

Software Testing, Quality Assurance and Maintenance

Winter 2009

Patrick Lam

Brief Overview

As you have no doubt discovered, software never works right from the start. A key technique for getting more acceptable software is testing. Organized testing can help identify problems in software systems, enabling developers to fix these problems. This course will introduce software testing techniques; while it's not my goal to produce testers, you should at least be conversant with up-to-date testing methodologies and techniques.

In this class, we will also discuss software maintenance. While we greatly (over?) emphasize design in engineering school, maintenance consumes a large fraction of today's software development resources.

General Information

Course Web Page: <http://patricklam.ca/stqam>

Lectures: SE465 MWF 11:30-12:20, RCH 305
ECE465/CS447/CS647 MWF 3:30-4:20, RCH 309

Instructor:

Prof. Patrick Lam
Office: DC2534
Office Hours: Thursday 2:00-3:00, or by appointment
Email: p.lam@ece.uwaterloo.ca
Phone: Use email instead!

Teaching Assistants:

Reza Asadollahi
Office: DC2542
Office Hours: Tuesdays 4:00-5:00
Email: rasadoll@uwaterloo.ca

Mehdi Amoui Kalareh
Office: DC2542
Office Hours: Thursdays 3:00-4:00
Email: mamouika@uwaterloo.ca

Sen Li
Office: DC2542
Office Hours: Wednesdays 1:00-2:00
Email: s351li@engmail.uwaterloo.ca

Mohammad Yazdandoost
Office: DC3727
Office Hours: Tuesdays 1:00-2:00
Email: m.yazdandoost@gmail.com

Course Description

Objectives

- You will be able to create test suites for reasonably-sized software systems.
- You will be exposed to tools for software maintenance and verification.
- You will gain experience with carrying out modifications to a large pre-existing software package.

Topics. While the formulation of the topics is a bit different than the standard formulation, I expect that we will cover all of the topics in the standard course description. We will follow the book by Ammann and Offutt for most of the class, but I will supplement the material in the book with some additional material.

- Coverage, subsumption and infeasibility
- Graph Coverage (includes path and dataflow testing, state-based testing, call-graph-based testing, path-based testing)
- Logic Coverage (includes decision tables)
- Input Space Partitioning
- Syntax-Based Testing
- Testing in Practice (including OO testing)
- Non-testing-based Software Quality Assurance (code reviews, pair programming, software verification)
- Software Maintenance
- Testing for Performance

Additional topics, as time permits:

- Concurrency (perils of and tools to deal with)
- Failure-obliviousness and exception handling

Reference Material

The textbook for the class will be:

Paul Ammann and Jeff Offutt. Introduction to Software Testing. Cambridge University Press, 2008.

I also strongly recommend the following book:

Andreas Zeller. Why Programs Fail: a Guide to Systematic Debugging. Morgan Kaufmann, 2005.

The Zeller book is quite practical and I expect that it will be useful to you in the future as well.

Evaluation

This course includes assignments, a midterm, a course project, and a final examination.

2 individual assignments	10% each
Course project (in groups)	20%
Midterm	10%
Final exam	50%

I intend to make the midterm and final exams open-book, open-notes (subject to finding suitable questions!). The first assignment will be out by January 16 and due February 6, while the second assignment will be out by February 13 and due March 6. The midterm is scheduled for February 27. The project will be due by the last day of class.