UNIVERSITY OF SASKATCHEWAN
Department of Computer Science
176 Thorvaldson Building
110 Science Place Saskatoon, SK S7N 5C9 Canada
Telephone: (306) 966-4886 Facsimile: (306) 966-4884

COURSE SYLLABUS
CMPT 470/816: ADVANCED SOFTWARE ENGINEERING

Catalogue Description:

Covers advanced software engineering principles and techniques. Includes: software architecture; software evolution; reverse engineering; design recovery; refactoring; software comprehension; software analysis; domain specific techniques; requirements and specification; advanced design and modeling techniques; formal methods; and the business of software.

Prerequisite(s):	CMPT 370
Class Time & Location:	Tuesdays & Thursdays at 10:00-11:20 AM in Thorv 205A
Website:	Moodle and Dropbox

Instructor Information

Instructor:	Dr. Chanchal Roy
Contact:	Email**:** chanchal.roy@usask.ca
	Office Phone: 966-4163
Office Hours:	Location**:** Thorv 205A
	Hours**:** By appointment and after lecture.

Course Objectives and Course Contents

Covers advanced software engineering principles and techniques. Includes: software architecture; software evolution; reverse engineering; design recovery; refactoring; software comprehension; software analysis; domain specific techniques; requirements and specification; advanced design and modeling techniques; formal methods; and the business of software

This course builds on the understanding of software engineering presented in CMPT 370 and (to a lesser degree) CMPT 371. The focus is on techniques to help foster quality software engineering. The major topics covered will range from introductory but latest software engineering concepts to advanced software quality assurance and maintenance principles and techniques. The major topics are:

1. Source Transformation for Advanced Software Engineering

The students will learn the importance of source transformation systems and their applications in software engineering. In particular, the students will learn the TXL source transformation language and then will apply TXL in several advanced software engineering activities such as source to source transformation (e.g., C language system to Java language systems or vice versa), pretty-printing (standard formatting of one's code), improving the quality of the software systems including the performance, support for better software testing, software maintenance activities including clone detection, analysis and refactoring. The students will also learn the associated state of the art tools.

2. Design Patterns

Design patterns are one of the important aspect of quality software and thus for quality software development. The course will extensively talked about all those advanced design patterns from Gang of Four Design Patterns. In particular, the course will discuss the following design patterns: Singleton, Adapter, Facade, Observer, Builder, Bridge, Chain of responsibility, Factory, Abstract Factory, Interpreter, State, Strategy, Prototype, Mediator, Memento, Command, Flyweight, Visitor, Proxy, Composite and Decorator design patterns.

3. Development Anti-Patterns

As of the Design Patterns, Anti-Patterns, in particular, development Anti-Patterns are equally important for quality software development. In this course, we will also discuss in details a few of the development anti-patterns such as Spaghetti Code, Mushroom Management, Stovepipe Enterprise, Cover Your Assets, Vendor Lock-In, Design By Committee, Warm Bodies, Jumble, Wolf Ticket, Swiss Army Knife and so on.

4. Software Quality Assurance

Software quality assurance is the primary theme of this course and the students will learn the details of software quality, what is it, why do we need it, how to measure it, what are the different methods of providing quality assurance. The students will learn the details of different testing methods including black box, white box, grey box and mutation testing methods.

5. Code Smells and Refactoring

Given that software quality is the primary goal of this course, the students will also learn the details of different code smells including the number one code smell, the software clones. And then they will learn the different refactoring methods of how to remove those code smells including code clones.

6. Software change

This part of the lecture deals with software change, which is a foundation of most of the software engineering processes. We will first explain the classification of changes, followed by the explanation of the requirements, their analysis, and the product backlog. Software change consists of several phases, and the first one is the selection of a specific change request from product backlog. This is followed by the phase of concept location where the programmers find what specific software module needs to be changed. Then impact analysis decides on the strategy and extent of the change. Actualization implements the new functionality and incorporates it into the old code. Refactoring reorganizes software so that the change is easier (prefactoring), or cleans up the mess that the change may have created (postfactoring). Verification validates the correctness of the change. Change conclusion creates a new baseline, prepares software for the next change, and possibly releases the latest version to the users. This part of the lecture establishes a foundation on which the next part, software processes, builds.

7. Software Process

This part of the lecture presents the most common software processes. It explains what a software process is and what the forms are (model, enactment, performance, and plan). Then it deals with the iterative process of a solitary programmer (SIP), the team agile iterative process (AIP), directed iterative process (DIP), and centralized iterative process (CIP); these processes are primarily applicable to the stages of software evolution, servicing, but they also apply also to initial implementation and reengineering. This part then covers initial development and the final stages of software lifespan and hence it presents an overview of processes applicable to all stages of software lifespan.

8. Advanced Software Engineering Topics

In addition to the above topics, this course will also discuss on some other advanced software engineering topics as follows:

a. Subtyping, Subclassing and Liskov Substitute principle, Open Close principle

b. The Map/Reduce Framework

c. Garbage Collection

d. Multithreading in Java

e. Architectural Styles

f. Jini

g. Dynamic Proxies In Java

h. Remote Method Invocation (Java RMI)

i. Actor Model

j. Java IDL

k. Evolutionary Coupling

Student Evaluation

Grading Scheme

Participation (grad + urd)	3%
Assignments (grad + urd)	32%
Midterm Exam (grad + urd)	15%
Final Exam (urd)	50%
Project (grad)	50%
Total	**100%**

Assignments and Due Date:

Assignment 1: Change requests for Software Maintenance (group) [6 marks]

The students will be given a set of change requests. Based on the requests the students are expected to do concept location, impact analysis, prefactoring/refactoring, and actual implementation with due consideration of verification. Tentative due date: Feb 1, 2015

Assignment 2: Research paper presentations with critical analysis (group) [7 marks]

The students read a research paper in the area of the advanced software engineering, make a critical review, and present the paper in the class. Tentative due date: Feb 13, 2015

Assignment 3: Tutorial on Advanced Topics of Software Engineering (individual) [7 marks]

The students choose an advanced topic in the area of advanced software engineering and make a tutorial report. Based on the report the students make slides and present the tutorial in the class. Tentative due date: March 1,

2015

Assignment 4: Making Pretty Printer for C programs using TXL (group) [8 marks]

The students use the source transformation language TXL and then make a grammar for toy C language for parsing and facilitate source transformation and analysis. Tentative due date: March 14, 2015

Assignment 5: C to Pascal (Or Pascal to C) Programs Translator using TXL (group) [5 marks]

Using the TXL grammars for C and Pascal languages, the students build source transformation systems that will translate C programs to Pascal or Pascal programs to C. Tentative due date: March 25, 2015

Mid-terms

The Mid-Term for the class is scheduled for February 26, 2015 in the lecture.

Criteria that must be met to Pass

-The students should obtain pass marks both on the assignments and exams in order to pass the course.

Attendance Expectation

Attendance is not mandatory. However, there is up to 5% marks (including bonus) on class participation and discussion.

Final Exam Scheduling:

The Registrar schedules all final examinations, including deferred and supplemental examinations. Students are advised not to make travel arrangements for the exam period until the official exam schedule has been posted.

Note: All students must be properly registered in order to attend lectures and receive credit for this course.

Textbook Information

Required Text

Readings from a variety of sources will be provided as PDFs on the course website. Also the following texts:

J.R. Cordy, "Excerpts from the TXL Cookbook", Generative and Transformational Techniques in Software Engineering, Lecture Notes in Computer Science 6491, January 2011, pp. 27-91.

Vaclav Rajlich “Software Engineering: The Contemporary Practice”, CRC Press, Nov. 2011, 315 pages.

Lecture Schedule and Course Overview

Week of	Topics
Week 1 Monday, January 5, 2015	Introduction to Software Engineering and Software Change
Week 2 Monday, January 12, 2015	Software Change –Concept location, Impact Analysis, PreRefactoring
Week 3 Monday, January 19, 2015	Refactoring, Actualization
Week 4 Monday, January 26, 2015	Software Verification + Software Quality + Student Presentations
Week 5 Monday, February 2, 2015	Change-based Software Processes + Student Presentations
Week 6 Monday, February 9, 2015	Case studies on software change process + Student Presentations
Week 7 Monday, February 16, 2015	Family Day + Winter-Mid-Term Break (No Classes)
Week 8 Monday, February 23, 2015	TXL source transformations and analysis + Student Presentations
Week 9 Monday, March 2, 2015	TXL source transformations and analysis + Student Presentations
Week 10 Monday, March 9, 2015	Design Patterns + Advanced Software Engineering Topics + Student Presentations
Week 11 Monday, March 16, 2015	Design Anti-Patterns + Advanced Software Engineering Topics + Student Presentations
Week 12 Monday, March 23, 2015	Formal methods + Student Presentations
Week 13 Monday, March 30, 2015	Domain specific techniques + Student Presentations
Week 14 Monday, April 6, 2015	Final Exam Overview + Student Presentations

Policies

Recording of Lectures

I usually record the lectures. Students are also welcome to record the lectures if they wish.

Late Assignments

For any late assignments please contact the instructor. Depending on the reasons we may reach an agreement without any penalties.

Missed Assignments

Please talk to the instructor. There might be scope for alternative assignments.

Missed Examinations

1. Students who miss an exam should contact the instructor as soon as possible. If it is known in advance that an exam will be missed, the instructor should be contacted before the exam.

2. "A student who is absent from a final examination due to medical, compassionate, or other valid reasons, may apply to the College of Arts and Science Undergraduate Student's Office for a **deferred** exam. **Application must be made within three business days of the missed examination** and be accompanied by supporting documents." (http://artsandscience.usask.ca/students/help/success.php)

Incomplete Course Work and Final Grades

"*When a student has not completed the required course work, which includes any assignment or examination including the final examination, by the time of submission of the final grades, they may be granted an extension to permit completion of an assignment, or granted a deferred examination in the case of absence from a final examination.*

Extensions past the final examination date for the completion of assignments must be approved by the Department Head, or Dean in non-departmentalized Colleges, and may exceed thirty days only in unusual circumstances. The student must apply to the instructor for such an extension and furnish satisfactory reasons for the deficiency. Deferred final examinations are granted as per College policy.

In the interim, the instructor will submit a computed percentile grade for the class which factors in the incomplete coursework as a zero, along with a grade comment of INF (Incomplete Failure) if a failing grade.

In the case where the student has a passing percentile grade but the instructor has indicated in the course outline that failure to complete the required coursework will result in failure in the course, a final grade of 49% will be submitted along with a grade comment of INF (Incomplete Failure).

If an extension is granted and the required assignment is submitted within the allotted time, or if a deferred examination is granted and written in the case of absence from the final examination, the instructor will submit a revised assigned final percentage grade. The grade change will replace the previous grade and any grade comment of INF (Incomplete Failure) will be removed.

A student can pass a course on the basis of work completed in the course provided that any incomplete course work has not been deemed mandatory by the instructor in the course outline and/or by College regulations for achieving a passing grade." (http://policies.usask.ca/policies/academic-affairs/academic-courses.php)

For policies governing examinations and grading, students are referred to the Assessment of Students section of the University policy "Academic courses: class delivery, examinations, and assessment of student learning" (http://policies.usask.ca/policies/academic-affairs/academic-courses.php)

Academic Honesty

The University of Saskatchewan is committed to the highest standards of academic integrity and honesty. Students are expected to be familiar with these standards regarding academic honesty and to uphold the policies of the University in this respect. Students are particularly urged to familiarize themselves with the provisions of the Student Conduct & Appeals section of the University Secretary Website and avoid any behavior that could potentially result in suspicions of cheating, plagiarism, misrepresentation of facts and/or participation in an offence. Academic dishonesty is a serious offence and can result in suspension or expulsion from the University.

All students should read and be familiar with the Regulations on Academic Student Misconduct (http://www.usask.ca/secretariat/student-conduct-appeals/StudentAcademicMisconduct.pdf) as well as the Standard of Student Conduct in Non-Academic Matters and Procedures for Resolution of Complaints and Appeals (http://www.usask.ca/secretariat/student-conduct-appeals/StudentNon-AcademicMisconduct.pdf) Academic honesty is also defined and described in the Department of Computer Science Statement on Academic Honesty (http://www.cs.usask.ca/undergrad/honesty.php).

For more information on what academic integrity means for students see the Student Conduct & Appeals section of the University Secretary Website at:
http://www.usask.ca/secretariat/student-conduct-appeals/forms/IntegrityDefined.pdf

Examinations with Disability Services for Students (DSS)

Students who have disabilities (learning, medical, physical, or mental health) are strongly encouraged to register with Disability Services for Students (DSS) if they have not already done so. Students who suspect they may have disabilities should contact DSS for advice and referrals. In order to access DSS programs and supports, students must follow DSS policy and procedures. For more information, check http://students.usask.ca/health/centres/disability-services-for-students.php, or contact DSS at 966-7273 or dss@usask.ca.

Students registered with DSS may request alternative arrangements for mid-term and final examinations. Students must arrange such accommodations through DSS by the stated deadlines. Instructors shall provide the examinations for students who are being accommodated by the deadlines established by DSS.