

COMP354

Software Engineering

Lecture 0

Course Overview and Introduction

Instructor

Greg Butler ER-603-53 gregb@cs Ph: 848-2424 ext. 3031

<http://www.cs.concordia.ca/~gregb>

Lectures: Tuesdays and Thursdays 11:45 – 13:00 FG-B070

Tutorials: Tuesdays 13:15 – 14:05 CL-227

Scheduled Labs: TBA H-929

Office Hours: Tuesdays 15:00 – 17:00; at lectures; and by appointment

Textbook

Roger Pressman, *Software Engineering: A Practitioner's Approach*, McGraw-Hill Education; ISBN: 0073655783.

Reference Books: *Software Engineering*, by Ian Sommerville, Addison-Wesley.

Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process, by Craig Larman, Prentice-Hall.

The Unified Modeling Language User Guide, by G. Booch, J. Rumbaugh, I. Jacobson, Addison-Wesley.

Communication

Web Page comp354

Group email

Evaluation

Quiz 1	20%
Quiz 2	20%
Project Phase 1	15%
Project Phase 2	15%
Project Phase 3	15%
Project — individual work	15%
<hr/> Total	<hr/> 100%

Course Components

Lectures: Tuesdays 11:45 – 13:00 FG-B070

Tutorials: Tuesdays 13:15 – 14:05 CL-227

Labs: H-929 at times to be determined (2 hours per week)

Project: Material Tracking System

Course Outline

Introduction to Course Introduction to SE

- What is SE: discipline, systematic, team, budget constraints, includes maintenance; process, products, tools, people, principles
- Lifecycle overview
- Managerial and Technical Perspectives
- Dimensions of SE projects: business context, size, novelty, type of application
- People Issues: Skills needed on a team; Communication, Organization; Phases of team formation; Meetings

Tying it all together

- Vision/alignment: getting every phase focussed on same priorities
- Traceability: navigating from document to document
- Visibility: concrete description of the process; concrete evidence of the status of project
- Quality control: verification and validation

Requirements and Analysis

- WHAT is required
- use cases, mini-use case/functionality, scenarios
- other constraints
- documents: SRD, user manual, test cases

Course Outline (continued)

Principles of SE

- Major obstacles: complexity, re-work
- Errors will occur: prevention: rigor, models, what-if; Review/Testing: catch errors as soon as possible
- Rigor and Formality
- Abstraction
- Separation of Concerns

Lifecycle Models

- What is a phase: project standards, input/info sources, output documents activities, quality/review, audience for output; entry/exit conditions; example on effectiveness of reviews
- Code-and-fix
- Waterfall: an ideal: document-driven
- Incremental, iterative
- Risk-driven

Design

- HOW to meet requirements
- Activities: iteration, brainstorming, tracing scenarios, issue-resolution
- Information hiding, cohesion and coupling
- architectural design: components, connectors, constraints, how to analyse; client/supplier, peer-to-peer, uses, is_a relationship; example architectures: layers, pipe-and-filter, interactive interface, continuous transformation
- design patterns: observer, mediator, facade; expression tree example
- documents: Design documents

Course Outline (continued)

Code and Test

- REALISATION of product
- verification and validation
- testing: top-down vs bottom-up: drivers and stubs; black box vs white box; coverage
- reviews, walkthroughs, inspections
- documents: test plan (unit, module, integration), test cases, test results acceptance test

Formal methods

- Pros and cons: mix informal descriptions with formal
- Z example: proofs
- Larch example: rewriting, inductive proofs

Quality

- factors-criteria-metrics
- common metrics: function points, McCabe, Halstead
- quality control: record metrics, defect rates from all projects

Plus material on diagrammatic notations:
class diagrams, sequence diagrams, state diagrams

Project: Material Tracking System

Emphasis is on experiencing a complete software lifecycle (not final product)

- connections/dependencies between phases
- feedback/change request, re-work
- working as a team
- standards, review and testing to ensure quality/consistency of documents and software

Average load approx 10 hours per week (but varies)

Groups of 9-12 students

- 3 teams: Team Specification, Team Design, Team Implement
- team responsibilities
- individual responsibilities

Group dynamics are an important part

- minimise conflicts by establishing common goals/workload at start
- be specific about task assignments/deadlines
- allow for mistakes and re-work in schedule
- assign tasks as early as possible, so individuals can schedule their other work

Keep a diary of project activities.

Individual mark for project is based on peer evaluation.

URGENT: Form your groups!!!