.5 The Glossary and Table of Abbreviations

If you use any abbreviations, obscure terms or esoteric acronyms in the project report then their meaning should be explained where they first occur. If you go on to use any of them extensively then it is helpful to list them all in a table at the end so that readers can quickly remind themselves of their meaning.

6.6 The Appendices

Appendices are repositories for material which you want to include in the report, but which would seriously obstruct the flow of ideas if put anywhere in the main body. In computing project reports it is normal to include a printout of (the final version of) the source code of the system in an appendix. (If you have carried out the project for a commercial organisation make sure that you have their permission to do this.) Other documents you have written, such as user manuals, technical manuals or formal specifications should go here too.

Appendices should be headed by letters in alphabetical order, i.e. `Appendix A`, `Appendix B`, etc.

7 Sources of Further Guidance

We have said several times that this guide is not complete. Textbooks usually demonstrate good technical writing especially if they are produced by a reputable publisher. They also provide illustrations of the use of descriptive devices, etc.

Appendix B contains a list of other books and Internet resources, specifically designed to assist in writing essays and reports. While writing this guide, we have tried to follow the guidelines it contains. Obviously though, a guide like this has a different purpose to that of a project report so total adherence has been impossible. We have tried to set an example, but it is not perfect.

Finally, there will be specific aspects of your project report that only your supervisor can advise you on. It is important that you discuss an outline of the project report with him or her before you begin to write up.

8 Conclusions

These are our main recommendations:

- Record all relevant information generated by the project;
  - use a notebook,
  - keep a diary,
  - log debugging sessions.
- Gather further material from publications or other external resources.
- Organise the material into sections agreed with your supervisor
  - e.g.“Background”, and so on.
- Turn this material into written prose to form the project report’s main body.
- When writing the main body;
  - keep your readership in mind,
  - identify commonality,
  - use sections and subsections,
  - follow stylistic conventions.
- Where appropriate use;
  - Cross-references,
  - references,
  - figures and other descriptive devices.
- Produce all required supporting structures according to convention, after completing the main body.
- For examples to follow, look at;
  - textbooks from reputable publishers,
  - the way this guide is written.
- Discuss an outline of the project report with your supervisor before you begin to write up.

We hope this guide is of some use. Good Luck ☺

Appendix A Rules for Report Presentation

(MSc. students should check current university submission regulations, which take precedence over the rules given below)

Length: Reports longer than 20,000 words, not including the supporting structures, will not be accepted. First class/Distinction reports are **typically much shorter** than this. There is no minimum length but it is mainly through the report that your project will be judged so the report should adequately reflect the work done in the project.

Print Quality: Reports should be typed or word-processed. Letter quality matrix print is acceptable.

Font Size: Reports should be printed using 11 or 12 point typefaces.

Line Spacing: Lines should be 1.5 spaced. (This guide is 1 spaced.) The report should be economical on paper. It should not, for example, contain excessive amounts of white space. Only the major sections (as illustrated in Figure 3.1) need to begin on a new page.

Appendix B Books and Internet Resources for Writers

Creme P., Lea M.R., *Writing at University: A Guide For Students*, Open University Press (1997), 0-335-19642-X
Contains help for all aspects of writing while at university.

Duprė L., *Bugs In Writing*, Addison-Wesley, (1995), 0-201-60019-6
Contains tips in small, individual chapters to improve your writing. A good reference book.

Fry R., *Improve Your Writing*, Kogan Page, (1997), 0-7494-2347-1
A guide for students producing a written project. Easy to read, full of handy hints, and guides you through the whole process from carrying out research for your report through to producing the final draft.

Internet Resources

Quick Tips for Better Writing, Criscadian's Active Voice, Ayres J.A.,
< http://members.aol.com/criscadian/tips.html> (1997)

10 Ways To Improve Your Technical Writing, Center for Technical Communication, Bly R.W. < http://smartbiz.com/sbs/arts/bly10.htm>

Capital Community – Technical College, *Guide to Grammar & Writing*,
< http://webster.commnet.edu/HP/pages/darling/original.htm > (1999)

Capital Community – Technical College, *Guide to Grammar & Writing*,
< http://webster.commnet.edu/HP/pages/darling/grammar/dictionaries.htm>

English Skills Support, University of Central England, Gould S.
< http://lmu.uce.ac.uk/lmu/esu/> (1999)

Internet Resources on Writing, University of St. Thomas, Landsberger J.,
< http://www.iss.stthomas.edu/studyguides/bib_writing%20webs.htm>

Michigan State University, *Writing and Presenting Your Thesis or Dissertation*,
<http://www.canr.msu.edu/aee/dissthes/guide.htm> (1999)

University of Victoria Writers' Guide, The University of Victoria,
< http://www.clearcf.uvic.ca/writersguide/welcome.html> (1997)

Appendix C Alternative Project Report Layout

In this appendix we briefly outline alternative project report layouts to the one given on page 3. If the nature of the project is not to design and implement some software, but for example to compare two algorithms, a more suitable layout could be the one shown in Figure C.1.

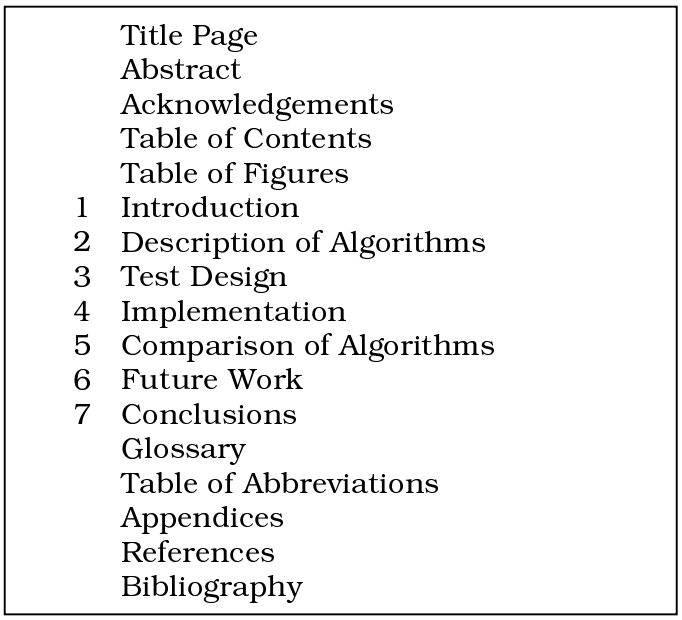

Figure C.1 Alternative Project Report Layout

Another project might involve the design and analysis of an algorithm. Here, there might be very little said about user interface, but a lot about analysis compared to the type of project on page 3. A possible report layout for the project could be:

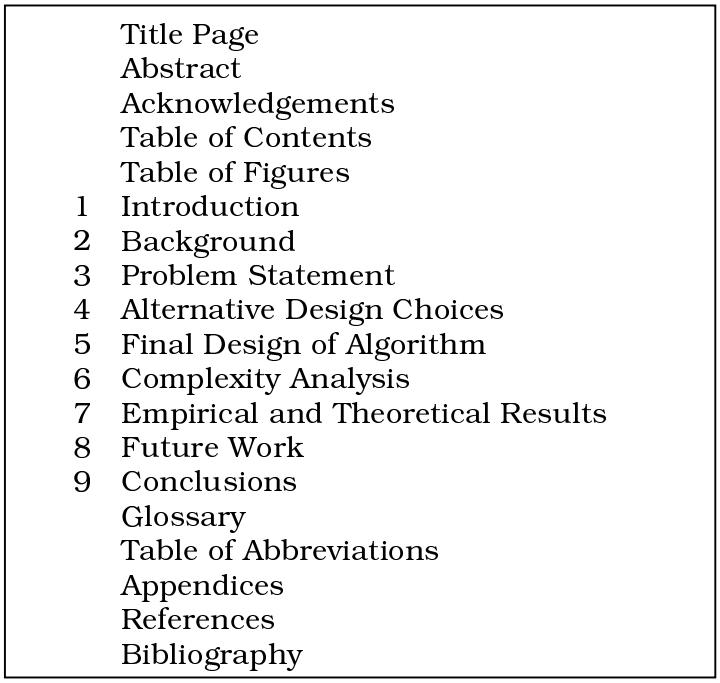

Figure C.2 Another Project Report Layout

Other possible types of project also exist. It is important that you consult your supervisor and suitably modify the suggested structure on page 3 to suit your own project.