SARDAR RAJA COLLEGE OF ENGINEERING, ALANGULAM

DEPARTMENT OF COMPUTER APPLICATIONS

MICRO LESSON PLAN

SUBJECT NAME	**:**	**SOFTWARE ENGINEERING**
SUBJECT CODE	**:**	**MC 9233**
DEPT	**:**	**MCA**
YEAR/SEM	**:**	**II/III**

Handled by

Mrs.P.UMA

Asst.Prof / MCA

MC 9233 SOFTWARE ENGINEERING **L T P C 3 0 0 3**

UNIT I INTRODUCTION **9**

Software Engineering Paradigms – Waterfall Life Cycle Model – Spiral Model – Prototype Model – Fourth Generation Techniques – Planning – Cost Estimation – Organization Structure – Software Project Scheduling, – Risk Analysis and Management – Requirements and Specification – Rapid Prototyping.

UNIT II SOFTWARE DESIGN **9**

Abstraction – Modularity – Software Architecture – Cohesion – Coupling – Various Design Concepts and Notations – Real Time and Distributed System Design – Documentation – Dataflow Oriented Design – Jackson System development – Designing for Reuse – Programming Standards.

UNIT III SOFTWARE METRICS **9**

Scope – Classification of Metrics – Measuring Process and Product Attributes – Direct and Indirect Measures – Reliability – Software Quality Assurance – Standards.

UNIT IV SOFTWARE TESTING AND MAINTENANCE **9**

Software Testing Fundamentals – Software Testing Strategies – Black Box Testing – White Box Testing – System Testing – Testing Tools – Test Case Management – Software Maintenance Organization – Maintenance Report – Types of Maintenance.

UNIT V SOFTWARE CONFIGURATION MANAGEMENT (SCM) & CASE TOOLS **9**

Need for SCM – Version Control – SCM Process – Software Configuration Items – Taxonomy – Case Repository – Features.

Total = 45 Hours

REFERENCES:

1. Roger S. Pressman, "Software Engineering: A Practitioner Approach", Sixth Edition, McGrawHill, 2005. Part-1, Part-2, Part-3
2. I. Sommerville, "Software Engineering", Sixth Edition, Addison Wesley-Longman, 2004.
3. Pankaj Jalote, "An Integrated Approach to Software Engineering", Second Edition, Springer Verlag, 1997

MC 9233 SOFTWARE ENGINEERING

COURSE DESCRIPTION:

- Software engineering is the branch of computer science that creates practical, cost-effective solutions to computing and information processing problems, preferentially by applying scientific knowledge, developing software systems in the service of mankind.
- This course covers the fundamentals of software engineering, including understanding system requirements, finding appropriate engineering compromises, effective methods of design, coding, and testing, team software development, and the application of engineering tools.
- The course will combine a strong technical focus with a capstone project providing the opportunity to practice engineering knowledge, skills, and practices in a realistic development.

COURSE OBJECTIVIES:

- To define software engineering and explain its importance
- To discuss the concepts of software products and software processes
- To explain the importance of process visibility
- To introduce the notion of professional responsibility

Micro Lesson Plan

Hours	Lecture Topics	Reference Books
	UNIT I-INTRODUCTION	
1	Software Engineering Paradigms	
2	Waterfall Life Cycle Model	
3	Spiral Model, Prototype Model	
4	Fourth Generation Techniques , Planning	**R1**
5	Cost Estimation, Organization Structure	
6	Software Project Scheduling	
7	Risk analysis and Management	
8	Requirements and Specification	
9	Rapid Prototyping	
	UNIT II-SOFTWARE DESIGN	
10	Abstraction, Modularity	
11	Software Architecture,	
12	Cohesion, Coupling	
13	Various Design Concepts and Notations	
14	Real time and Distributed System Design, Documentation	**R1**
15	Dataflow Oriented design	
16	Jackson System development	
17	Designing for reuse	
18	Programming standards	
	UNIT III-SOFTWARE METRICS	
19	Scope	
20	Classification of metrics	
21	Measuring Process and Product attributes	
22	Measuring Process and Product attributes	**R1**
23	Direct and Indirect measures	
24	Reliability	
25	Software Quality Assurance	

26	Software Quality Assurance	
27	Standards	
	UNIT IV- SOFTWARE TESTING AND MAINTENANCE	
28	Software Testing Fundamentals	
29	Software testing strategies	
30	Black Box Testing, White Box Testing	
31	System Testing	**R2**
32	Testing Tools	
33	Test Case Management	
34	Software Maintenance Organization	
35	Maintenance Report	
36	Types of Maintenance	
	UNIT V-SOFTWARE CONFIGURATION MANAGEMENT (SCM) & CASETOOLS	
37	Need for SCM	
38	Version Control	
39	SCM process	
40	SCM process	**R3**
41	Software Configuration Items	
42	Taxonomy	
43	Case Repository	
44	Case Repository	
45	Features	

Reg. No. : | | | | | | | | | | | |

Question Paper Code : 96746

M.C.A. DEGREE EXAMINATION, AUGUST 2011.

Second Semester

DMC 1923 — SOFTWARE ENGINEERING

(Regulation 2009)

Time : Three hours Maximum : 100 marks

Answer ALL questions.

PART A — (10 × 2 = 20 marks)

1. How does software differ from the artifacts produced by other engineering disciplines?

2. Mention the drawbacks in software reuse and also give your suggestions to bring the reuse in software industry.

3. What is meant by the term modularity' in the context of software design? Why modularity is considered desirable?

4. What kind of metrics you use for finding software quality?

5. List the advantages and disadvantages of using LOC as a metric.

6. What are the steps to be followed to estimate effort?

7. Distinguish between "known risk" and "predictable risk".

8. Why is integration testing harder than unit testing?

9. What is the difference between "white box testing" and "black box testing"?

10. In general, how do we know when to stop testing and declare it "done", or at least done for now?

PART B — (5 × 16 = 80 marks)

11. (a) (i) Why do we need different process models? Describe the process model you would adopt for the car manufacturing project and justify your choice with its advantages and disadvantages. (10)

(ii) Brief the various applications of software. (6)

Or

(b) (i) Explain in detail the methods available for project scheduling? With an example describe how to track the schedule. (8)

(ii) For the following project description, generate estimates for total effort, project duration and number of people needed using the intermediate COCOMO model. (8)

12. (a) Consider the following scenario :

EZ Videos is the local video store. To borrow Videos or DVD's, a customer must first apply for a membership card by completing an application form and providing identification. When hiring items, the customer presents the items together with their membership card to the clerk. The clerk then totals their rental. The customer pays the rental and is issued a receipt that includes the return date for the items. Items rented are recorded on the Customer's record. If the items are overdue a notice is placed on the Customer records and the Customer must pay a late fee the next time they borrow an item. A monthly report, together with the receipts for the daily banking from the store, is sent to the Accounting Department listing the monthly rentals and any late items outstanding

(i) Construct a Context Level DFD for the above. (8)

(ii) Decompose your answer in (a) to a Level-0 DFD (8)

Or

(b) (i) Explain the role of functional independence, coupling and cohesion with respect to modular design. (12)

(ii) Describe the various design issues to be considered while designing software. (4)

13. (a) (i) Distinguish between "Process" and "Project" metrics. Give examples. (8)

(ii) What is defect classification? How can an organization make use of this metrics for its process improvement? (8)

Or

(b) Who should do Quality assurance? Mention the goals of software quality group and also norms for formal technical review meeting. (16)

14. (a) (i) Consider the following algorithm : (8)

```
Public double calculate (int amount)
{
Double rushcharge=0;
if (nextday.equals ("yes"))
{
rushcharge= 14.50;
}
double tax=amount*.0725;
if (amount>=1000)
{
shipcharge=amount*.06+rushcharge;
}
else if(amou .t>=200)
{
shipcharge=amount*.08+rushcharge;
}
else if (amount>= 100)
{
shipcharge=13.25+rushcharge;
}
else if(amount>=50)
{
shipcharge=9.95+rushcharge;
}
else if(amount>=25)
{
shipcharge=7.25+rushcharge;
}
else
```

```
{
shipcharge=5.25+rushcharge;
}
total=amount+tax+shipcharge;
return total;
}//end calculate
```

(1) Draw the flow graph

(2) Determine the cyclomatic complexity.

(3) Determine the basis set of independent paths.

(4) Prepare the test cases.

(ii) Integration testing can be tracked top-down or bottom-up. Describe each of these strategies. (8)

Or

(b) What is software maintenance? Describe various categories of maintenance. Which category consumes maximum effort and why?

15. (a) How does software configuration management facilitate the changes that may occur during various stages of a system development lifecycle? Illustrate your explanation with all example at each stage. (16)

Or

(b) Discuss in detail about configuration management tool and its usage in SCM activities. (16)