Daffodil Institute of IT

Department of COMPUTER Technology

Semester Plan

Course Title: Principles of Software Engineering

Course Code: 66661

Semester: 6th

Course Title: Principles of Software Engineering

Course Code: 66661

Semester: 6th

Course Objective:

To be able to …

• To study the approaches of application of engineering to software.
• To develop knowledge and skill to apply systematic, disciplined, quantifiable approach to the development,
operation, and maintenance of software.

Short Description:

Concept of software engineering, Basics of Software development life cycle (SDLC), Project management,
Requirements analysis, Design basics, Analysis & Design tools, Design strategies, User Interface design,
understanding of Design complexity, Software implementation, Testing and quality assurance, Maintenance, CASE
tools overview;

Course Teacher:

Santosh Kumar Sushil
Head of Computer Department,
Dept. of Computer Technology
Daffodil Institute of IT
Cell: 01814328156
E-mail: santosh.kumar@diit.info

Course Structure:

SL. NO	SUBJECT CODE	NAME OF THE SUBJECT	T	P	C	MARKS				
						THEORY		PRACTICAL		TOTAL MARKS
						TC	TF	PC	PF	
1	66661	Principles of Software Engineering	2	6	4	40	60	50	50	200

Course Plan:

Class	Chapter	Detail Description
01	**01**	1**. Understand the concept of software engineering** 1.1 Define software engineering. 1.2 Describe the evolution of software engineering. 1.3 List software evolution laws.
02		1. **Understand the concept of software engineering** 1.4 Describe E-Type software evolution laws. 1.5 Describe software paradigms. 1.6 Necessity of software engineering. 1.7 List the characteristics of good software.
03	**02**	2**. Understand the basics of software development life cycle (SDLC)** 2.1 Describe the software development life cycle activities. 2.2 Describe software development paradigm (Waterfall model, Iterative model, spiral model, agile development) 2.3 Describe agile development.
04		2**. Understand the basics of software development life cycle (SDLC)** 2.4 State the agile manifesto. 2.5 List agile manifesto items. 2.6 List key principles of agile. 2.7 Describe agile methodologies
05	**03**	**3. Understand the software project management** 3.1 State the need of software project management. 3.2 Describe role of software project manager. 3.3 List software management activities.
06		**3. Understand the software project management** 3.4 Describe configuration management. 3.5 Describe project management tools.

07	**04**	**4. Understand software requirement engineering** 4.1 Describe software requirement engineering process. 4.2 List requirement elicitation process. 4.3 Describe requirement elicitation techniques.
08		**4. Understand software requirement engineering** 4.4 List software requirements characteristics. 4.5 Describe types of software requirements. 4.6 Describe the role of software system analyst. 4.7 List software metrics and measures.
09	**05**	5**. Understand the software design basics, analysis and design tools** 5.1 Describe software design levels. 5.2 State modularization and concurrency. 5.3 State coupling and cohesion
10		5**. Understand the software design basics, analysis and design tools** 5.4 Describe design verification. 5.5 State data flow diagram, structure charts. 5.6 Describe Hierarchical Input Process Output (HIPO) diagram.
11		5**. Understand the software design basics, analysis and design tools** 5.7 State pseudo code. 5.8 Describe decision table. 5.9 Describe entity relationship model. 5.10 State data dictionary.
12	**06**	**6. Understand software design strategies** 6.1 Define structured design. 6.2 Describe function-oriented design.
13		**6. Understand software design strategies** 6.3 Describe object oriented design. 6.4 Describe software design patterns. 6.5 Describe software design approaches.

14	**07**	**7. Understand user interface design** 7.1 Describe command line interface (CLI). 7.2 Describe graphical user interface (GUI).
15		**7. Understand user interface design** 7.3 State user interface design activities. 7.4 List GUI implementation tools. 7.5 State user interface golden rules.
16	**08**	**8. Understand software design complexity** 8.1 Describe Halstead's complexity measures. 8.2 Describe Cyclomatic complexity measures. 8.3 State function point
17	**09**	**9. Understand software implementation** 9.1 Describe structured programming. 9.2 State functional programming.
18		**9. Understand software implementation** 9.3 State programming style and coding guideline. 9.4 Describe software documentation 9.5 State software implementation challenges.
19	**10**	**10. Understand software testing process** 10.1 Describe software validation and verification 10.2 State manual vs automated testing 10.3 Describe testing approaches
20		**10. Understand software testing process** 10.4 State testing levels 10.5 Describe testing documentation 10.6 State testing vs quality control & assurance and audit
21	**11**	**11. Understand software maintenance overview** 11.1 Describe types of maintenance 11.2 List cost of maintenance 11.3 State maintenance activities
22		**11. Understand software maintenance overview** 11.4 State software re-engineering 11.5 Describe component reusability

23	**12**	**12. Understand Scrum agile method** 12.1 Describe scrum framework and sprints 12.2 Sate scrum roles 12.3 State scrum master roles
24		**12. Understand Scrum agile method** 12.4 Describe scrum events (sprint, planning, daily scrum meeting, sprint review, retrospective) 12.5 State artifacts 12.6 State user stories
25		**12. Understand Scrum agile method** 12.7 Describe burn down charts 12.8 State estimation process 12.9 State scrum tools and benefits