**CSC 2113 Software Engineering (4 CU)**

**Description :** This course introduces students to the foundations of software engineering as a discipline. Students are introduced to the evolving role of software engineering, especially with emphasis on software engineering process and process models.

Key topics covered include Software configuration management, Requirement analysis, Software Specification, Design methods, Software testing, Software project management techniques; Software project planning, Risk management; Software Quality Assurance; Software reuse; and Computer aided software engineering: CASE tools and application.

**Aims and Objectives**

• To introduce software engineering and to explain its importance

• To set out the answers to key questions about software engineering

• To introduce ethical and professional issues and to explain why they are of concern to software engineers

**Learning outcomes** : On successfully completing of this unit students will be able to

• Demonstrate competence in handling software engineering projects.

• Know the fundamental software engineering processes and models

• Know what is involved in a typical software engineering project’s life cycle.

• Employ good project management principles in handling projects and know why these principles are important in constructing quality software.

• Be competent in using CASE tools in real world projects.

**Teaching and Learning Pattern :** Teaching and learning is to be implemented through lecture, lab and tutorial sessions. Students are also expected to make presentations of their work.

**Indicative Content**

• Evolving role of software, software characteristics; Systems and environment; system engineering hierarchy, information and knowledge engineering; Information strategy; Business Area analysis, modeling enterprise and business-level data modeling, system architecture and associated information flow; writing system specification.

• Software Engineering as a layered technology: Software process, software process models. Software configuration management: the SCM process, Identification of objects in software configuration, version control, change control, configuration audit, SCM standards.

• Requirement analysis: Communication techniques, Information gathering tools; organizing and structuring information; analysis principles; Analysis modeling.

• Software Specification: Design process, principles and concepts: Abstraction, refinement, modularity, control hierarchy, structural partitioning, information hiding, functional independence, cohesion, coupling, design heuristics;

• Design methods: data design, architectural design, transform mapping, design optimization, human computer interface design, procedural design and tools; Design documentation.

• Software testing: Testing objectives, Testing principles, Testability, test case designing, white box testing; Basis path testing: Condition testing, data flow testing, loop testing; Black box testing: graph based testing methods, equivalence partitioning, Boundary value analysis, comparison testing; Testing documentation and help facilities; Software testing strategy: unit testing, integration testing, validation testing, system testing.

• Software project management techniques: project metrics, software measurement and metrics, software quality metrics;

• Software project planning: objectives of planning, resources, project estimation and estimation models, project decomposition techniques, make-buy decisions; automated estimation tools.

• Risk management: software risks, risk identification, risk projection, risk mitigation, monitoring and management; Project Scheduling: people and effort relationships, defining tasks, defining task network, scheduling techniques; Software teams and intra-team relationships; role of project manager.

• Software Quality Assurance: Concept of quality, quality control vs. quality assurance, cost of quality, factors that affect quality, quantitative view of quality, quality metrics, defect removal efficiency SQA activities, ISO standards and CMM practices, SEI levels, Software reviews, Formal approaches to SQA, Statistical Quality Assurance. Software reliability, reliability metrics, reliability models, meeting reliability requirements.

• Effective metrics for software process: Measurement principles, attributes of software metrics, metrics for analysis model, metrics for design model, metrics for source code, metrics for maintenance.

• Software reuse: difficulties in reuse, hardware reuse vs. software reuse, reusable artifacts, domain engineering approach, analysis design and construction of reusable components, classification and retrieval of components, economic impact of reuse and reuse metrics.

• Computer aided software engineering: CASE tools and application.

**Assessment method**

• Continuous assessment through practical exercises and Coursework, together with two scheduled tests (40%)

• Final exam at the end of the Semester and accounts for 60% of the final grade.

**Reading lists**

i. J.F. Peters, W.Pedrycz, Software engineering: An Engineering approach, John Wiley, 2000.

ii. R.S. Pressman, Software engineering: A Practitioners Approach, 5th Edition, McGraw Hill, 2005. iii. Sommerville, Software engineering, 8th Edition, Addison Wesley, 2008.

iv. D. Ghezzi, M. Jazayeri, D. Mandrioli, Fundamentals of software Engineering, Prentice Hall of India,

2004.