Welcome to CS 476

Software Development Project

Course Outline

Dr. Samira Sadaoui

sadaouis@uregina.ca

Course Description

This course introduces modern techniques and tools involved in the software development with a focus on architecture styles and design patterns. Software architecture is crucial for the success of the overall development process. In this course, students will gain knowledge and experience in designing and implementing real-world software systems (more than trivial information systems). More precisely, students will employ use cases for the requirements specification phase, architectural styles, O.O. design patterns and UML for the design phase, and component frameworks for the implementation phase.

The goal of this project is the direct experience with large-scale software projects in which the students play the roles of the customer and software engineer. Students will develop high-quality application software that are Web-based and distributed. They will carry out their projects from the problem statement to the software testing. Students will analyze, design and implement software systems by following the phases of the software development life cycle: problem definition, feasibility study, requirements elicitation and specification, architectural design, detailed design, code construction and quality testing.

Required Textbook

Software Engineering: A Practitioner's Approach by Roger Pressman, Mc Graw Hill.

Supplemental References

- *Software Design: From Programming to Architecture* by Eric Braude, John Wiley & Sons, INC.
- *Design Patterns: Elements of Reusable Object-Oriented Software* by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides.

Office Hours

M/W/F from 2:30 to 3:30pm or by appointment.

Grading Scheme

Semester-long project	70%
Project presentation and software demonstration	30%

Major Topics (Lecture notes posted on UR Courses)

Software System Development	Feasibility study; Requirements analysis and specification; Top-level design/architecture; Detailed design; Implementation; Module and integration testing; Object-oriented analysis and design.
Software Architectural Styles	Software architecture definition and benefits; Several architectural styles: pipe and filter, layered, MVC, data-centered, peer to peer, event-based, service-oriented, client-server, N tiers.
O.O. Design Patterns	Design patterns definition and benefits; Structural, behavioral and creational patterns; Design pattern description template; Facade, Observer and Factory Method patterns; Integration of design patterns in software architectures; MVC and Observer.
Frameworks and Components	Framework types; Component definition; Component frameworks; COM/DCOM/COM+; .NET; EJB/J2EE; CORBA.

Detailed Project Requirements and Guidelines (posted on UR Courses)

Policies

1. Announcements and other relevant information will be posted on UR Courses.
2. Lecture notes and detailed project requirements will be posted on UR Courses.

3. LATE projects will NOT be accepted.

4. Strong programming skills are recommended for this course.

5. YOU SHOULD NOT read, copy or share other students' work. Please consult the section of the university calendar on Academic Misconduct and Penalties: Section 5.14.2.1. Academic Integrity, Section 5.14.2.2. Violations - Acts of Academic Misconduct, Section 5.14.4.3. Penalties: Academic Misconduct.

6. E-mail questions can be directed to your professor at **sadaouis@uregina.ca** or through UR Courses. You should always use your UR or CS account. If you send messages from other e-mail addresses, they will be ignored. Include CS476 at the beginning of the subject e-mail.

7. **Lecture attendance is mandatory!** Little time will be available to assist those who have missed many lectures.