

ECE 330

Software Design

Overview

Course Catalog Description: Design of software systems using modern modeling techniques. Relationship between software design and process, with emphasis on UML and its interface application code. Exposure to design patterns, software frameworks, and software architectural paradigms.

Prerequisites: ECE 231

Textbook:

- E. Braude, Software Design: From Programming to Architecture, Wiley, 2004.
- M. Page-Jones, Fundamentals of Object-oriented Design in UML, Addison-Wesley, 2000. (optional)

Class Goals: To provide an in-depth treatment of software design using modern tools and techniques. The course begins by setting software design within the context of the software lifecycle, and also provides a description of how it fits within a typical software process. Students will learn how to start from requirements, and create a software design that satisfies these requirements, along with a software model that captures the design. Students will make extensive use of modern software design tools; in particular the Unified Modeling Language (UML). Low-level (code-level) design will be considered, with particularly attention to issues such as robustness and flexibility in implementation. Mid-level design issues based upon software design patterns and components will then be introduced, with an emphasis on recognizing how these design patterns can be used to satisfy specific types of requirements. Finally, high-level design principles will be applied to the student of application frameworks and software architectures.

Course Coordinator: Prof. Gregory L. Heileman

Table I: Objectives and Implementation

Objectives		Implementation	A	B	C	D	E	F	G	H	I	J	K
O ₁	Background, Review, Software Requirements Software Lifecycles, and Software Processes.	Lectures	X		X		X					X	
O ₂	Object-oriented Programming and Software Modeling using the Unified Modeling Language (UML).	Lectures, homework, and software projects	X				X						
O ₃	Principles of Software Design	Lectures, homework, and software projects	X		X		X						
O ₄	Software Design Patterns	Lectures, homework, and software projects	X		X		X						
O ₅	Software Components	Lectures and programming assignments	X		X		X						
O ₆	Software Architectures and Application Frameworks	Lectures and programming assignments and robotics project	X		X		X						

Sample Course Schedule

Introduction (1.5 weeks)

- Goals of software design
- Programming review
- Software lifecycle
- Software processes

Object orientation – formal perspective (2 weeks)

- Classes and objects
- Inheritance and polymorphism
- Associations and aggregations

Software modeling – The Unified Modeling Language (UML) (3 weeks)

- Use cases
- Sequence and collaboration diagrams
- Architecture diagrams
- Round-trip engineering

Software Design Principles (2 week)

- Program correctness
- Refactoring
- Robustness
- Flexibility
- Reusability
- Efficiency

Design Patterns (2 weeks)

- Creational
- Structural
- Behavioral

Software Components (2 weeks)

- Java Beans
- Microsoft .NET Framework

Architectures and Frameworks (2 weeks)

- Three-tier architectures
- Web architectures
- Software security