MONASH University
Information Technology

# FIT3140
# Advanced programming

## Unit Guide

## Semester 1, 2013



*Last updated: 04 Mar 2013*

# Table of Contents

# FIT3140 Advanced programming - Semester 1, 2013

This unit develops the students' ability to design, implement and maintain moderately complex, realistically-sized programs using an Agile software development methodology. It builds upon the basic programming techniques introduced in introductory programming unit and offers the first introduction to the implementation of more complex real-world programs. Examples of such systems include compilers and interpreters, simulations, visualisation tools, drawing packages, database systems, graphical games. Such systems may be implemented in the context of non-traditional computing environments such as smartphone "apps". The unit may offer students the opportunity to get acquainted with a second programming language within the procedural-object oriented paradigm, such as C++, Python or one of their cousins, depending on the scope of the project chosen in a particular semester.

The unit bridges between core programming knowledge and the large-scale software engineering context. It will emphasize the implementation and use of intermediate to advanced data structures (such as search trees, hash structures, graphs and graph algorithms etc.) and the embedding into an actual computing system (i.e. interacting with the O/S, networking components etc).

## Mode of Delivery

Clayton (Day)

## Contact Hours

2 hrs lectures/week, 3 hr laboratory/week

## Workload requirements

You are expected to spend 12 hours per week during semester on this unit.

This includes 2 hours per week of lectures, 3 hours per week of labs, and an average of 7 hours per week of project work, private study, and exam revision.

Substantial parts of the project work must be completed in pairs.  Students will need to schedule time to meet with their project partners outside scheduled classes.

## Unit Relationships

### Prerequisites

FIT1008

## Chief Examiner

**Dr Robert Merkel**

# Campus Lecturer

## Clayton

**Robert Merkel**

# Academic Overview

## Learning Outcomes

Upon successful completion of the unit, students will have an understanding of:

- agile software development practices including iteration, test-driven development, spiking, and continuous customer involvement;
- how to design moderately complex programs where that design will typically incorporate a number of modules and a number of levels of refinement;
- the role of software architecture in program design and a knowledge of a number of commonly-applied software architectures;
- how to make use of design patterns, re-usable components and software libraries in designing modular software;
- how to make design decisions that take into account desirable quality attributes such as flexibility, maintainability and re-usability;
- how to implement programs in a systematic manner using an integrated testing procedure in such a way that modules are highly likely to function as specified;
- how to isolate faults within a program in a systematic manner;
- how to use software tools to aid in the program design and implementation process. These tools might include program design tools, integrated program development environments, configuration management systems, re-factoring tools, automatic testing environments and debuggers;
- how to adequately document a software project.

They will have developed attitudes that enable them to:

- evolve a software system in response to feedback over time;
- recognise the importance of process in achieving quality in a repeatable manner;
- appreciate the distinction between analysis of program requirements and design that seeks to meet specifications;
- develop software designs that place appropriate importance to the user experience;
- adopt an approach to making design decisions that involves considering a range of options for design decisions and evaluating potential design decisions with reference to a system of values;
- evaluate product and process development with the aim of continuously improving their software development methodology;
- understand the importance of being able to communicate all aspects of the program development process, and identify the most appropriate medium for that communication.

They will have developed the skills to:

- design moderately complex, real-world programs where that design involves multiple levels of refinement and the specification of a non-trivial number of modules;
- learn a new programming language or environment efficiently when that programming language or environment is similar to a programming language the student already knows;
- develop software in a modern software environment that may include software development tools such as those found in an integrated, programming environment, configuration management systems and automated testing systems;
- design and implement programs that can interface with complex software systems such as graphical-user interfaces, database systems and mathematical libraries;
- design and implement programs that may need to communicate via a computer network with software systems on other computer devices;
- design and implement systems on alternative computing platforms including smartphones and embedded systems;

- identify performance-critical aspects of a software system and learn to apply their analysis skills in larger systems.

They will have demonstrated the communication skills to:

- create design documents that can be used to present a view of the software to other stakeholders;
- identify items that are insufficiently clear from written documentation and proactively seek clarification from stakeholders;
- create documents or on-line help that enable people to understand how to use the program;
- create documents that allow a programmer to understand the program in sufficient detail to allow the software to be maintained;
- produce literate programs, i.e. program source statements that are well commented.

# Unit Schedule

| Week | Activities | Assessment |
|---|---|---|
| 0 | Register for lab classes | No formal assessment or activities are undertaken in week 0 |
| 1 | The problem domain - an introduction to cryptography | |
| 2 | Introduction to Agile development | Introductory programming assignment due Friday 15 March 2013 |
| 3 | Requirements gathering - an Agile approach | |
| 4 | The problem domain - welcome to Android | Vision statement (pairs) due Friday 29 March 2013 |
| 5 | Design and modeling | Spiking exercises (pairs) due Friday 12 April 2013 |
| 6 | Software Project Lifecycles | |
| 7 | Computer science in the real world - performance analysis and testing | Design and benchmarking (pairs) due Friday 26 April 2013 |
| 8 | Software engineering tools | |
| 9 | Design and modeling II - user interfaces | Iteration 1 (pairs) due Friday 10 May 2013 |
| 10 | Distributed applications | |
| 11 | Into the wild - releasing your software | Iteration 2 (pairs) due Friday 24 May 2013 |
| 12 | Summary and guest lecture | Final Report (individual) due Friday 31 May 2013 |
| | SWOT VAC | No formal assessment is undertaken in SWOT VAC |
| | Examination period | LINK to Assessment Policy: http://policy.monash.edu.au/policy-bank/academic/education/assessment/assessment-in-coursework-policy.html |

*Unit Schedule details will be maintained and communicated to you via your learning system.

# Assessment Summary

Examination (3 hours): 50%; In-semester assessment: 50%

| Assessment Task | Value | Due Date |
|---|---|---|
| Introductory programming assignment | 5% | Friday 15 March 2013 |
| Vision statement (pairs) | 5% | Friday 29 March 2013 |
| Spiking exercises (pairs) | 10% | Friday 12 April 2013 |
| Design and benchmarking assignment (pairs) | 5% | Friday 26 April 2013 |
| Iteration 1 (pairs) | 10% | Friday 10 May 2013 |

| | | |
|---|---|---|
| Iteration 2 (pairs) | 10% | Friday 24 May 2013 |
| Final report (individual) | 5% | Friday 31 May 2013 |
| Examination 1 | 50% | To be advised |

# Teaching Approach

- **Lecture and tutorials or problem classes**
  This teaching and learning approach provides facilitated learning, practical exploration and peer learning.
- **Laboratory-based classes**
  This learning and teaching approach provides the opportunity for practical experimentation with approaches taught in lectures.

# Assessment Requirements

## Assessment Policy

Faculty Policy - Unit Assessment Hurdles
(http://www.infotech.monash.edu.au/resources/staff/edgov/policies/assessment-examinations/unit-assessment-hu

Academic Integrity - Please see the Demystifying Citing and Referencing tutorial at
http://lib.monash.edu/tutorials/citing/

## Assessment Tasks

## Participation

Students are expected to attend lectures and lab classes.

While lab exercises will not be marked, many of the labs will be used for project work in pairs, including assessment. As such, students will need to attend these.

Students will identify a project partner to work with by Week 3, who is enrolled in the same lab class. Students will work with their partner for subsequent in-semester assessment unless there are exceptional circumstances.

- **Assessment task 1**

  **Title:**
  Introductory programming assignment
  **Description:**
  Students will complete an introductory programming assignment which introduces them to the general problem domain of future assignments.
  **Weighting:**
  5%
  **Criteria for assessment:**

  - Delivery of required functionality.
  - Quality of program design.
  - Quality of coding practices.
  - Layout and documentation.

  **Due date:**
  Friday 15 March 2013

- **Assessment task 2**

  **Title:**
  Vision statement (pairs)
  **Description:**
  Project pairs will write a brief, high-level description of the key requirements of the semester-long project which forms the basis for subsequent assignment tasks.
  **Weighting:**
  5%
  **Criteria for assessment:**

- Effectively capturing key functional requirements.
- Identification and description of key attributes of system context.
- Readability and presentation quality of document.

This assignment will be completed in pairs. Pairs are expected to share the workload and will usually receive the same mark. Individual contributions will be documented, and if workloads are not appropriately shared over the semester marks will be adjusted accordingly.

**Due date:**

Friday 29 March 2013

- **Assessment task 3**

**Title:**

Spiking exercises (pairs)

**Description:**

Project pairs will conduct several "spiking" exercises, conducting exploratory/prototype coding activities to better characterise and mitigate identified risks to the project.

Each individual spike will consist of some exploratory/proof-of-concept coding, and a report which documents the information gained.

Students will conduct demonstrations in labs.

**Weighting:**

10%

**Criteria for assessment:**

- Quality of report, including risk description, description of coding tasks, clear reporting of outcome, and assessment of risk after spiking.
- Effectiveness of coding undertaken to mitigate risk.

There will be multiple spikes. Some may be conducted by only one member of project pairs, some will be conducted by both members. Spiking reports will be assessed individually.

**Due date:**

Friday 12 April 2013

- **Assessment task 4**

**Title:**

Design and benchmarking assignment (pairs)

**Description:**

Project pairs will work together to write a document describing aspects of the architecture and design of their system. As well as structured text, the document will include UML diagrams.

To inform their design, pairs will conduct "benchmarking" to measure the performance of alternative designs, and report the results in written form.

**Weighting:**

5%

**Criteria for assessment:**

- Comprehensiveness of design.
- Quality of design decisions.

- Readability and layout of report.
- Rigorousness of benchmarking
- Correct use of UML notation to document design.

The assignment will be assessed as a pair. Students will document their individual contributions to the assignment. Generally, students will receive the same mark, unless there are significant discrepancies over the semester in contributions to the pair project work.

**Due date:**

Friday 26 April 2013

- **Assessment task 5**

**Title:**

Iteration 1 (pairs)

**Description:**

Project pairs will complete their first iteration of the active development phase of the project.

They will deliver working and tested software implementing several "user stories" as developed in earlier assignments.

The deliverables will also include release notes and user documentation as required.

**Weighting:**

10%

**Criteria for assessment:**

- Completeness of implementation.
- Quality of implementation.
- Appropriateness of testing.
- Correct use of provided tools.
- Appropriate supply of documentation.

The assignment will be assessed as a pair. Students will document their individual contributions to the assignment. Generally, students will receive the same mark, unless there are significant discrepancies over the semester in contributions to the pair project work.

**Due date:**

Friday 10 May 2013

- **Assessment task 6**

**Title:**

Iteration 2 (pairs)

**Description:**

Students will complete a second iteration of development for the system, implementing additional functionality. They will use "pair programming" for at least part of this development, and reflect on whether it is useful for them.

Another release will be delivered at the end of the iteration, which will include appropriate testing and documentation for the implemented functionality.

Students will demonstrate this functionality in labs.

**Weighting:**

10%

**Criteria for assessment:**

- Completeness of implementation.
- Quality of implementation.
- Appropriateness of testing.
- Correct use of provided tools.
- Appropriate supply of documentation.
- Conduct of pair programming.
- Quality of written reflection on pair programming.

The assignment will be assessed as a pair. Students will document their individual contributions to the assignment. Generally, students will receive the same mark, unless there are significant discrepancies over the semester in contributions to the pair project work.

**Due date:**

Friday 24 May 2013

- **Assessment task 7**

**Title:**

Final report (individual)

**Description:**

Students will reflect on their project work, including:

- Reflecting on the overall quality of the deliverables to date.
- Identifying the strengths and weaknesses of the development processes used in the context of the current project.
- Determine a realistic timeframe for completion of the full vision as described earlier in semester.
- Describe the process they would follow to achieve completion.

**Weighting:**

5%

**Criteria for assessment:**

- Quality and depth of reflection.
- Plausibility of analysis of future prospects.
- Quality of proposed process.
- Clarity of written report.

**Due date:**

Friday 31 May 2013

# Examinations

- **Examination 1**

**Weighting:**

50%

**Length:**

3 hours

**Type (open/closed book):**

Closed book

**Electronic devices allowed in the exam:**

Assessment Requirements

None

## Learning resources

## Reading list

- The Scrum Primer. Free download: http://scrumfoundation.com/libraryScrum
- Scrum and XP from the Trenches by Henrik Kniberg. Free download: http://www.infoq.com/minibooks/scrum-xp-from-the-trenches
- Cryptography notes by Graham Farr will be provided on Moodle.

Monash Library Unit Reading List
http://readinglists.lib.monash.edu/index.html

## Feedback to you

Types of feedback you can expect to receive in this unit are:

- Informal feedback on progress in labs/tutes
- Graded assignments with comments
- Solutions to tutes, labs and assignments

## Extensions and penalties

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.

## Returning assignments

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

## Resubmission of assignments

No resubmission of assignments will be permitted.

## Referencing requirements

Code based on algorithms or information from third-party sources (such as books or websites) must acknowledge these sources in comments, in sufficient detail for markers to find and check these sources. Some assignments will explicitly prohibit the use of this - if so, you must not use them.

Where third party material is used in written assignments, they should be cited. Students may use any of the referencing methods described in the Monash University Library's tutorial on referencing to do so:

http://guides.lib.monash.edu/content.php?pid=88267&sid=656564

## Assignment submission

It is a University requirement (http://www.policy.monash.edu/policy-bank/academic/education/conduct/plagiarism-procedures.html) for students to submit an assignment coversheet for each assessment item. Faculty Assignment coversheets can be found at http://www.infotech.monash.edu.au/resources/student/forms/. Please check with your Lecturer on the submission method for your assignment coversheet (e.g. attach a file to the online assignment submission, hand-in a hard copy, or use an online quiz).

## Online submission

If Electronic Submission has been approved for your unit, please submit your work via the learning system for this unit, which you can access via links in the my.monash portal.

## Recommended Resources

Students will be provided with a VMWare virtual machine image, which contains a full development environment for the lab and project work they will be expected to complete through the semester. Students can download VMWare Player for Linux, Mac OS X and Windows at no charge.

If they wish can also install a Java development environment, the Eclipse Java IDE, and the Android SDK at no cost on their own computer directly, but no support will be provided for this.

Students with Android smartphones may, if they wish, use these to test and debug the software developed throughout the unit.  This, however, is not necessary, as sufficient smartphones have been purchased for the purpose, and a limited number will be available for students to borrow overnight.

### Recommended text(s)

Burnette, Ed. (2008). *Hello, Android (3rd edition): Introducing Google's Mobile Development Platform*. (3rd Edition) No Starch Press (ISBN: 978-1-93435-656-2).

Kent Beck; Cynthia Andres; Erich Gamma. (). *Extreme Programming explained*. (2nd Edition) Addison-Wesley Professional (ISBN: 978-0321278654).

## Examination material or equipment

Students should refer to the unit website for details about the exam.

# Other Information

## Policies

Monash has educational policies, procedures and guidelines, which are designed to ensure that staff and students are aware of the University's academic standards, and to provide advice on how they might uphold them. You can find Monash's Education Policies at: www.policy.monash.edu.au/policy-bank/academic/education/index.html

Key educational policies include:

- Plagiarism; http://www.policy.monash.edu/policy-bank/academic/education/conduct/plagiarism-policy.html
- Assessment in Coursework Programs; http://www.policy.monash.edu/policy-bank/academic/education/assessment/assessment-in-coursework-po
- Special Consideration; http://www.policy.monash.edu/policy-bank/academic/education/assessment/special-consideration-policy.ht
- Grading Scale; http://www.policy.monash.edu/policy-bank/academic/education/assessment/grading-scale-policy.html
- Discipline: Student Policy; http://www.policy.monash.edu/policy-bank/academic/education/conduct/student-discipline-policy.html
- Academic Calendar and Semesters; http://www.monash.edu.au/students/dates/
- Orientation and Transition; http://intranet.monash.edu.au/infotech/resources/students/orientation/
- Academic and Administrative Complaints and Grievances Policy; http://www.policy.monash.edu/policy-bank/academic/education/management/complaints-grievance-policy.h
- Code of Practice for Teaching and Learning; http://www.policy.monash.edu.au/policy-bank/academic/education/conduct/suppdocs/code-of-practice-teac

### Graduate Attributes Policy

http://www.policy.monash.edu/policy-bank/academic/education/management/monash-graduate-attributes-policy.ht

## Student services

The University provides many different kinds of support services for you. Contact your tutor if you need advice and see the range of services available at http://www.monash.edu.au/students. For Sunway see http://www.monash.edu.my/Student-services, and for South Africa see http://www.monash.ac.za/current/.

## Monash University Library

The Monash University Library provides a range of services, resources and programs that enable you to save time and be more effective in your learning and research. Go to www.lib.monash.edu.au or the library tab in my.monash portal for more information. At Sunway, visit the Library and Learning Commons at http://www.lib.monash.edu.my/. At South Africa visit http://www.lib.monash.ac.za/.

## Disability Liaison Unit

Students who have a disability or medical condition are welcome to contact the Disability Liaison Unit to discuss academic support services. Disability Liaison Officers (DLOs) visit all Victorian campuses on a regular basis.

Website: http://www.monash.edu/equity-diversity/disability/index.htmlTelephone: 03 9905 5704 to book an appointment with a DLO; or contact the Student Advisor, Student Commuity Services at 03 55146018 at SunwayEmail: dlu@monash.eduDrop In: Equity and Diversity Centre, Level 1, Building 55, Clayton Campus, or Student Community Services Department, Level 2, Building 2, Monash University, Sunway Campus

## Your feedback to Us

Monash is committed to excellence in education and regularly seeks feedback from students, employers and staff. One of the key formal ways students have to provide feedback is through the Student Evaluation of Teaching and Units (SETU) survey. The University's student evaluation policy requires that every unit is evaluated each year. Students are strongly encouraged to complete the surveys. The feedback is anonymous and provides the Faculty with evidence of aspects that students are satisfied and areas for improvement.

For more information on Monash's educational strategy, see:

www.monash.edu.au/about/monash-directions and on student evaluations, see:
www.policy.monash.edu/policy-bank/academic/education/quality/student-evaluation-policy.html

## Previous Student Evaluations of this Unit

Project has been changed to provide more of a computer science emphasis.

Environment has been switched back to using a virtual machine to avoid the many hassles that were happening with the Windows native environment.

If you wish to view how previous students rated this unit, please go to
https://emuapps.monash.edu.au/unitevaluations/index.jsp