CISC 3171 [30.1] Introduction to Software Engineering
3 hours; 3 credits

A broad view of software engineering that introduces a variety of software engineering techniques that can be applied to practical software projects. Topics include: process models, software specification, software design, software development methods and tools, verification and validation, reliability, and human factors.

Syllabus

1. Software Engineering: A preview. (1 week)
   History of SE, role of the software engineer, the software life cycle, relationship of SE to programming languages, operating systems, data bases, artificial intelligence, and theory of computer science.

2. Software: It's Nature and Qualities (1 week)
   Classification of software qualities, quality requirements in different application areas, measurement of quality.

3. Software Engineering Principles (2 weeks)
   Rigor and formality, separation of concerns, modularity, abstraction, anticipation of change, generality, incrementability.

4. Software Design (1 week)
   Software design activity and objectives, modularization techniques, object-oriented design, handling anomalies, a design case study, concurrent software.

5. The Software Production Process (2 weeks)
   Software production process models, waterfall model, evolutionary model, transformation model, spiral model, model assessment, case studies, organizing the process.

6. Software Specification (1 week)
   Specification usage and qualities, specification styles, specification verification, operation vs. descriptive specifications, specification in practice.

7. Software Verification (1 week)
   Goals and requirements, approaches, testing, analysis, symbolic execution, debugging, verifying other software properties.

8. Management of Software Engineering (1 week)
   Management functions, project planning, project control, organization, risk management.

9. Software Engineering Tools and Environments (2 weeks)
   Historical evolution, classification, representative tools, role of programming
languages, sample tools and environments, future scenario.

10. Testing Programs and Systems (2 weeks)
software faults and failures, testing issues, unit testing, integration testing, performance
testing, reliability, fault tolerance, testing safety-critical systems, real-time examples.

Primary Text:


Other References


