MONASH University
Information Technology

FIT3036
Computer science project

Unit Guide

Semester 2, 2010

Last updated: 14 Jul 2010

Table of Contents

FIT3036 Computer science project - Semester 2, 2010

Chief Examiner:

Dr Peter Tischer
Senior Lecturer
Phone: +61 3 990 55208
Fax: +61 3 990 55159

Contact hours: 2.00pm - 300pm Thursdays

Lecturer(s) / Leader(s):

Clayton

Associate Professor Bin Qiu
Associate Professor
Phone: +61 3 990 59688
Fax: +61 3 990 55159

Dr Peter Tischer
Senior Lecturer
Phone: +61 3 990 55208
Fax: +61 3 990 55159

Malaysia

Introduction

Welcome to FIT3036 Third Year Project. It is a 6 credit point unit, offered in first and second semester during 2010. The unit gives you an opportunity to tackle a significantly larger software design and implementation than you have been able to do previously in your course. You will work on a project allocated to you (please remember to identify your preferences) under the supervision of an academic staff member. The nature of the project is largely unconstrained, and will be determined in consultation with your project supervisor.

Unit synopsis

This unit is intended to provide practical experience in designing, developing and testing a non-trivial computer science project. Projects are generally software-based, although sometimes they may involve hardware development or investigation of theory. Projects cover the whole process of software (or hardware) development, from analysis through design to implementation and testing. Comprehensive written documentation on the project is required. Students are assigned in groups to a project supervisor. There are no lectures in this unit, although students will be expected to attend regular meetings with their project supervisor.

Learning outcomes

At the completion of this unit students will have -
A knowledge and understanding of:

- strategies for developing a non-trivial programming, hardware, or theory-based project.
- how to locate and utilise prior research and methods on a particular topic;
- how to cite bibliographic references the student has used to understand various components of the project, support claims on knowledge, events, hypotheses and theories;
- how to document software development from a user and application programming perspective;
- software development methods: analysis, design, implementation and testing applied to the design and development of a non-trivial project.

Developed attitudes that enable them to:

- acknowledge the importance of attending and contributing to meetings as a method of gaining important information and ideas about the project;
- understand the basic requirements of software development from both user and developer perspectives;
- appreciate the importance of correctly acknowledging the work of others in researching solutions to problems;
- value the role of work books in documenting a projects progress and keeping track of its development.

Developed the skills to:

- search, access, and analyse research literature as part of the process of developing solutions to problems;
- understand the importance of analysis, design, documentation, and testing in developing a non-trivial software project;
- write a moderately detailed report explaining methodology, outlining their contributions and the contributions of others, documenting the developed project from developer and user perspectives.

Demonstrated the communication skills necessary to:

- understand the role of the client (or user) in the software development process;
- appreciate the importance of written communication in documenting project development;
- understand the importance of assessing time and resource requirements in the successful completion of non-trivial projects;
- appreciate the importance of time and resource management in order to deliver non-trivial projects to deadlines.

Contact hours

1 hr project meeting/week

Workload

The university standard for a 6-credit point unit is 12 hours of work per week over a semester. Students must be prepared to commit to at least 8 hours of private study per week on this unit, in addition to the contact hours (1 hour per week)

Unit relationships

Prerequisites

One of FIT2004 or CSE2304 or CSE2040 and one of FIT2001, FIT2024, CSE2305 or CSE2050

Prohibitions

CSE3301

Teaching and learning method

Teaching approach

Weekly project group meetings: 1 hour per week

Individual design, coding, testing: 9 hours per week

Timetable information

For information on timetabling for on-campus classes please refer to MUTTS,
http://mutts.monash.edu.au/MUTTS/

Tutorial allocation

On-campus students should register for tutorials/laboratories using the Allocate+ system:
http://allocate.its.monash.edu.au/

Off-Campus Learning or flexible delivery

CONTRARY TO WHAT IS STATED ABOVE, THERE ARE NO TUTORIALS IN THIS UNIT.

Unit Schedule

Week	Date*	Topic	Key dates
1	19/07/10	Preliminary reading & Project selection	22 & 23 July
2	26/07/10	Preliminary reading	31 July
3	02/08/10	Plan of Attack	
4	09/08/10	Milestone 1	14 August
8	06/09/10	Milestone 2	11 September
10	20/09/10	Milestone 3	25 September
Mid semester break			
12	11/10/10	Milestone 4	16 October
13	18/10/10	Completion and Submission of report, and final demonstration	23 October

*Please note that these dates may only apply to Australian campuses of Monash University. Off-shore students need to check the dates with their unit leader.

Improvements to this unit

Student responses to FIT3036 were positive in 2009 and no changes are currently planned.

Unit Resources

Prescribed text(s) and readings

There are no prescribed text books for this unit. Individual project supervisors may recommend additional reading.

Recommended text(s) and readings

Any textbooks required will be determined by individual project supervisors on a case-by-case basis.

Required software and/or hardware

Projects will normally need access to a computer and programming environment. Individual requirements will be identified by project supervisors

Equipment and consumables required or provided

Students studying off-campus are required to have the minimum system configuration specified by the Faculty as a condition of accepting admission, and regular Internet access.On-campus students, and those studying at supported study locations may use the facilities available in the computing labs.Information about computer use for students is available from the ITS Student Resource Guide in the Monash University Handbook.You will need to allocate time each week for use of a computer, including time for newsgroups/discussion groups.

You need to be aware that the computing resources that may be supplied to you in order to undertake this unit cost real money, and the university has had to impose limits on the use of the internet for all staff and students. A quota system is to be introduced this year, but in the meantime you must make yourself aware of the "Acceptable Use" policies of both the faculty and the university.

These can be accessed on the web at:

- http://www.infotech.monash.edu.au/myfit/students/student_labinfo_rules_netusage.cfm
- http://www.adm.monash.edu.au/unisec/pol/itec12.html

Note that accessing these and other course-related URLs from within Monash is free from within the Monash network, and is not regarded as part of quota.

Study resources

Study resources we will provide for your study are:

During semester 2, 2010, the unit will be supported by Moodle.

Assessment

Overview

Projects are assessed by individual project supervisors.

Faculty assessment policy

To pass a unit which includes an examination as part of the assessment a student must obtain:

- 40% or more in the unit's examination, and
- 40% or more in the unit's total non-examination assessment, and
- an overall unit mark of 50% or more.

If a student does not achieve 40% or more in the unit examination or the unit non-examination total assessment, and the total mark for the unit is greater than 50% then a mark of no greater than 49-N will be recorded for the unit.

The unit is assessed on the basis of a completed project report, and work done during the semester (see below). There is no examination in this unit.

Assignment tasks

Assignment coversheets

Assignment coversheets are available via "Student Forms" on the Faculty website:
http://www.infotech.monash.edu.au/resources/student/forms/
You MUST submit a completed coversheet with all assignments, ensuring that the plagiarism declaration section is signed.

Assignment submission and return procedures, and assessment criteria will be specified with each assignment.

Assignment submission and preparation requirements will be detailed in each assignment specification. Submission must be made by the due date otherwise penalties will be enforced. You must negotiate any extensions formally with your campus unit leader via the in-semester special consideration process: http://www.infotech.monash.edu.au/resources/student/equity/special-consideration.html.

- **Assignment task 1**

  Title:
  Attendance
  Description:
  Meetings are usually held weekly at a time and place convenient to the individual supervisors and each project group. Times & locations will be listed on the third-year notice-board and the online project list (see link above) as soon as they are announced. The first meeting for each group will usually occur in the first week of semester so please check these lists until you have found the time for your first meeting.
  Weighting:
  10 = min(12x1, 10) marks
  Criteria for assessment:
  Attendance = 1 mark

Absence = 0 mark

Due date:

Weekly, as arranged with supervisor

- **Assignment task 2**

Title:

Achievement

Description:

This mark will be allocated by the project supervisor, and reflects the outcomes of the project as realised by the student

Weighting:

25 marks

Criteria for assessment:

To be advised by the individual project supervisors, during weekly discussions

Due date:

17 October

- **Assignment task 3**

Title:

Report

Description:

See the weekly schedule

Weighting:

30 marks

Criteria for assessment:

The Project Report provides a complete account of your efforts towards completing the assigned project.

The contents of the bulk of the report should be organized in a manner which allows the casual reader to quickly identify *what* you were aiming to achieve and *how much* has been achieved. At the same time, the serious reader of the report must be able to easily determine *how* your achievements have been realized.

One suggested organization for your report is given below; obviously the detailed format and contents is a matter of choice depending upon the nature of the project (especially for non-programming projects) and prior agreement between the student and the supervisor.

Unless told otherwise by the supervisor concerned, the report must be printed on A4 stationery with standard margins. It is anticipated that most reports will be between 8 and 16 pages in length.

2.1. Identification

(1a) Your Name

(1b) Your Student ID

(2) Project Title

(3) Report Date

(4) Supervisor's Name

(5) Access information for the supervisor/examiner to run the project software, such as, machine on which developed, usercode and password.

2.2. General Description

(1) A brief description (in your own words!) of the project task or tasks, including the relationship to other work (by previous students, the supervisor or your peers).

(2) Any theoretical material or reference material relevant to understanding either the task or the solutions to be evaluated and/or employed in the project. It is important to include information on references YOU have consulted and a discussion (where appropriate) of relevant previous work by others (including references to THEIR work).
References *must* include full bibliographic data, e.g. Nurd, Fred J. (1983): "Counting the Vertices of a Circle", *Journal of Irreproducible Results*, vol. 7, no 5,.pp 13-12.

(3) A discussion of any assumptions made in developing the solution to the problem, and an evaluation of alternative solutions.

(4) An outline of the method of attack.

2.3. Achievements

A summary, with particular emphasis on limitations of the solution and where the achievements are less than was anticipated (based on the statement of the problem) an explanation of why the goals were not reached.

2.4. Project Documentation

This section should provide a detailed description of the project solution aimed at both users and implementors (modifiers).

2.4.1. User Interface

Essential components of this section include:

1. How to start using the "thing" (program or lump of hardware); configuration and/or initialization parameters, defaults, etc.
2. Complete definition of the user input data and/or command syntax and semantics.
3. Types and meanings of results expected for the various user input options.
4. Permanent data files (e.g. input-output files, database files, libraries); names and uses.
5. Temporary files (e.g. sort scratch); names and uses
6. When the information is available provide estimates of resource consumption (e.g. data file sizes, run-times, report sizes) expressed as mean or typical values and maximum/minimum expected values.
7. In situations where error reporting and recovery is not self-evident, include a step-by-step description of how to recover from automatically detected errors, and a synopsis of error conditions which are not (or cannot be) detected except by the end user.
8. Where the solution provides simple and "bells and whistles" modes of operation, ensure that the user requiring only the simple facilities can readily determine how much of the user interface is relevant.

2.4.2. Implementation

Start with a functional description of what the solution does and a synopsis of how the solution has been structured (e.g. include a call tree or block diagram).

Include a detailed description of all file structures, significant data structures and non-trivial algorithms.

For each substantive module in the solution provide a description of sufficient detail to allow the reader to discover

1. What the module does
2. The interface between the module and the environment in which it must operate (e.g. pin assignments and signals, or parameter definitions and essential global data)
3. The method by which the module implements its stated function
4. The relationship between this module and other modules.

2.4.3. Evidence of Testing

Describe in general terms the tests which have been performed in an attempt to validate the correct operation of the project solution. The emphasis should be on *summarizing* the testing procedures that were employed, rather than including a box of listings and claiming this proves the program was tested.

Where practical, include the transcript of one or more invocations of the solution which demonstrate all the supported features. Such a transcript must be self contained to the extent that another person could reproduce the results using your solution (i.e. include all system-level commands, file assignments, input data, option settings and output data.)

2.5. Concluding Remarks

What has been achieved? Attempt to summarize your own accomplishments, as opposed to the prior achievements of others.

Suggested enhancements to overcome limitations, or to support additional useful facilities.

2.6. References

It is very rarely that a student completes a project without reference to the literature or the prior work of others. Include all books, articles and notes which you used during the course of the project.

2.7. Appendix A

The actual program listings or circuit diagrams which should be well commented and (where possible) cross-referenced.

REMEMBER, REQUIREMENTS VARY DEPENDING ON THE PROJECT AND YOU MUST CONFIRM WITH YOUR SUPERVISOR WHETHER THE ABOVE SCHEME IS APPROPRIATE FOR YOUR PARTICULAR PROJECT.

Due date:

31 Oct 2010

- **Assignment task 4**

  Title:
  Testing
  Description:
  Evidence that the software has been adequately tested
  Weighting:
  10 marks
  Criteria for assessment:
  Evidence that the software has been adequately tested
  Due date:
  17 October 2010

- **Assignment task 5**

  Title:
  Workbook
  Description:
  A notebook (or computer file) containing weekly entries describing what has been accomplished through the week. Details on how the workbook should be organized are supplied with the project details.
  Weighting:
  10 marks
  Criteria for assessment:
  At least 10 weekly entries
  Due date:
  Friday 17 October

- **Assignment task 6**

  Title:
  Interim Demonstration
  Description:
  A demonstration of the software in a working environment
  Weighting:
  5 marks
  Criteria for assessment:
  Students will lose 1 mark for each feature not adequately working
  Due date:
  held during the week of 23 August 2010

- **Assignment task 7**

  Title:
  Final Demonstration
  Description:
  A demonstration of the software in a working environment, including feedback from the Interim Demonstration
  Weighting:
  10 marks
  Criteria for assessment:
  Due date:
  held during the week of 11 October 2010

Due dates and extensions

Please make every effort to submit work by the due dates. It is your responsibility to structure your study program around assignment deadlines, family, work and other commitments. Factors such as normal work pressures, vacations, etc. are not regarded as appropriate reasons for granting extensions. Students are advised to NOT assume that granting of an extension is a matter of course.

Students requesting an extension for any assessment during semester (eg. Assignments, tests or presentations) are required to submit a Special Consideration application form (in-semester exam/assessment task), along with original copies of supporting documentation, directly to their lecturer within two working days before the assessment submission deadline. Lecturers will provide specific outcomes directly to students via email within 2 working days. The lecturer reserves the right to refuse late applications.

A copy of the email or other written communication of an extension must be attached to the assignment submission.

Refer to the Faculty Special consideration webpage or further details and to access application forms: http://www.infotech.monash.edu.au/resources/student/equity/special-consideration.html

Late assignment

Assignments received after the due date will be subject to a penalty of 10% per day per assignment (including weekends).

Return dates

Students can expect assignments to be returned within two weeks of the submission date or after receipt, whichever is later.

Appendix

Please visit the following URL: http://www.infotech.monash.edu.au/units/appendix.html for further information about:

- Continuous improvement
- Unit evaluations
- Communication, participation and feedback
- Library access
- Monash University Studies Online (MUSO)
- Plagiarism, cheating and collusion
- Register of counselling about plagiarism
- Non-discriminatory language
- Students with disability
- End of semester special consideration / deferred exams