Software Engineering 2A04

Software Design I

Fall 2002

Course Outline

Revised: 10 September 2002

Note: This course outline contains important information that may affect your grade. You should retain it throughout the semester as you will be assumed to be familiar with the rules specified in this document.

Instructor

Dr. William M. Farmer
Office: ITB 163
Extension: 27039
E-mail: `wmfarmer@mcmaster.ca`
Web: `http://imps.mcmaster.ca/wmfarmer/`
Office hours: M 9:30–10:20, M 16:30–17:20, R 9:30–10:20.

Teaching Assistants

Huan Gao, Sara Kennedy, Yamama Khadduri, Xiaoyang Yu

Course Web Site

`http://www.cas.mcmaster.ca/~wmfarmer/SE-2A04-02/`

Lecture and Lab Schedule

Lectures:	MWR	10:30–11:20	HSC 1A4
Tutorial:	F	14:30–15:20	ABB 136
Lab:	F	15:30–17:20	ITB 235,236

Calendar Description

"Software development with precise specifications. Implementation, inspection, integration, and testing of precisely specified sequential modules and programs. Assembly of software from independent modules; incremental design."

Mission

"The mission of this course is to give students an understanding of the professional responsibilities of software engineers and the role of precise

specifications in software development. Students learn how to read precise specifications and how to use specifications in program design, program inspection, and program testing. They will be introduced to the basic principles of software design. This course teaches students how to read and use specifications for individual (terminating) programs, program packages or modules, and complete systems. Later courses teach about how to design interfaces and write specifications as well as about non-terminating and concurrent programs."

Introduction

The primary responsibility of all engineers is to design products that are fit for their intended use. An important step towards fulfilling this responsibility is collecting information about a product or component in the form of a specification, a description of the requirements that the product/component must meet. The specification must be precise enough that any product that satisfies it will be acceptable in use. On the other hand, the specification should not constrain the designer unnecessarily. It should allow a variety of implementations and allow the implementor to be creative.

As a Software Engineer, you will have two distinct responsibilities:

1. To make sure that a product's specification is complete and correct, i.e., that any product that satisfies the specification will be fit for its intended use.

2. To make sure that your software product satisfies its specification.

Today's software products cannot be monolithic; they must be composed of manageable components. One of the most important requirements of any component is that it be compatible with the components that it might replace or that might replace it. Integrating a new version of a component into an existing product, often called "upgrading", should be a trouble-free experience. To achieve this, a component's specification must describe its interfaces precisely. Thus, the above statements apply to both complete products and to components of products. Your product may be a component of a larger product.

This course is intended to give students an understanding of the role of precise specifications in software development. You must learn how to read precise specifications, how to review specifications, how to write programs that satisfy specifications and how to use specifications in program design, program inspection, and program testing. You will be introduced to the basic principles of software design and provided with opportunities to work with a variety of forms of specifications.

This course is part of a three course sequence on software design. Later courses teach how to design interfaces and write specifications as well as how to work with nonterminating and concurrent programs. Next term's course includes a large software project that must be done by teams, and will

require teams to exchange components. An understanding of specifications will be essential for the success of your project next term.

We expect your programming skills and your familiarity with our laboratory computer systems will grow during the course, but you should never forget that your main task in this course is to learn to:

1. Check if a specification is appropriate for the intended use.
2. Design programs that meet specification.
3. Test programs to see if they meet the specification.
4. Inspect programs to make sure that they meet the specification.

Logs

Keeping a detailed, up-to-date log is an essential part of Professional Engineering. A log must record all of the steps in producing your product. This means that you must record all sources of information, and summaries of all events including consultations with teachers and colleagues. The log should describe all errors that you discovered and the corrective actions that you took. You may need this record if you are accused of negligence, theft of intellectual property, or incompetence. The entries in the log book should be listed chronologically with dates and times.

A separate log is required for each lab exercise, and no modifications to it are allowed after the lab exercise's due date. *A copy of your log must be in a universally readable text file named* `log-lab-ex-`*n (where n is the lab exercise number) in the top level of your home directory.*

Lab Exercises

The course consists of both lectures, tutorial sessions, and laboratory sessions. There are 6 laboratory experiments designed to give you experience in applying what you learn in the lectures. Each will take two weeks. In the first week, you will receive a specification and begin work during the laboratory session. This will give you an opportunity to discuss it and ask questions of the instructor and teaching assistants. You will then have one week in which to complete the programming, develop a test plan based on the specification, and test your product. You are expected to bring your working product along with a test report and your log to the second week session.

In the second week session you will first demonstrate your program to one of the TAs and then exchange it with another student assigned to you. In the rest of the second week, you must apply your test program to the other student's program and prepare a second test report.

At the beginning of the tutorial session of the three week session, you are required to turn in your lab report for the lab exercise. (The requirements for the lab reports will be given later.)

You must do the laboratory work yourself or you will not learn what we want to teach. The subject matter of the lab exercises will be covered on the surprise quizzes, midterms, and final exam. You must not represent the work of others as your own. *A consultation that is not recorded in your log will be considered as academic dishonesty.*

5 of the 6 programs will be written in the Oberon-2 programming language and one will be written in C. All programs must execute properly on the computers in the ITB lab.

Texts

1. **Required**: F. P. Brooks, Jr., *The Mythical Man-Month*, Addison Wesley, 1995.
2. **Required**: H. Mössenböck, *Oberon-2*, McMaster Custom Courseware, SE 2A04, September 2002 (or H. Mossenbock, *Object-Oriented Programming in Oberon-2*, Springer-Verlag, 1995).
3. **Optional**: E. Nikitin, *Into the Realm of Oberon*, Springer-Verlag, 1998.
4. **Optional**: D. Hoffman and P. Strooper, *Software Design, Automated Testing Maintenance*, McMaster Custom Courseware, SE 2A04, September 2002.
5. **Optional**: D. M. Hoffman and D. M. Weiss, *Software Fundamentals: Collected Papers by David L. Parnas*, Addison Wesley, 2001.

Grading

Your course grade will be based on your performance on the lab exercises, surprise quizzes, midterm tests, and a final exam as follows:

Lab exercises (6)	20%
Surprise quizzes (2)	10%
Midterm tests (2)	30%
Final exam	40%
Total	**100%**

Notes:

1. A student's final score will be reduced by one half point for each missed class (there is no penalty for the first *four* missed classes).
2. If you are not present in class when a surprise quiz is given, you will receive a 0 (unless you have a valid medical excuse).

3. The two midterm tests will be 50 minutes long. They will be given during the normal class hour on September 26 and October 24 in ABB B163.

4. The final exam will be 3 hours long. It will take place on the date scheduled by the University.

5. At the end of the term, two scores will be computed for you: (1) the *total score* as described above and (2) the *partial score* of just the marks on your midterm tests and final exam. *If the partial score is below 50%, you will fail the course (and your course grade will be the minimum of the total and partial scores).* Otherwise, your grade is awarded on the basis of the total score.

6. The instructor reserves the right to adjust the grades for a lab exercise, surprise quiz, midterm test, or final exam by increasing or decreasing every score by a fixed percentage.

Policy Statements

1. The 2A04 team is eager to support you and help you to have a good learning experience. We would appreciate your suggestions on how we can improve our teaching methods.

2. Significant study and reading outside of class is required.

3. Regular class attendance is expected, and attendance will be taken. Faked attendance will be considered as academic dishonesty.

4. You are *strongly* urged to ask questions during class.

5. You may want to discuss the lab exercises with your fellow students. *If you do that, you must record a summary of your discussions in your log and your lab report must include a list of all those with whom you discussed the exercise and describe what information you received.* It is part of your professional responsibility to give credit to all who have contributed to your product.

6. *Your final program must be your own.* If it is discovered that you have not written your own program, or that you have consulted with people not mentioned in your lab report, it will be considered as academic dishonesty.

7. You may use your texts and notes during the surprise quizzes, midterm tests, and final exam.

8. Lab exercises may not be turned in late and midterm tests may not be taken later without *prior* approval from the instructor.

9. The instructor reserves the right to require a deferred final exam to be oral.

10. Any calculator can be used on tests and examinations.

11. "The Faculty of Engineering is concerned with ensuring an environment that is free of all adverse discrimination. If there is a problem, that cannot be resolved by discussion among the persons concerned, individuals are reminded that they should contact their Department Chair, the Sexual Harassment/Anti-Discrimination Officer (SHADO), as soon as possible."

12. "Students are reminded that they should read and comply with the Statement on Academic Ethics and the Senate Resolutions on Academic Dishonesty as found in the Senate Policy Statements distributed at registration and available in the Senate Office" (see Senate Secretariat, Gilmour Hall, Room 104, 905-525-9140 or 905-529-7070, ext. 24337).

Schedule

September 5	R	00. Overview
September 6	F	Introduction to Unix and Oberon
September 9	M	01 Fundamental Programming Concepts 1
September 11	W	01 Fundamental Programming Concepts 2
September 12	R	01 Fundamental Programming Concepts 3
September 13	F	Lab 1 started
September 16	M	01 Fundamental Programming Concepts 4
September 18	W	01 Fundamental Programming Concepts 5
September 19	R	02 Professional Responsibilities of Software Engineers
September 20	F	Lab 1 demonstrations
September 23	M	03 Software Specification and Description 1
September 25	W	03 Software Specification and Description 2
September 26	R	Midterm test 1
September 27	F	Lab 1 reports due; Lab 2 started
September 30	M	03 Software Specification and Description 3
October 2	W	03 Software Specification and Description 4
October 3	R	04 Software Modules 1
October 4	F	Lab 2 demonstrations
October 7	M	04 Software Modules 2
October 9	W	04 Software Modules 3
October 10	R	04 Software Modules 4
October 11	F	Lab 2 reports due; Lab 3 started
October 14	M	THANKSGIVING HOLIDAY
October 16	W	04 Software Modules 5
October 17	R	05 Verification and Analysis 1
October 18	F	Lab 3 demonstrations

October 21	M	05 Verification and Analysis 2
October 23	W	06 Software Development 1
October 24	R	Midterm Test 2
October 25	F	Lab 3 reports due; Lab 4 started
October 28	M	06 Software Development 2
October 30	W	06 Software Development 3
October 31	R	07 Software Structure 1
November 1	F	Lab 4 demonstrations
November 4	M	07 Software Structure 2
November 6	W	07 Software Structure 3
November 7	R	08 More Modules 1
November 8	F	Lab 4 reports due; Lab 5 started
November 11	M	08 More Modules 2
November 13	W	08 More Modules 3
November 14	R	08 More Modules 4
November 15	F	Lab 5 demonstrations
November 18	M	08 More Modules 5
November 20	W	09 Recursion 1
November 21	R	09 Recursion 2
November 22	F	Lab 5 reports due; Lab 6 started
November 25	M	09 Recursion 3
November 27	W	10 Project Management 1
November 28	R	10 Project Management 2
November 29	F	Lab 6 demonstrations
December 2	M	All Questions Answered!
December 6	F	Lab 6 reports due