Course name	**ECE 37200 Principles of Software Design**
Credit and contact hours	(3 cr.) 3
Course coordinator's name	Stanley Chien
Textbook	Software Engineering 10th Edition, Author: Ian Sommerville, ISBN # 978-0-13-394303-0
Course information	This course is designed to teach students best practices in designing and implementing object-oriented systems of high quality. To accomplish this task, we start with an overview of software design patterns and their role in developing high-quality software. We then begin surveying different design-level software design patterns, such as the Bridge, Strategy, Wrapper Facade, and Visitor software design patterns. Next, we touch on software design patterns for building distributed systems. Finally we finish the course by surveying Software anti-patterns, which are common design mistakes that negatively impact system quality, such as degrading performance as the system scales in size and complexity. Students will have the opportunity to apply learned techniques on several programming projects throughtout the semester. **Prerequisites/ Co-Requisite** P: CSCI 24000 R: CSCI 36200 (recommended) **Required, Elective, or Selected Elective:** EE Elective, CE Elective
Goals for the course	Upon successful completion of the course, students should be able to 1. Understand and define the purpose of software design patterns [1], 2. Identify and describe the design goal and intent of different design patterns [1] 3. Knowing what software design pattern to apply to a particular problem [6] 4. Apply techniques for implementing and debugging object-oriented software systems [6] 5. Apply techniques for identifying design flaws that negatively impact system quality [2,6]
List of topics to be covered	1. Principles of Rotten Design: Fragility, Rigidity, Portability, 2. Viscosity of Design and Environment 3. Design Principles: Object-Oriented Class Design (including 4. Open-Close Principle, Liskov Substitution Principle, 5. Dependency Inversion Principle, Interface Segregation 6. Principle), Principles of Package Design (including Release

	7. Reuse Equivalency Principle, Common Closure Principle, 8. Acyclic Dependencies Principle, Stable Dependencies 9. Principle, Stable Abstractions Principle), Composition 10. (including Inheritance and Aggregation) 11. Software Design Patterns: Design-Level Software Patterns e.g., GoF, Architecture-Level Software Design Patterns, e.g., POSA, Distributed System Design Patterns Anti-Patterns: Software Anti-Patterns, Software Performance Anti-Patterns 12. Clean Code: Refactoring, Code Smells & Heuristics
Syllabi approved by	Stanley Chien
Date of approval	12/16/2021