School of Electrical and Information Engineering
University of the Witwatersrand, Johannesburg
ELEN3009A — Software Development II

Course Brief and Outline — 2019

Academic Staff:

Dr S.P. Levitt (course co-ordinator)
Room: CM3.237
Tel: 011 717 7209
Email: stephen.levitt@wits.ac.za

1 Course Background

The course forms part of the "Software Development" line of courses. It follows on from Software Development I and is a pre-requisite for Software Development III. It deepens and broadens students' technical knowledge and skills with respect to software engineering and development.

2 Course Objectives

This course has three main objectives. It introduces the student to object-oriented modelling and design, programming using modern C++, and key software engineering technical practices such as unit testing and version control. More specialized software engineering courses flow from, and build on, the conceptual knowledge and practical skills developed in this course.

3 Course Outcomes

On successful completion of this course, the student is able to:

1. describe how C++ fits into the software development ecosystem and the trade-offs that it makes with respect to other programming languages;
2. perform object-oriented analyses of moderately complex software problems;
3. design and implement object-oriented solutions in C++ for such problems;
4. program in C++ in a modern, idiomatic fashion;
5. create a graphical user interface (GUI) using a game programming library;
6. utilise a test framework for software testing and verification;
7. use version control in a simple manner for tracking changes and sharing code;
8. use a formal notation to document a software design as well as an automated documentation tool.

4 Course Content

C++'s Place in the World Compiled versus interpreted languages; strongly-typed versus weakly-typed languages; language popularity; programming paradigms

The Development Environment The Integrated Development Environment (IDE); the build toolchain: preprocessor, compiler, linker; debugging

C++ Fundamentals Primitive types, pointers and references; parameter passing; const; static, smart pointers, auto, range-based for loops, tuples

The C++ Standard Template Library Containers; iterators; algorithms; vector and string

Programming Principles and Practices Readable code, DRY principle, defensive coding, separation of concerns

Object-Oriented Analysis and Design Levels of abstraction; interface and implementation; object-oriented analysis and design process; Tell, Don't Ask principle; Liskov substitution principle; code smells; modelling pitfalls

Object-Oriented Programming in C++ Classes and objects; constructors and destructors; copy construction and assignment; inheritance and polymorphism; aggregation.

Unit Testing Automated unit testing, unit test frameworks, test-driven development; writing good tests

Version Control Git and GitHub, commits, fast-forward merges, pull requests, collaboration

Software Documentation Unified Modelling Language (UML); class and sequence diagrams; technical documentation; extracting documentation from code

Error Handling Exceptions; assertions; pre- and post-conditions; alternative error handling schemes

Graphical User Interfaces Game development library; graphics; event handling

5 Prior Knowledge Assumed

This course depends on content covered in Software Development I. In particular it assumes that students have a working knowledge of programming fundamentals (in C++). This includes variables; scope; flow control; raw pointers; functions, standalone classes, and the ability to write small programs to solve problems.

The prerequisites and co-requisites to register for this course are defined in the current *Rules & Syllabuses: Faculty of Engineering and the Built Environment*.

6 Assessment

6.1 Formative Assessment

The assessment of laboratories is regarded as formative in that the emphasis is on student participation and learning rather than whether the solutions are correct or not. Feedback on laboratories is provided through comments on each group's source code submissions as well as discussions held during dedicated lab review sessions.

6.2 Summative Assessment

Assessment Contributor	Duration (hours)	Component	Method & Weight	Calculator Type	Permitted Supporting Material
Laboratories	33	No	Marks, 5%	–	–
Project	30	No	Marks, 35%	–	–
Test	1	No	Marks, 10%	1	Reference sheets provided
Exam	3	No	Marks, 50%	1	Reference sheets provided

6.3 Assessment Methods

Students will be assessed working alone (test and exam) and in groups (labs and project).

The project will require the student to work as part of a team to creatively identify, assess, formulate and solve a software development problem by using concepts, methods, tools and techniques introduced in this course. Additionally, the student must communicate the results critically and effectively by means of a set of group-effort reports using appropriate structure, style and graphical support. A marking rubric will accompany the project brief and this rubric will present the criteria for assessing the project outcomes.

The test is a written test which will take place during the semester. The material to be covered will be indicated by the lecturer.

The final exam will be time-tabled within the Oct/Nov examination period. All the material covered in the lectures, laboratories, readings, and the course project is examinable.

7 Satisfactory Performance (SP) Requirements

For the purpose of Rule G.13 *satisfactory performance in the work of the class* means attendance and completion of prescribed laboratory activities, attendance at tutorials designated as compulsory in this CB&O, submission of assignments, writing of scheduled tests unless excused in terms of due procedure.

8 Teaching and Learning Process

8.1 Teaching and Learning Approach

The basic material of the course is covered in lectures. From time to time, additional material will be handed out or referred to. Lectures should be seen as a forum for discussion of the course material. This requires lecturer-student interaction and participation of the student in lecture activities.

Lectures will be used to introduce and explain key concepts, but these will need to be reinforced through practical laboratory work. The Computer Laboratory, on the ground floor of the Chamber of Mines building, is available for students to use.

The software development tools used during this course are freely available enabling students to use their own computer facilities if they wish.

8.2 Information to Support the Course

Supporting information for this course can be found on the course website and in the handouts/articles provided to students.

There is no prescribed textbook for this course, although the textbook that is used in Software Development I covers some of the course material. Many good introductory texts on C++ programming are available in the Engineering and Geo/Maths libraries and at major bookstores. Additionally, there is an enormous amount of material that is accessible on the web. These resources can be used to supplement the course material.

8.3 Learning Activities and Arrangements

Lectures

There will be *three* 45-minutes lectures a week. Students need to take their own notes during lectures; however, all the lecture slides will be made available on the course home page. Students are expected to attend lectures and it will be taken that announcements made during lectures have been received by all students.

Laboratories

There are a number of laboratory exercises associated with this course. These will be carried out in the Computer Laboratory on certain afternoons as published in the third year timetable. *Doing the laboratories is indispensable for understanding the course material.* During the laboratory sessions demonstrators will be on hand to provide assistance. One of the requirements for earning the marks for the laboratories is that the student must complete laboratory exercises each week.

The School's rules governing the use of the Computer Laboratory must be obeyed.

Project

The project brief will handed out to the students during the course. The project will be carried out over a number of weeks. Students are required to work in groups of two for the project.

Consultation

Students should contact the lecturer either in person or via email in order to arrange a meeting. During the examination period consultation will be limited to fixed times. These times will be posted on the course homepage.

9 Course Home Page

Further information and announcements regarding the course are posted on the course home page: https://witseie.github.io/software-dev-2/

All students are expected to consult the course home page at regular intervals.