Course Type	Course Code	Name of Course	L	T	P	Credit
DE	NCSD508	**Advanced Software Engineering**	3	0	0	3

Course Objective
• Knowledge of basic to advanced software engineering methods. • A General understanding of designing, developing, maintaining high quality object-oriented and component-based software to produce efficient, reliable, robust and cost-effective solutions of real-world problem.
Learning Outcomes
At the end of the course the students should be able to: • Basic to advanced knowledge and understanding of software engineering methods. • Ability to apply software engineering principles and techniques. • Ability to design, develop, maintain and evaluate object-orientated software. • To produce efficient, reliable, robust and cost-effective solutions for component-based software. • Ability to design a good software.

Unit No.	Topics to be Covered	Lecture Hours	Learning Outcome
1	**Software Engineering** Introduction, Software Development Life Cycle, software process models, requirement analysis and design, software design process, coding, software testing, implementation and Maintenance, Software Metrics.	4	Basics of software engineering will be covered to set the platform for the course.
2	**Software Reliability** Introduction, Software Failure Mechanisms, Reliability Measurement Techniques, Reliability Models, Reliability Metrics, Fault Tolerance.	4	This will describe the various reliability models and measurement techniques.
3	**Object-Oriented Software Engineering** Introduction, object-orientated paradigm, object modeling languages, object-oriented analysis, object-oriented design, object- oriented metrics, object-oriented case tools, object-oriented software testing.	8	It will describe the use of object-oriented programming (OOP) concepts and related methodologies to design, develop, and maintain software systems that are modular, reusable, maintainable, and adaptable.

4	**Component-based Software Engineering** Introduction, CBSE and software reuse, CBSE vs. object-oriented software engineering, CBSE processes, domain engineering, component engineering, component-based software development life cycle, component vs. object, component- oriented programming, component-oriented programming vs. object-oriented programming, component-based technology, component-based software testing, component-oriented metrics.	8	Learn about improving the software development productivity, promote code reuse, enhance system modularity, and facilitate the construction of flexible and maintainable software systems by leveraging modular and interoperable software components.
5	**Aspect-Oriented Software Engineering** Introduction,Software engineering with aspects, aspects, aspect vs. object, aspect vs. component, join points and point-cuts, separation of concerns, crosscutting concerns, scattering and tangling, aspect-oriented programming, aspect-oriented software testing.	5	This will help in enhancing the design, development, and maintenance of software systems by effectively managing crosscutting concerns, improving modularity, and promoting reusable and adaptable components.
6	**Cleanroom Software Engineering** Cleanroom approach, Functional Specification, Cleanroom Design, Cleanroom Testing.	5	Learn to produce high-quality software with a strong emphasis on defect prevention rather than defect detection.
7	**Software Re-Engineering and Reverse Engineering** Re-engineering concept and approaches, redevelopment vs. reengineering, reengineering process, software re-engineering techniques, reverse engineering, levels of reverse engineering: re-documentation, design recovery, specification recovery, conditions for reverse engineering, forward engineering, restructuring, re-engineering, benefits of reverse engineering.	8	Helps to improve the maintainability, upgrade technology, enhance performance and usability, address quality issues, gain system understanding, reconstruct designs, facilitate integration and interoperability, modernize legacy systems, and protect intellectual property.
Total		**42**	

Text Books:

1. Pressman Roger S., Software Engineering: Practitioner's Approach, McGraw-Hill Inc, 9th Edition.
2. Ian Sommerville, Software Engineering, Pearson Education, 9th Edition.

Reference Books:

1. Pankaj Jalote, An Integrated Approach to Software Engineering, Narosa Publishing House, New Delhi, 2005.
2. J. Rumbaugh, M. Blaha, W. Premerlani, Object-Oriented Modeling and Design, PHI, 1991.
3. George T. Heineman, William T. Councill, Component-Based Software Engineering: Putting the Pieces Together, Addision Wesley, 2001.
4. Robert E. Filman, Tzilla Elrad, Siobhán Clarke, Mehmet Aksit, Aspect-Oriented Software Development, Addison-Wesley Professional, 2004.