Software Engineering 2A04

Software Design I

Fall 2001

Course Outline

Revised: 6 September 2001

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.

Instructors

Dr. William M. Farmer
Office: ITC 163
Extension: 27039
E-mail: `wmfarmer@mcmaster.ca`
Web: `http://imps.mcmaster.ca/wmfarmer/`
Office hours: M 14:30–16:20, W 9:30–10:20, R 13:30–15:20

Dr. David L. Parnas
Office: ITC 159A
Extension: 27353
E-mail: `parnas@mcmaster.ca`
Web: `http://www.crl.mcmaster.ca/SERG/parnas.homepg`
Office hours: To be announced.

Teaching Assistants

Xiaohui Jin, Yan Li, Aaron O'Reilly, Mark Pavlidis, Srdjan Rusovan.

Course Web Site

`http://www.cas.mcmaster.ca/~wmfarmer/SE-2A04-01/`

Lecture and Lab Schedule

Lectures:	MWR	10:30–11:20	HH 320
Tutorial:	F	14:30–17:20	BSB 304
Lab:	F	14:30–17:20	ITC 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 project, and no modifications to it are allowed after the project'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 project number) in the top level of your home directory.*

Lab Exercises

The course consists of both lectures and laboratory sections. 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 instructors 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 to the second week session. A copy of your log should be included in your report.

In the second week session you will demonstrate your program to another student and give that student a copy. In the rest of the second week, you must apply your test program to the other student's program and prepare a test report. You will be required to demonstrate the other student's program to one of the TAs when you turn in your report. A copy of your log should be included in your report.

You must do the laboratory work yourself or you will not learn what we want to teach. You must not represent the work of others as your own. *A consultation that is not recorded in your log can be treated as academic dishonesty.*

All programs will be written in the Oberon programming language and must execute properly on the lab computers.

Texts

1. **Required**: F. P. Brooks, Jr., *The Mythical Man-Month*, Addison Wesley, 1995.
2. **Optional**: E. Nikitin, *Into the Realm of Oberon*, Springer-Verlag, 1998.
3. **Optional**: D. M. Hoffman and D. M. Weiss, *Software Fundamentals: Collected Papers by David L. Parnas*, Addison Wesley, 2001.

The first and second texts are available at the McMaster bookstore. The third can be purchased through Dr. Parnas at a 33% discount.

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

Notes:

1. 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).
2. The two midterm tests will be 50 minutes long. They will be given in class on September 27 and October 25.
3. The final exam will be 3 hours long. It will take place on the date scheduled by the University.
4. 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.

5. The instructors reserve 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.
4. You are urged to ask questions during class.
5. You may want to discuss the assignments with your fellow students. *If you do that, you must record a summary of your discussions in your log and your report must include a list of all those with whom you discussed the assignment and describe what information you received.* It is part of your professional responsible to give credit to all who have contributed to your product.
6. *Your final program must be your own.* If we discover that you have not written your own program, or that you have consulted with people not mentioned in your report, it will be treated as a case of 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 instructors.
9. The instructors reserve 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, 525-9140 or 529-7070, ext. 24337).

Schedule

September 6	R	Professional software development (Parnas)
September 7	F	Lab 0: Introduction to Unix and Oberon
September 10	M	Fundamental programming concepts 1 (Farmer)
September 12	W	Fundamental programming concepts 2 (Farmer)
September 13	R	Program design by construction 1 (Parnas)
September 14	F	Lab 1
September 17	M	Program design by construction 2 (Parnas)
September 19	W	Program design by construction 3 (Parnas)
September 20	R	Program design by construction 4 (Farmer)
September 21	F	Lab 1 cont.
September 24	M	Program design by construction 5 (Parnas)
September 26	W	Program design by construction 6 (Parnas)
September 27	R	Midterm test 1
September 28	F	Lab 2
October 1	M	Program design by construction 7 (Farmer)
October 3	W	Program design by construction 8 (Farmer)
October 4	R	Program description and specification 1 (Farmer)
October 5	F	Lab 2 cont.
October 8	M	Program description and specification 2 (Parnas)
October 10	W	Program description and specification 3 (Farmer)
October 11	R	Program description and specification 4 (Farmer)
October 12	F	Lab 3
October 15	M	Testing (Farmer)
October 17	W	Programs and Modules 1 (Parnas)
October 18	R	Programs and Modules 2 (Farmer)
October 19	F	Lab 3 cont.
October 22	M	Programs and Modules 3 (Farmer)
October 24	W	Software pragmatics 1 (Farmer)
October 25	R	Midterm Test 2
October 26	F	Lab 4
October 29	M	Software pragmatics 2 (Farmer)
October 31	W	Modularization 1 (Parnas)
November 1	R	Modularization 2 (Parnas)
November 2	F	Lab 4 cont.
November 5	M	Module interfaces 1 (Parnas)
November 7	W	Module interfaces 2 (Parnas)
November 8	R	Module interfaces 3 (Farmer)
November 9	F	Lab 5
November 12	M	Module interfaces 4 (Farmer)
November 14	W	Software development models (Farmer)
November 15	R	Software hierarchies 1 (Parnas)
November 16	F	Lab 5 cont.
November 19	M	Software hierarchies 2 (Parnas)
November 21	W	Traces 1 (Parnas)
November 22	R	Traces 2 (Parnas)
November 23	F	Lab 6
November 26	M	Traces 3 (Parnas)
November 28	W	Software design products (Farmer)
November 29	R	Inspection (Parnas)
November 30	F	Lab 6 cont.
December 3	M	Review