Computer Science 4306 Software Engineering

Student Learning Outcomes

1. Students will demonstrate knowledge of the distinction between critical and non-critical systems.
2. Students will demonstrate the ability to manage a project including planning, scheduling and risk assessment/management.
3. Students will author a software requirements document.
4. Students will demonstrate an understanding of the proper contents of a software requirements document.
5. Students will author a formal specification for a software system.
6. Students will demonstrate an understanding of distributed system architectures and application architectures.
7. Students will demonstrate an understanding of the differences between real-time and non-real time systems.
8. Students will demonstrate proficiency in rapid software development techniques.
9. Students will be able to identify specific components of a software design that can be targeted for reuse.
10. Students will demonstrate proficiency in software development cost estimation.
11. Students will author a software testing plan.

Course Content

Textbook: *Software Engineering*, Eighth Edition, by Ian Sommerville. Some or all of the following chapters are covered (See textbook "Contents").

1. **Socio-technical systems**. Emergent system properties, systems engineering, organizations, people and computer systems, legacy systems.
2. **Critical systems.** System dependability, availability and reliability, safety, security.
3. **Software processes.** Process models, process iteration, process activities, computer-aided engineering.
4. **Project management.** Project planning, scheduling, risk management.
5. **Software requirements.** Functional and non-functional requirements, user requirements, system requirements, interface specification, documentation.
6. **Requirements engineering processes.** Feasibility studies, requirements elicitation and analysis, validation, management.
7. **System models.** Context models, behavioral models, data models, object models, structured models.
8. **Critical systems specification.** Risk-driven specification, safety specification, security specification, software reliability specification.
9. **Formal specification.** Formal specification in the software process, sub-system interface specification, behavioral specification.

10. **Architectural design.** Architectural design decisions, system organization, modular decomposition styles, control styles, reference architectures.
11. **Distributed systems architectures.** Multiprocessor architectures, client-server architectures, distributed object architectures, inter-organizational distributed computing.
12. **Application architectures.** Data processing systems, transaction processing systems, event processing systems, language processing systems.
13. **Object-oriented design**. Objects and classes, object-oriented design process, design evolution.
14. **Real-time software design**. System design, real-time operating systems, monitoring and control systems, data acquisition systems.
15. **Rapid software development**. Agile methods, extreme programming, rapid application development, prototyping.
16. **Software reuse**. Design patterns, generator-based reuse, application frameworks.
17. **Component-based software engineering**. Components and component models.
18. **Critical systems development**. Dependable processes, dependable programming, fault tolerance, architectures.
19. **Software evolution**. Software maintenance, evolution processes, legacy system evolution.
20. **Verification and validation.** Planning, inspections, automated analysis, verification and formal methods.
21. **Software testing.** System testing, component testing, test case design, test automation.
22. **Software cost estimation.** Software productivity, estimation techniques, algorithmic cost modeling, project duration and staffing.
23. **Quality management.** Quality assurance and standards, planning and control, software measurement and metrics.
24. **Configuration management.** Configuration management planning, changes, version and release management, CASE tools.