BSc Applied Computing, BSc Games, Final Year Project Modules (Project Preparation, Development Project) — General Guidelines

BSc Projects Team,
Letterkenny Institute of Technology,
Letterkenny, Co. Donegal, Ireland.

URL http://www.jgcampbell.com/bscp/
jonathan.campbell@lyit.ie

Report No: 2009/jc/14/r
Revision 3.3

This report gives guidelines on the conduct and documentation of BSc Final Year Project Modules at Letterkenny Institute of Technology.

9 September 2009

Contents

Chapter 1

Introduction

1.1 Purpose

This report presents guidelines regarding the conduct BSc Applied Computing, BSc Games, Final Year Project Modules (Project Preparation, Development Project). Although the modules Project Preparation and Development Project are separate, one lead on to the other, consequently this document considers them together. That is, it considers the whole project life-cycle, starting with the choice of topic and drawing up of a project proposal, through to the preparation of the final report and oral presentation.

A good book on student computing projects is Dawson (1999).

Chapter 2

Deliverables

2.1 Deliverables for Project Preparation Module (Semester 1)

Project Proposal An agreed project proposal and brief specification (see Appendix A) must be submitted to the projects coordinator and the supervisor (one copy each) by 11.59am, Tues. 6th October 2009. This will be assessed as satisfactory/not-satisfactory; note that it will be contained in the Detailed Aims and Objectives, Feasibility Study and Plan.

Detailed Aims and Objectives, Feasibility Study and Plan Document A formal report must be submitted to the projects coordinator by 11.59am, Tues. 10th November 2009. The assessment mark for this report will contribute 25% of the module mark. This document should take the form of a *draft Software Requirements Specification (SRS)*, see next item.

Software Requirements Specification (SRS) Note that the SRS contains content from earlier deliverables, namely Detailed Aims and Objectives, Feasibility Study and Plan.

A final report for the module must be submitted to the projects coordinator by 11.59am, Tues. 8th December 2009. The assessment mark for this report and the Presentation will contribute 75% of the module mark.

For a suggested contents list, see Appendix B.

Presentations Mon. 14th and Tues. 15th December 2009.

In connection with the Detailed Aims and Objectives, Feasibility Study and Plan Document, and Software Requirements Specification (SRS), supervisors have asked for the following to be pointed out:

- Many projects involving software technologies with which you are not already *very* familiar will require some prototype software to be developed at this stage; you would normally conduct such prototyping as part of *risk analysis*;
- Often, students hand in a document which has been hastily researched and produced at the last minute and which shows no knowledge of the subject or progress towards the goals of the project — in other word they create a pile of waffle at the last minute that tells nothing worthwhile about the project that is to be debveloped *this year we will be careful to see that such reports are awarded very low marks*.

Marking Scheme Note the detailed marking scheme in section D.1.

2.2 Deliverables for Development Project Module (Semester 2)

Draft Dissertation A Draft Dissertation must be submitted to the projects coordinator and the supervisor (one copy each) by 11.59am, Tues. 23rd March 2010. Worth 15% of module.

Draft Dissertation Presentations Presentations of Draft Dissertation 24th-25th March 2010.

Final Dissertation A Draft Dissertation must be submitted to the projects coordinator and the supervisor (one copy each) by 11.59am, Tues. 27th April 2010. Worth 85% of module (in conjunction with the Final Presentation).

Final Presentations and Oral Examinations Thurs. 6th and Fri. 7th May 2010.

2.3 Project Tasks and Phases

Note: *this section is merely a general discussion*.

Any project can be broken into the following major phases/tasks:

1. Choose a project topic.
2. Proposal: identify problem, do preliminary analysis, and make solution proposal;
3. Plan: Do more problem analysis, break the solution into tasks; estimate effort (working hours) and duration (elapsed days / weeks) for each task; iterate as necessary to fix into available time and effort – if the project will not fit, you must warn your supervisor, and take steps to reduce the scope; arrange the tasks according to time (e.g. bar-chart);
4. Manage: ensure that the plan is being followed; revise the plan as necessary; in parallel with the next two phases.
5. Technical work: Do the technical work – with interim technical and management reports.
6. Final report.
7. Oral presentation.

Marking Scheme Note the detailed marking scheme in section D.2.

2.4 Choice of Project Topic

A software program must be designed, implemented, and tested. You must study and evaluate competing/state-of-the-art approaches; you must be able to place your chosen approach in perspective.

- Be careful to ensure that your report/project meets the requirements laid down in the course regulations;
- Seek and heed the advice of your supervisor;
- **a project that does not have a supervisor's approval is not a project**;
- Thinking ahead, you might consider employment prospects — your project is always likely to be a prime topic in job interviews;
- It is helpful to be interested in the topic — the long hours that you will spend may actually be enjoyable;
- Plan the project carefully (difficult I realise), and beware taking on too much;
- Beware relying on software or hardware resources that are not available and installed in the college **now**.

2.5 Outline of Document

This report contains two remaining chapters. Chapter 2 gives guidelines on the conduct and management of the project. Chapter 3 discusses the final report. References are given.

Chapter 3

Project Management

3.1 Introduction

To some, planning seems to threaten to weave such a web of formality around projects that no technical work will get done. Far from it. The aim is to ensure that the supervisor and student are on the same wavelength from the start, and stay that way, with a minimum of misunderstanding. Thus, even for a small project, the major benefits will be:

- Force an early allocation of appropriate time to each task;
- Identification of problems early;
- Avoidance the infamous software project 90% syndrome – the first 90% of the project takes 90% of the time (and cost), the remaining 10% takes another 90%, i.e. nothing really gets done until you have to deliver something.

3.2 Project Specification and Proposal

For student projects this consists of the one page *project specification*, see Appendix A; in a commercial project, this would consist of a full plan and costing, see chapter 3.3, below.

3.3 Planning

3.3.1 Work Breakdown

A typical project might break down into the following Work Packages (WPs):

WP0 Project Specification.

Student and supervisor(s) agree project proposal. What is to be done. Identify preliminary technical approach. Restrict to do-able proportions.

Output/Deliverable: Project Specification. Due: xx Nanober 2050.

WP1 Project Planning.

Student and supervisor(s) further sharpen project proposal. Identify major risks (e.g. awaiting software or hardware, technical novelty). Do effort and schedule estimation.

Output/Deliverable: Project Planning Document (time plan). The formality of this plan will be according to the wishes of individual supervisors.

Optional: Revised Project Specification. Due: 15 October 2050.

WP2 Software Requirements Analysis and Specification.

See any book on software engineering.

Should include **risk analysis** – identify work/tasks that threaten the success of the project. They must be identified so that their severity can be addressed early, and, if necessary, backup strategies devised.

Output/Deliverable: Software Requirements Document. Due: xx Decober 2050.

Note: This feeds into Requirements and Analysis chapter of final report.

WP3 Software Design.

See any book on software engineering.

Output: Software Design and relevant chapter of final report.

WP4 Implementation.

Build and test the software. Document (as necessary).

Output: Working Software Code. Software documentation and test results. Include in final report.

WP5 Test and Evaluation.

- Does the software do what it was supposed to do? Do objective testing and evaluation. How does this solution compare with others (that this is supposed to replace, or that you identified in your research and survey).
- Possibly usability evaluation;
- Possibly gather tables of results and evaluate them.

Output: Test/Evaluation chapter in final report

WP6 Reporting.

Generate the Final and any interim reports.

WP7 Management.

- Preparation of proposal, planning. Updating proposal, plan throughout project as necessary.
- Periodic progress meetings with supervisor.
- Preparation of progress reports for progress meetings.

Outputs: Project schedules. Project progress reports.

3.3.2 Project Schedule

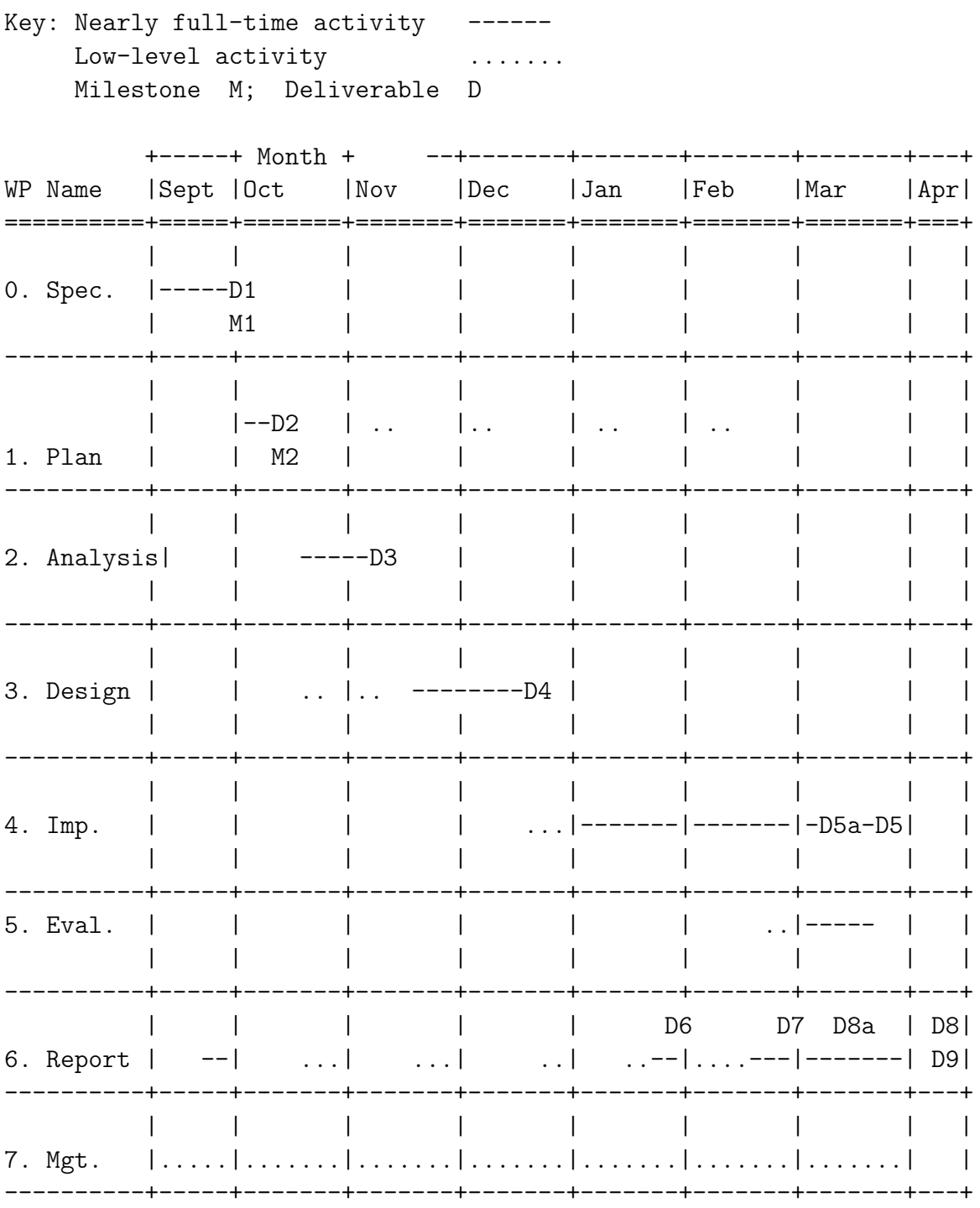
Effort Estimate Estimate, in person-hours, the time to complete each task.

Schedule Lay out on a calendar the schedule for the conduct of the WPs. Use the contents of the WP – or be guided by the necessity to finish on time! – and show how each WP will occupy part of the time between now and deadline for final report.

Usually, there will be dependencies, e.g. WPx must be finished before WPy starts. E.g. You can do very little Software design before the completion of Software Requirements. Specification (except, maybe, if you are using prototyping - or hacking!).

Best shown using a bar-chart — see Figure 3.1. Large projects with complex interdependencies require PERT or CPM – one person projects hardly ever need such complications. You can use a simple drawing like Figure 3.1, or, if you or your supervisor prefer, use something like Microsoft project.

Make sure to show deliverables; and list them, along with dates. Figure 3.1 is **only an example**; we have deliberately made it vague and incompatible with the Project Preparation and Development Project modules so that we do not receive carbon-copies of it. Make sure that *the plan covers both modules*.



Key to deliverables:

D1 Project Specification
D2 Project Planning Document.
D3 Software Requirements Document
D4 Design -- draft of chapter of final report.
D5 Software.
D5a Interim software for presentation.
D6 Draft 1 of Final Report.
D7 Draft 2 of Final Report.
D8 Final Report.
D8a Draft of Final Report for Interim Evaluation
D9 Presentation and oral.

Figure 3.1: Project Bar-chart

3.4 Progress Reporting

3.4.1 Introduction

This part will be dependent on the wishes of individual supervisors.

Normally progress meetings should be scheduled for a set time each week. The student is normally expected to prepare a brief progress report before the meeting. The student is responsible for preparing a record (minutes) of the meeting.

3.4.2 Progress Reports

The student should prepare a brief report under the following headings:

- Brief summary of technical achievement, problems.
- Status of actions from last progress meeting.
- Plans for next period (until next progress. meeting).
- (Maybe) Summarise under the following headings (SOFT):

  Satisfactory i.e. according to plan.

  Opportunities i.e. change the plan to improve the outcome of the project, without adversely affecting the feasibility.

  Failures i.e. not according to plan; must describe recovery plans, and prepare amended schedule

  Threats i.e. failures on the horizon. Outline avoidance.

Chapter 4

Reports and Disserations

4.1 Purpose

The purpose of this section is to suggest a *template* contents list for student project final reports, and to give guidelines on how to construct the report and its constituent sections.

4.2 Rationale

Most student project reports are a mess. Not because the work is bad, but because the report has no structure and because of inappropriate bad writing style.

The student project reports and dissertation is often the first substantial report writing task encountered; report writing is not easy. But, by starting with a proper structure / contents list, the task of writing can be considerably eased. Furthermore, and more crucially, the task of reading - by those who mark the projects - is made easier and the message get across.

Although report writing is a skill which must be learned, an appropriate layout focuses thought and makes the job fairly mechanical. It's just like software construction: design first and the job may succeed; start coding immediately and you end up with a mess.

4.3 Layout Guidelines and Style

Writing reports is a skill. Like any other it can be learned.

It doesn't matter what the project title (wheel-barrow design, software design, . . .), the structure of the report will remain largely unchanged. Also, you will find that different sizes or levels of report, or books or papers, can use the same structure.

The LaTeXdocumentation preparation system that you were taught in Year 3, can make life easy — this document is an example.

Style Fortunately, in report writing, simple mechanical procedures can be used to improve poor writing style. See especially (Turabian 2004), (John Grossman (ed.) 1993), (Williams 1995), (Murray 2006) and (Dupré 1998); there are many other relevant books and reports in the bibliography.

Bear in mind the following:

- If your English is poor, then avoid the difficult parts (long sentences etc.);
- Almost certainly, your report must be a logical, sequenced argument:
  1. analysis of problem (maybe biased towards your solution);
  2. description of your solution - trying your best to persuade the reader that you chose the best solution, and that you really know what you are talking about;
  3. a mature and objective evaluation of the work;
  4. summary and conclusions — this must get across the main items for those who are too busy (or lazy) to read the whole report.
- All parts must structured into sections / chapter, subsections, paragraphs, sentences;
- A paragraph should get across one (only one!) coherent set of ideas or an argument; paragraphs should be used to separate, for example, different themes in your argument;
- Unless you are a very skilled writer, a sentence should express one single idea, concept, or fact;
- We all know about readability in programs: indentation, use of white-space – do the same for your report. A page of English that is not broken by a diagram or new section heading will put most readers to sleep;
- Use lists to save you the problem of constructing complex English sentences and paragraphs; but note that isolated single phrases hardly ever explain anything, or provide much information;
- Similarly, tables and diagrams - to summarise; lead or follow-up with the English; (obviously) all Figures and Tables must be referenced and/or described in the text;
- The subject about which you are writing is computing; i.e. there may be numbers involved, and formulae, and equations. Or, for non-numeric data, there are processes, which have inputs and outputs, and functions that can be described algorithmically. That is, precision is inevitable: equations, graphs, diagrams, process descriptions in structured English, formal specifications, data flow diagrams, . . . ;
- Your style needs to be fairly formal. For example, this document (the one you are reading) is probably written less formally than would do for a project report. It is a formal technical document that you are writing, don't write like you are writing for the Sun newspaper, or a comic;
- Avoid subjective statements like: "Visual Basic is useless for large projects", better "We considered Visual Basic to be unsuitable for this project because..." give good reasons... better still quote some authoritative textbook or paper;

- The inclusion of screen-dumps is often very helpful;
- A diagram can be worth 1000 words;
- An example can be worth 2000 words;
- Never underestimate the ability of a reader to misunderstand; if in doubt repeat your statement in two or more ways, from two or more perspectives;
- Use a spelling checker and/or get a friend to check your report; this is not the supervisor's responsibility.

4.3.1 References and Bibliography

You **must** use the Harvard citation system (as used in this document); for example (Dupré 1998). Apart from the citation system, it is important that all references have information sufficient to enable it to be accessed:

- Author(s).
- Title.
- Publisher.
- Year, for a book.
- For a journal paper: Journal name, Volume, Part, page numbers, Date.
- Web documents must be cited properly. You should give: author (give the organisation if the author's name is not indicated), title, organisation or company, date produced, indication that the document was accessed on the web (available on-line at: http:/www.xxx.yyy.zzz/), date accessed.

4.4 Contents — Project Preparation Final Report

There are *four* major parts, the title page, service pages, the body of the report, and appendices.

4.4.1 Title Page

The title page should be of the form:

Title of Dissertation
A Dissertation
submitted in partial fulfillment of
the requirements for the
DEGREE BSc in Applied Computing
in the Department of Computing
Letterkenny Institute of Technology
by
Your Name
Date of Submission

4.4.2 Service Pages

Abstract or Summary .

The summary is composed of answers to the following questions in three-quarters of a page or less.

1. What was problem to be solved, or what had to be done?
2. What was your approach?
3. What were the major outcomes?
4. What are the major conclusions?

Each of these should be answered with a single paragraph, with no more than three sentences.

Acknowledgments Thank anyone who helped/contributed to your project. Also anyone who provided financial support.

Declaration page Page which contains a signed declaration that this is your own work.

Contents List A list of chapters, sections, subsections, appendices, with page numbers.

4.4.3 Body of Report

[This is only a a suggestion.]

```
1. Introduction.

Tells the reader what the document is, and how to read it.

1.1 General (or Purpose and Scope)

One sentence stating that report is '... dissertation on...
project on the subject of ... leading to the degree of...'

One or two sentences stating what work was about - could copy
from Abstract.

1.2 Background and Objective

State briefly why this project was undertaken. Where was
starting point, where were you aiming. E.g. a HCI project to
evaluate a new design technique: (1) summarise the
inadequacies of current design techniques, (2) introduce (one
or two sentences) the new technique, and say why it is
promising, and (3) say how you aimed to implement and
evaluate it (this is the objective). Or, you might be
continuing from another project, or exploring some aspect of
your supervisor's research - in which case, the background is
a description of that project or research.

1.3 Outline of requirements.

1.4 Outline of Report.

e.g. This report is divided into X sections. Section 2
contains a literature and product survey. Section 3
contains... Section X-1 contains conclusions and
recommendations for further work. Section X contains
references.

2. Survey

Literature and/or product survey. Show that you are
knowledgeable about the subject of your project.

3. Requirements Analysis and Specification

Based on your stated objectives (Introduction) and on the Survey,
```

```
analyse and, finally, state in a clear and unambiguous way what are the
software/product requirements (see any textbook on software
engineering).

4. Design

This is very difficult to write - especially if you hacked the
software). If you read this BEFORE you have developed the
software, or done the technical work (whatever), remember that
you have to write a report on what you have done. If you
proceed in a logical, rational, manner it will be easy to
report.
          ... etc

5. Implementation.

6. Testing and Evaluation.

7. Conclusions.

Say what you did (see Abstract - though could be more
verbose here), what were major problems, lessons learned,
conclusions drawn.

Since part of the purpose of the project is to have you  apply
some of what you learned, it might be no harm to mention such
topics. Certainly mention any difficulties you encountered and
boast about how cleverly you solved them.

8. References and Bibliography.
```

4.4.4 Appendices

Place program listings in Appendices. The following should be included:

1. Annotated listings of all code developed during the project;
2. A brief user manual giving precise details on how to use the software, and examples of input and the corresponding output from the software.

.

4.5 Diagram and Table Presentation and Numbering

All diagrams and pictures (Figures) and Tables must be titled and numbered, above or below the Figure in question. The Figure must be introduced or otherwise referenced from the test. The best numbering method is to number figures:

Figure 1-1 (e.g.) Software Sales 1985-1992, for section 1, Figure 2-1, 2-2 for section 2, ... Similarly, Table 1-1, etc.

Place the Figure close to the text that refers to it.

4.6 Plagiarism

Plagiarism is the direct or indirect copying of the words or ideas of another writer (student project, textbook, . . .). If you directly copy the words of another, you must enclose them in quotes, **and** cite the source. If you copy or describe the idea, e.g. in a literature survey and requirements analysis, you must cite the source. You should make yourself aware of the Institute statement on plagiarism contained in the Student Handbook.

All reports will be required to include a form containing a signed declaration that the document is entirely the student's own work.

We have been directed by the external examiner to pay particular attention to plagiarism in the dissertation and in software. We will be using a package such as `www.turnitin.com` or similar to automatically scan reports.

More importantly, during interim and final presentations, we will be demanding that students explain parts of their software. To quote the external examiner, *if the student cannot explain the workings of a piece of software during the presentation, the student must fail*.

When submitting source software in hard copy or in machine readable form, the software must be commented sufficiently that it is immediately obvious what software is (a) created by you, and (b) what has been downloaded or otherwise acquired.

Bibliography

Barrass, R. (1995). *Students Must Write*, 2nd edn, Routledge.

Bates, B. (2001). *Game Design: the art and business of creating games*, Prima Tech (Prima Publishing). ISBN: 0-7615-3165-3.

Blicq, R. (1987). *Writing Reports to Get Results: Guidelines for the Computer Age*, IEEE Press.

Bolker, J. (1998). *Writing Your Dissertation in Fifteen Minutes a Day: a guide to starting, revising, and finishing your doctoral thesis*, Prima Publishing / Henry Holt.

Booth, W., Colomb, G. & Williams, J. (2008). *The Craft of Research (Chicago Guides to Writing, Editing & Publishing)*, 3rd edn, Chicago University Press. ISBN-10: 0226065669.

Cooper, B. M. (1984). *Writing Technical Reports*, Penguin.

Dawson, C. (1999). *The Essence of Computing Projects: A Student's Guide*, Prentice Hall. Later edition: Projects in Computing and Information Systems, Addison-Wesley 2005.

Dickheiser, M. J. (2007). *C++ for Game Programmers*, 2nd edn, Charles River Media. ISBN: 1-58450-452-8. This is the second edition of a first edition by Llopis.

Diller, A. (1999). *LaTeX Line by Line*, 2nd edn, Wiley.

Dupré, L. (1998). *Bugs in Writing: a guide to debugging your prose*, Addison-Wesley.

Forsyth, P. (1997). *How to be better at Writing Reports and Proposals*, Kogan Page.

Freeman, E. & Freeman, E. (2006). *Head First HTML with CSS & XHTML*, O'Reilly.

Goossens, M., Mittelbach, F. & Samarin, A. (1994). *The LaTeX Companion*, Addison-Wesley.

Gowers, E., Baum, S. G. & (rev.), J. W. (1987). *The Complete Plain Words*, Penguin.

John Grossman (ed.) (1993). *The Chicago Manual of Style*, 14th edn, University of Chicago Press.

Kopka, H. & Daly, P. (1999). *A Guide to LaTeX*, 3rd edn, Addison-Wesley.

Krug, S. (2000). *Don't Make Me Think! a common sense approach to web usability*, 2nd edn, New Riders.

Lamport, L. (1994). *Latex User's Guide and Reference Manual*, 2nd edn, Addison-Wesley.

McShaffry, M. (2005). *Game coding complete*, 2nd edn, Paraglyph Press. ISBN: 1932111913.

Microsoft (1995). *The Microsoft Manual of Style for Technical Publications*, Microsoft Press.

Mort, S. (1992). *Professional Report Writing*, Gower.

Murray, R. (2006). *How to Write a Thesis*, 2nd edn, Open University Press. ISBN-10: 0335219683.

Nielsen, J. (1999). *Designing Web Usability*, New Riders.

Nielsen, J. & Tahir, M. (2002). *Homepage Usability, 50 Websites Deconstructed*, New Riders.

O'Luanaigh, P. (2006). *Game Design Complete*, Paraglyph Press. ISBN: 1-93309700-0.

Phillips, E. M. & Pugh, D. S. (2000). *How to Get a PhD: A Handbook for Students and Their Supervisors*, 3rd edn, Open University Press.

Strunk, W. & White, E. (1979). *The Elements of Style*, 3rd edn, Macmillan. Online at: http://www.bartleby.com/141/; accessed 2009-02-06.

Sun Technical Publications (1996). *Read Me First!: A Style Guide for the Computer Industry*, Prentice Hall.

Turabian, K. L. (2004). *A Manual for Writers of Term Papers, Theses and Dissertations (Chicago Guides to Writing, Editing & Publishing)*, Chicago University Press.

Williams, J. M. (1995). *Style: Toward Clarity and Grace (Chicago Guides to Writing, Editing & Publishing)*, Chicago University Press.

Appendix A

Project Proposal Specification

The project specification should be prepared according to the following format and layout. An electronic copy must be available for eventual inclusion in a web site. The specification should be two pages or less (normal font size). **Please attach a photocopy of your student card**.

Format and Layout

BSc Applied Computing Final Year Project, Letterkenny I.T

Project Specification

Student name, id., email address: Joe Bloggs, Id 12345, `j.bloggs@yahoo.ie`.

Title: (a phrase which describes the project).

Objectives:

A few bullet points or sentences identifying what hope to be the outcome of the project

Description. Fifty or so words describing (for example):

1. the origin of the idea — possibly give a reference to a book, e.g. (Knuth, 1997) or article, or to a discussion with a staff member;
2. why is this worth making/studying?
3. what is current state of the art (but be brief);
4. the approach you will take; obviously at this stage you cannot give a full design, but do your best to allow us to assess the realism/feasibility of your idea.

Resources required:

Hardware, software, other; if some special software or hardware is needed, you must flag this immediately so that we can either set about acquiring it, or telling you to get another project. If you will use resources of your own, state that; and make sure it be possible to demonstrate the product in the Institute? Will you need a web server? Etc.

References: (for example)

D.E. Knuth, The Art of Computer Programming, Volume 1, 3rd ed., Addison-Wesley, 1997, pp. 290-292.

Sun Microystems, Java Biometrics Applet, http://www.java.sun.com/biometrics/.

Appendix B

Software Requirements Specification (SRS) Document

A suggested Contents List is as follows.

```
   Title Page

   Contents List

   1 Introduction

   1.1 Purpose

   Tell the reader that this is the Software Requirements Specification
(SRS) Document for the Project Preparation module at Letterkenny ...
on the topic
   of ...

   1.2 Background and Objectives

   In the case of software development, you should describe clearly
   what the software should do (functional requirements), what the user
   interface should be, what sort of hardware the software should run
   on, compiler restrictions etc.

   A block diagram showing interaction of the (built) system with other
   systems, and with users, data sources etc. will help a lot.

   It will help a lot to give a rationale for your project -- why is it
   worthwhile.

   1.3 Outline of this report.
```

See any book or technical report. This section should give a one sentence summary of each chapter. I know you are going to say, "there is a contents list", so why do we need more. However, documents with this section included impress readers.

2. Technology Survey and Background

2.1 Introduction

2.2 Literature Survey

This gives a tutorial/academic survey of the chosen areas (e.g. human-computer interaction, image processing, client/server/distributed computing). This has two purposes: (a) to give a foundation for other parts -- so that when you mention a term, it will have been introduced here. Or, when you choose one of competing technologies, you will have summarised some academic papers and textbooks that provide justification.

You should show that you are aware of current trends. Text-books are okay, but most of the work is done for you. Web, okay, but may not impress some examiners. Ideally, you should show that you have read (and synthesised here) and cited a number of key sources.

Learn how to cite other literature and the web. See the main document.

2.3 Product (Technology?) Survey

For example: (a) of competing and related products; (b) of tools to be used (e.g. PHP, ASP, MySQL, UML, Rational Rose, ...);

Note: you may wish to use separate chapters for Literature and Product/Technology.

3. Requirements

What we need here is a clear and
detailed statement of the goals of your project. This has already partially been given in your project specification. We now need more detail, perhaps revision, after you have thought more about the project and discussed it.

Note: if we do not get a clear view of the goals of the project, we will be duty bound to award low marks.

If you are extending currently software (your's or someone else's)

it is absolutely essential that you state your starting point clearly and unambiguously.

I think it would be worth the effort to provide a decomposition of high-level requirements into lower level requirements.

It may be worth performing some 'risk analysis', see the general guidelines. Big risks (e.g. you need a Linux server with PHP and MySQL, or a similar Microsoft system ...) need to be addressed NOW.

We would expect risks to be identified and discussed with plausible workarounds identified, e.g. if the college provides no PHP capable server, then maybe you can install Linux and Apache (and PHP) on you own machine.

It may be worth reporting the results of prototyping -- perhaps to resolve some risk, e.g. can you get the Java language to talk to a certain database manager?

Use cases. Other UML diagrams.

Test plan.

Games students should refer to the BSc 3 Team Project documentation,

http://www.jgcampbell.com/teamp/

4. Prelimimary Design

It may be easier to express requirements in the form of a preliminary design.

UML diagrams of classes and their inter-relationships?

Again, games students should refer to the BSc 3 Team Project documentation, http://www.jgcampbell.com/teamp/

5. Conclusions

Summarise the document. Identify significant decisions. Identify significant problems. Identify significant progress.

6. References

There should be references. Give references in a standard format. See 'Survey' above, 'citation'; see the 'general guidelines'.

Appendix C

Syllabuses from the Course Document

The following subsections quote from the course document.

C.1 Project Preparation (Semester 1)

C.1.1 Module Aims

- To give students practice in researching and preparing for a personal software development project similar to what they will be expected to perform at the beginning of their professional career.
- To develop in students the ability, with limited supervision, to work with from informal requirements through feasibility study, requirements capture and requirements analysis phases of a software development project.

C.1.2 Module Learning Outcomes

Upon completion of this module students will be able to:

1. .From an informal statement of requirements and broad aims, develop and document detailed aims and objectives;
2. Research and document a feasibility study on the project topic;
3. Prepare a plan for a programme of work showing milestones, deliverables and phasing of tasks;
4. Perform requirements capture (depends on project);
5. .Analyse requirements and document a detailed software requirements specification (SRS).

C.1.3 Syllabus Content

Section A. Feasibility study. (35%) Choice of topic and preparation of a feasibility study. Literature and product survey.

Section B. Project Planning. (10%) Prepare a plan for a programme of work showing milestones, deliverables and phasing of tasks. Iterated.

Section C. Requirements Capture and Documentation. (15%)

Section D. Requirements Analysis and Documentation. Output: Software Requirements Specification (SRS) (40%) Include risk analysis, prototyping, iteration, as necessary.

C.2 Development Project (Semester 2)

C.2.1 Module Aims

- To give students practice in performing a personal software development project similar to what they will be expected to perform at the beginning of their professional career.
- To develop in students the ability, with limited supervision, to work with from informal requirements through all the phases of a software development project.
- To develop in students the ability to summarise a project in a positive manner.

C.2.2 Module Learning Outcomes

Upon completion of this module students will be able to:

1. Develop a project approach and system design;
2. Perform project planning, including production of periodic progress reports and iteration as needed;
3. Know how to take responsibility for the request and acquisition of assistance and other inputs from a range of sources;
4. Perform the implementation of a series of prototypes and subsystems with evaluation and suggested revision at each step;
5. Implement and test a final software product;
6. Evaluate the products (analysis, design, implementation) of the project;

7. Prepare a final report on the work undertaken in the project;
8. Do a summarising presentation based on the final report and a demonstration of the key features of any artefact produced: project objectives, conduct of project, major outcomes, and major conclusions.

C.2.3 Syllabus Content

Section A. Plan iteration. (5%) Periodically evaluate and revise project plan, including production of periodic progress reports.

Section B. Architectural design. (5%) Starting with a Software Requirements Specification document (Project Preparation Module) develop a project approach and system design; iterate as necessary; document with associated justification.

Section C. Detailed design. (20%) Design. Implement a series of prototypes and subsystems with evaluation and suggested revision at each step;

Section D. Implementation and test. (40%) Implement and test a final software product/

Section E. Evaluation and system test. (10%) Evaluate the products (analysis, design, implementation) of the project;

Section F. Final report. (15%) Prepare a final report on the work undertaken in the project.

Section G. Presentation. (5%) Preparation of a summarising presentation based on the final report and a demonstration of the key features of any artefact produced. Provide a summary of: project objectives, conduct of project, major outcomes, and major conclusions.

C.2.4 Module Assessment

Continuous Assessment (Draft Dissertation): 15%; Final Dissertation: 85%.

Appendix D

Marking Schemes

D.1 Marking Scheme for Project Preparation

The learner will be required to submit two pieces of continuous assessment (interim reports) with the following weightings:

Assignment 1. Interim plan and feasibility study. 12%.

Assignment 2. Draft final report. 13%

D.1.1 Dissertation Assessment Criteria

Excellent. 80–100% Each of the feasibility study, plan and requirements specification is outstanding. The feasibility study shows profound understanding of the area of study; the requirements specification gives great clarity on what is to be implemented.

Background research is presented with original critical and analytical assessment of the major developments and theories in the area; primary and secondary sources are described with great clarity. Presentation and referencing are flawless.

Very good. 70–79% Each of the feasibility study, plan and requirements specification show mastery. The feasibility study shows complete understanding of the area of study; the requirements specification gives complete clarity on what is to be implemented.

Background research is presented with critical and analytical assessment of the major developments and theories in the area; sources are clearly identified, described and analysed. Presentation and referencing are almost flawless.

Good. 63-69% Each of the feasibility study, plan and requirements specification show a mature grasp of the task in hand. The feasibility study shows significant understanding of the area of study with some minor gaps; the requirements specification gives clarity on what is to be implemented, but there may be some requirements that are fuzzily described and untestable.

Background research is presented with good analytical assessment of significant contributions; there may be some gaps in the learner's knowledge of the field. Presentation and referencing are satisfactory.

Fair. 55–62% Each of the feasibility study, plan and requirements specification show a good working knowledge of the task being undertaken. The feasibility study may show significant gaps (which will need to be remedied in module DEPRCO801); the requirements specification gives a broad indication of what is to be implemented, but there are areas that need to be clarified in the early stages of module DEPRCO801.

Background research is presented as a collection of facts with much padding and repetition. There may be significant gaps in the learner's knowledge of the field. Presentation and referencing may indicate the need for further proof reading and revision.

Pass. 45–54% One or more of the feasibility study, plan and requirements specification shows a sketchy knowledge of the task undertaken. The feasibility study may show serious gaps in the learner's knowledge of the field; these gaps will need to be remedied in the early stages in module DEPRCO801; the requirements specification gives an indication of what is to be implemented only at the level

of broad aims; there are many requirements that need to be clarified in the early stages of module Development Project. The learner may need to seek tutorial assistance during module Development Project.

Background research is presented as minimal collection of facts. The learner's knowledge of the field is superficial. Presentation and referencing are sloppy and indicate the need for significant further proof reading and revision. The presentation of ideas may border on plagiarism.

Marginal pass. 35–45% One or more of the feasibility study, plan and requirements specification shows a seriously inadequate knowledge of the task undertaken. The feasibility study may show that the learner's knowledge of the field is no better that would achieve a 40% mark in a level 7 assignment. The learner's ability to progress to module Development Project is in serious question. At the very least, the learner will need to seek tutorial assistance during module Development Project.

Background research is presented as small collection of facts. The learner's knowledge of the field is almost non existent. Presentation and referencing are sloppy and indicative of a first or second draft. The presentation of ideas may border on plagiarism.

Fail. 15–35% Each of the feasibility study, plan and requirements specification shows a seriously inadequate knowledge of the task undertaken. The feasibility study may show that the learner's knowledge of the field would not achieve a 40mark in a level 7 assignment. The learner's ability is probably incapable of progressing to module Development Project. At the very least, the learner will need to seek special supervision and tutoring during module Development Project; in addition, the learner may need counselling to change to a simple project topic where there is some chance of obtaining a pass in Development Project.

Background research is presented as small collection of facts. The learner's knowledge of the field is almost non existent. Presentation is at the level of a carelessly produced first draft; referencing is incorrect or non-existent. The presentation of ideas may contain plagiarism.

Oral Examinations and Presentations In all cases, examiners may call any candidate to defend his or her dissertation in a presentation; or, if a short presentation is the norm, an additional oral examination may be required in order to determine mark/grade.

Plagiarism Suspicion of plagiarism will be taken extremely seriously. Where the there is evidence of plagiarism of ideas or software, the project should fail with a zero mark awarded.

D.2 Marking Scheme for Development Project

Continuous Assessment The learner will be required to submit one interim report — draft final dissertation report, this is worth 15% of the module.

D.2.1 Dissertation Assessment Criteria

Excellent. 80–100% The dissertation report and software or experimental results are outstanding original contributions. 1. In the case of software, the work is suggestive of a capability of working on innovative projects and producing high quality software with a minimum of supervision. The product produced could be the basis of a revenue generating product. 2. In the case of an study / evaluation of a new area, the work could be submitted to an academic conference or journal with little polishing.

The project was performed immaculately, with difficulties identified in a timely manner and resolution achieved quickly and/or backup solutions obtained. The project plan was developed, updated and followed, showing considerable vision and maturity.

Background research is presented with original critical and analytical assessment of the major developments and theories in the area; primary and secondary sources are described with great clarity. Presentation and referencing are flawless.

Very good. 70–79% The dissertation report and software or experimental results show great understanding and skill. 1. In the case of software, the work is suggestive of a capability of working on innovative projects and producing high quality software with a minimum of supervision. 2. In the case of an study / evaluation of a new area, the work is suggestive of migration of postgraduate research with great ease.

The project was performed in an orderly manner, with difficulties identified in a timely manner and resolution achieved quickly and/or backup solutions obtained, perhaps with supervisory assistance. Agreed plans were respected and regularly updated in a proactive manner.

Background research is presented with critical and analytical assessment of the major developments and theories in the area; sources are clearly identified, described and analysed. Presentation and referencing are almost flawless.

Good. 63-69% The dissertation report and software or experimental results show understanding and skill. 1. In the case of software, the work is suggestive of a capability of producing high quality software with a minimum of supervision. 2. In the case of an study / evaluation of a new area, the work is suggestive of postgraduate research capability.

The project was performed according to supervisory guidance, with difficulties identified quickly and/or backup solutions obtained, mostly with supervisory assistance. Agreed plans were respected.

Background research is presented with good analytical assessment of significant contributions; there may be some gaps in the learner's knowledge of the field. Presentation and referencing are satisfactory.

Fair. 55–62% The dissertation report and software or experimental results show limited understanding and skill. 1. In the case of software, the work is suggestive of a capability of producing software with a significant supervision. 2. In the case of an study / evaluation of a new area, the work contains a collection of ideas perhaps useful for a further project.

The project was performed according to close supervisory guidance. Agreed plans were adhered to for most of the time.

Background research is presented as a collection of facts with much padding and repetition. There may be significant gaps in the learner's knowledge of the field. Presentation and referencing may indicate the need for further proof reading and revision.

Pass. 45–54% The dissertation report and software or experimental results show flawed understanding and skill. 1. In the case of software, the work is suggestive of a capability of producing software code from detailed specifications and with close supervision. 2. In the case of an study / evaluation of a new area, the work contains a very basic collection of ideas that is little better than, for example, an essay assignment that would gain II.1 marks in a level 6 class.

The project often required close supervisory guidance, including technical assistance. Adherence to agreed plans may have been patchy.

Background research is presented as minimal collection of facts. The learner's knowledge of the field is superficial. Presentation and referencing are sloppy and indicate the need for significant further proof reading and revision. The presentation of ideas may border on plagiarism.

Marginal pass. 35–45% The dissertation report and software or experimental results show almost no understanding and skill. 1. In the case of software, the work is suggestive of a capability of producing software code from very detailed specifications and with very close (day to day) supervision. 2. In the case of an study / evaluation of a new area, the work contains a minimal collection of ideas that is little better than, for example, an essay assignment that would gain pass marks in a level 6 class.

The project required very close supervisory guidance, including detailed and regular technical assistance. Agreed plans may have been regularly ignored.

Background research is presented as small collection of facts. The learner's knowledge of the field is almost non existent. Presentation and referencing are sloppy and indicative of a first or second draft. The presentation of ideas may border on plagiarism.

Fail. 15–35% The dissertation report and software or experimental results show almost no understanding and skill. 1. In the case of software, the work is suggestive of almost no capability of producing software code no matter how close the supervision. 2. In the case of an study / evaluation of a new area, the work contains a minimal collection of ideas that would fail an essay assignment in a level 7 class.

The project required very close supervisory guidance, including detailed and regular technical assistance, but because of poor attendance or lack of interest, supervision was largely ineffective. Agreed plans may have been regularly ignored.

Background research is presented as small collection of facts. The learner's knowledge of the field is almost non existent. Presentation is at the level of a carelessly produced first draft; referencing is incorrect or non-existent. The presentation of ideas may contain plagiarism.

Oral Examinations and Presentations In all cases, examiners may call any candidate to defend his or her dissertation in a presentation; or, if a short presentation is the norm, an additional oral examination may be required in order to determine mark/grade.

Plagiarism Suspicion of plagiarism will be taken extremely seriously. Where the there is evidence of plagiarism of ideas or software, the project should fail with a zero mark awarded.