SOFTWARE ENGINEERING PROGRAM EDUCATIONAL OBJECTIVES:

1. Apply software engineering theory, principles, tools and processes, as well as the theory and principles of computer science and mathematics, to the development and maintenance of complex, scalable software systems.
2. Design and experiment with software prototypes
3. Select and use software metrics
4. Communicate effectively through oral and written reports, and software documentation
5. Elicit, analyze and specify software requirements through a productive working relationship with project stakeholders
6. Demonstrate professionalism including continued learning and professional activities.
7. Contribute to society by behaving ethically and responsibly.
8. Successfully assume a variety of roles in teams of diverse membership.
9. Apply a systematic, disciplined, quantifiable approach to the cost-effective development, operation and maintenance of software systems to the satisfaction of their beneficiaries.
10. Build solutions using different technologies, architectures and life-cycle approaches in the context of different organizational structures.
11. Insist the development, adoption and sustained use of standards of excellence for software engineering practices.

SOFTWARE ENGINEERING PROGRAM OUTCOMES:

- Upon completion of the course, students would have obtained:
- An ability to apply knowledge of mathematics, science, and engineering.
- An ability to design and conduct experiments, as well as to analyze and interpret data.
- An ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, safety, and sustainability.
- Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.
- An ability to identify, formulate, and solve engineering problems.
- An understanding of professional and ethical responsibility.
- An ability to communicate effectively.
- Demonstrate a knowledge and understanding of management and business practices, such as risk and change management, and understand their limitations.
- A recognition of the need for, and an ability to engage in life-long learning.
- An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.
- An understanding of real-time, safety-critical, embedded computer systems.

AFFILIATED INSTITUTIONS

ANNA UNIVERSITY, CHENNAI

REGULATIONS - 2013

M.E. SOFTWARE ENGINEERING

I TO IV SEMESTERS CURRICULA AND SYLLABI (FULL TIME)

SEMESTER I

SL. NO.	COURSE CODE	COURSE TITLE	L	T	P	C
THEORY						
1.	MA7155	Applied Probability and Statistics	3	1	0	4
2.	SE7101	Software Risk Management and Maintenance	3	0	0	3
3.	SE7102	Advances in Software Engineering	3	0	0	3
4.	SE7103	Formal Models of Software Systems	3	0	0	3
5.	CP7102	Advanced Data Structure and Algorithms	3	0	0	3
6.		Elective I	3	0	0	3
PRACTICAL						
7.	SE7111	Software Requirements and Design Laboratory	0	0	4	2
8.	SE7112	Advanced Data Structures Laboratory	0	0	4	2
		TOTAL	**18**	**1**	**8**	**23**

SEMESTER II

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
THEORY						
1.	SE7201	Software Project Planning and Management	3	0	0	3
2.	SE7202	Software Testing	3	0	0	3
3.	SE7203	Software Metrics and Quality Assurance	3	0	0	3
4.	IF7203	Data Warehousing and Data Mining	3	0	0	3
5.	SE7204	Big Data Analytics	3	0	0	3
6.		Elective II	3	0	0	3
PRACTICAL						
7.	SE7211	Software Testing Laboratory	0	0	4	2
8.	SE7212	Socially Relevant Mini Project	0	0	4	2
		TOTAL	**18**	**0**	**8**	**22**

SEMESTER III

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
THEORY						
1.	SE7301	Software Design Patterns	3	0	0	3
2.		Elective III	3	0	0	3
3.		Elective IV	3	0	0	3
4.		Elective V	3	0	0	3
PRACTICAL						
4.	SE7311	Project Work (Phase I)	0	0	12	6
		TOTAL	**12**	**0**	**12**	**18**

SEMESTER IV

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
PRACTICAL						
1.	SE7411	Project Work (Phase II)	0	0	24	12
		TOTAL	**0**	**0**	**24**	**12**

TOTAL NO OF CREDITS:75

LIST OF ELECTIVES

ELECTIVE I

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
1.	IF7013	Energy Aware Computing	3	0	0	3
2.	IF7202	Cloud Computing	3	0	0	3
3.	NE7002	Mobile and Pervasive Computing	3	0	0	3
4.	SE7001	Distributed System	3	0	0	3
5.	CP7028	Enterprise Application Integration	3	0	0	3

ELECTIVE II

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
6	SE7010	Software Architecture	3	0	0	3
7.	MU7011	Video Compression	3	0	0	3
8.	SE7002	Pattern Classification and Analysis	3	0	0	3
9.	CP7012	Computer Vision	3	0	0	3
10.	MU7008	User Interface Design	3	0	0	3

ELECTIVE III

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
11.	IF7301	Soft Computing	3	0	0	3
12.	SE7003	Machine Learning	3	0	0	3
13.	CP7024	Information Retrieval Techniques	3	0	0	3
14.	SE7004	Software Agents	3	0	0	3

ELECTIVE IV

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
15.	MP7001	XML and Web Services	3	0	0	3
16.	SE7005	Web Engineering and Management	3	0	0	3
17.	NE7011	Mobile Application Development	3	0	0	3
18.	NE7012	Social Network Analysis	3	0	0	3

ELECTIVE V

SL. NO	COURSE CODE	COURSE TITLE	L	T	P	C
19.	SE7006	Software Reliability	3	0	0	3
20.	SE7007	Software Documentation	3	0	0	3
21.	SE7008	Software Refactoring	3	0	0	3
22.	CP7015	Model Checking and Program Verification	3	0	0	3
23.	CP7006	Parallel Programming Paradigms	3	0	0	3
24.	SE7009	Software Process Models	3	0	0	3

MA7155 APPLIED PROBABILITY AND STATISTICS L T P C
3 1 0 4

OBJECTIVES:

- To introduce the basic concepts of one dimensional and two dimensional Random Variables.
- To provide information about Estimation theory, Correlation, Regression and Testing of hypothesis.
- To enable the students to use the concepts of multivariate normal distribution and principle components analysis.

UNIT I ONE DIMENSIONAL RANDOM VARIABLES 9

Random variables - Probability function – Moments – Moment generating functions and their properties – Binomial, Poisson, Geometric, Uniform, Exponential, Gamma and Normal distributions – Functions of a Random Variable.

UNIT II TWO DIMENSIONAL RANDOM VARIABLES 9

Joint distributions – Marginal and Conditional distributions – Functions of two dimensional random variables – Regression Curve – Correlation.

UNIT III ESTIMATION THEORY 9

Unbiased Estimators – Method of Moments – Maximum Likelihood Estimation - Curve fitting by Principle of least squares – Regression Lines.

UNIT IV TESTING OF HYPOTHESES 9

Sampling distributions - Type I and Type II errors - Testsbased on Normal, t, Chi-Square and F distributions for testing of mean, variance and proportions – Tests for Independence of attributes and Goodness of fit.

UNIT V MULTIVARIATE ANALYSIS 9

Random Vectors and Matrices - Mean vectors and Covariance matrices - Multivariate Normal density and its properties - Principal components Population principal components - Principal components from standardized variables.

TOTAL 45+15:60 PERIODS

OUTCOMES:

- The student will able to acquire the basic concepts of Probability and Statistical techniques for solving mathematical problems which will be useful in solving Engineering problems

REFERENCES:

1. Jay L. Devore, “Probability and Statistics for Engineering and the Sciences”, Thomson and Duxbury, 2002.
2. Richard Johnson. ”Miller & Freund’s Probability and Statistics for Engineer”, Prentice – Hall, Seventh Edition, 2007.
3. Richard A. Johnson and Dean W. Wichern, “Applied Multivariate Statistical Analysis”, Pearson Education, Asia, Fifth Edition, 2002.
4. Gupta S.C. and Kapoor V.K.”Fundamentals of Mathematical Statistics”, Sultan and Sons, 2001.
5. Dallas E Johnson, “Applied Multivariate Methods for Data Analysis”, Thomson and Duxbury press, 1998.

SE7101 SOFTWARE RISK MANAGEMENT AND MAINTENANCE LT P C
3 0 0 3

OBJECTIVES:

- To understand the various risk levels in software development
- to gain expertise in discovering risk and usage of risk assessment tools
- to understand the risk plan , implementation and tracking risks
- to realize the software maintenance process, measurement and benchmarking
- to expertise in the SQA maintenance tools

UNIT I RISK CULTURE AND MANAGEMENT PROCESS 9

Risk- Basic Terms- Risk Vocabulary – Risk- Driven Project Management- Controlling the Process, Environment and Risk- Maturity in Risk Culture – Risk Scale – Preparing for Risk – Risk Management- Paradigms- Five Models of Risk Management – Thinking about Less Risky alternatives – Risk Management at Different Levels – Risk Escalation – Risk Models- Risk Intelligence - Software Risk Management steps.

UNIT II DISCOVERING RISK AND ASSESSMENT 9

Identifying software risk- Classification of Risks – Risk Taxonomy – Risk Mapping – Statements – Risk Reviews – Risk Ownership and stakeholder management – Risk Assessment Approach – Risk Assessment tools and techniques – Risk Probability, impact, exposure, matrix and Application Problem- Self- assessment checklist.

UNIT III RESPONDING TO RISKS AND TRACKING 9

Special Treatment for Catastrophic risks- Constraint Risks – Risk Mitigation Plan Case Study – Contingency Plans- Implementing Risk Response- Tracking Risk Response and Hazards – Trigger Levels- Tracking Project Risks and Operational Risks- Learning by Tracking and Risk Tracker Tool.

UNIT IV MAINTENANCE PROCESS 9

Software Maintenance- Customer's Viewpoint- Economics of Maintenance- Issues in Maintenance- Software Maintenance Standard, Process, Activities and Categories – Maintenance Measurement – Service Measurement and Benchmarking – Problem Resolution- Reporting – Fix Distribution.

UNIT V ACTIVITIES FOR MAINTENANCE 9

Role of SQA for Support and Maintenance – SQA tools for Maintenance- Configuration Management and Maintenance – Maintenance of Mission Critical Systems – Global Maintenance Teams – Foundation of S3m Process Model- Exemplary Practices.

TOTAL: 45PERIODS

OUTCOMES:

- To students will be able to learn about various risk levels in software development
- Students are trained to discover risk and how to use risk assessment tools
- Students will be able to prepare risk plan, implement and track risks
- They learn about measurement, benchmarking and SQA maintenance tools

REFERENCES:

1. C. RavindranathPandian, “Applied Software Risk Management: A guide for Software Project Managers”, Auerbach Publications, 2007.
2. John Mcmanus, “Risk Management in Software Development Projects”, Elsevier Butterworth- Heinemann, First Edition, 2004.
3. Alian April and Alain Abran, “Software Maintenance Management: Evaluation and Continuous Improvement”, John Wiley & Sons Inc, 2008.
4. Gopalaswamy Ramesh and Ramesh Bhattiprolu, “Software Maintenance: Effective Practices for Geographically Distributed Environments”, Second Reprint, Tata McGraw-Hill, 2009.

SE7102 ADVANCES IN SOFTWARE ENGINEERING L T P C
3 0 0 3

OBJECTIVES:

- To have a clear understanding of Software Engineering concepts.
- To gain knowledge of the Analysis and System Design concepts.
- To learn how to manage change during development.
- To learn the SOA and AOP concepts.

UNIT I INTRODUCTION 9

System Concepts – Software Engineering Concepts - Software Life Cycle– Development Activities – Managing Software Development – Unified Modelling Language – Project Organization – Communication.

UNIT II ANALYSIS 9

Requirements Elicitation – Use Cases – Unified Modelling Language, Tools – Analysis Object Model (Domain Model) – Analysis Dynamic Models – Non-functional requirements – Analysis Patterns.

UNIT III SYSTEM DESIGN 9

Overview of System Design – Decomposing the system -System Design Concepts – System Design Activities – Addressing Design Goals – Managing System Design.

UNIT IV IMPLEMENTATION AND MANAGING CHANGE 9

Programming languages and coding- Human computer interaction-Reusing Pattern Solutions – Specifying Interfaces – Mapping Models to Code – Testing Rationale Management – Configuration Management – Project Management -real time interface design(eg: mobile design)

UNIT V ASPECT ORIENTED SOFTWARE DEVELOPMENT 9

AO Design Principles -Separations of Concerns, Subject Oriented Decomposition, Traits, Aspect Oriented Decomposition, Theme Approach, Designing Base and Crosscutting Themes, Aspect-Oriented Programming using Aspect-J.

TOTAL: 45PERIODS

OUTCOMES:

- A clear understanding of Software Engineering concepts.
- Knowledge gained of Analysis and System Design concepts.
- Ability to manage change during development.
- Basic idea of the SOA and AOP concepts.

REFERENCES:

1. Bernd Bruegge, Alan H Dutoit, Object-Oriented Software Engineering, 2nd ed, Pearson Education, 2004.
2. Craig Larman, Applying UML and Patterns, 3rd ed, Pearson Education, 2005.
3. Stephen Schach, Software Engineering 7th ed, McGraw-Hill, 2007.
4. AspectJ in Action, RamnivasLaddad, Manning Publications, 2003
5. Aspect-Oriented Software Development, Robert E. Filman, TzillaElrad, Siobhan Clarke, and Mehmet Aksit, October 2006.
6. Aspect-Oriented Software Development with Use Cases, (The Addison-Wesley Object Technology Series), Ivar Jacobson and Pan-Wei Ng, December 2004
7. Aspect-Oriented Analysis and Design: The Theme Approach, (The Addison-Wesley Object Technology Series), Siobhàn Clarke and Elisa Baniassad, March 2005.
8. Mastering AspectJ: Aspect-Oriented Programming in Java, Joseph D. Gradecki and Nicholas Lesiecki, March 2003.

SE7103 FORMAL MODELS OF SOFTWARE SYSTEMS LT P C
3 0 0 3

UNIT I FOUNDATIONS OF Z 9

Understanding formal methods – motivation for formal methods – informal requirements to formal specifications – validating formal specifications – Overview of Z specification – basic elements of Z – sets and types – declarations – variables – expressions – operators – predicates and equations.

UNIT II STRUCTURES IN Z 9

Tuples and records – relations, tables, databases – pairs and binary relations – functions – sequences – propositional logic in Z – predicate logic in Z – Z and boolean types – set comprehension – lambda calculus in Z – simple formal specifications – modeling systems and change.

UNIT III Z SCHEMAS AND SCHEMA CALCULUS 9

Z schemas – schema calculus – schema conjunction and disjunction – other schema calculus operators – schema types and bindings – generic definitions – free types – formal reasoning – checking specifications – precondition calculation – machine-checked proofs.

UNIT IV Z CASE STUDIES 9

Case Study: Text processing system – Case Study: Eight Queens – Case Study: Graphical User Interface – Case Study: Safety critical protection system – Case Study: Concurrency and real time systems.

UNIT V Z REFINEMENT 9

Refinement of Z specification – generalizing refinements – refinement strategies – program derivation and verification – refinement calculus – data structures – state schemas – functions and relations – operation schemas – schema expressions – refinement case study.

TOTAL: 45PERIODS

REFERENCES:

1. Jonathan Jacky, “The way of Z: Practical programming with formal methods”, Cambridge University Press, 1996.
2. Antony Diller, “Z: An introduction to formal methods”, Second Edition, Wiley, 1994.
3. Jim Woodcock and Jim Davies, “Using Z – Specification, Refinement, and Proof”, Prentice Hall, 1996.
4. J. M. Spivey, “The Z notation: A reference manual”, Second Edition, Prentice Hall, 1992.

5. M. Ben-Ari, “Mathematical logic for computer science”, Second Edition, Springer, 2003.
6. M. Huth and M. Ryan, “Logic in Computer Science – Modeling and Reasoning about systems”, Second Edition, Cambridge University Press, 2004.

CP7102 ADVANCED DATA STRUCTURES AND ALGORITHMS LT P C
3 0 0 3

OBJECTIVES:

- To understand the principles of iterative and recursive algorithms.
- To learn the graph search algorithms.
- To study network flow and linear programming problems.
- To learn the hill climbing and dynamic programming design techniques.
- To develop recursive backtracking algorithms.
- To get an awareness of NP completeness and randomized algorithms.
- To learn the principles of shared and concurrent objects.
- To learn concurrent data structures.

UNIT I ITERATIVE AND RECURSIVE ALGORITHMS 9

Iterative Algorithms:Measures of Progress and Loop Invariants-Paradigm Shift: Sequence of Actions versus Sequence of Assertions- Steps to Develop an Iterative Algorithm-Different Types of Iterative Algorithms--Typical Errors-Recursion-Forward versus Backward- Towers of Hanoi-Checklist for Recursive Algorithms-The Stack Frame-Proving Correctness with Strong Induction- Examples of Recursive Algorithms-Sorting and Selecting Algorithms-Operations on Integers- Ackermann’s Function- Recursion on Trees-Tree Traversals- Examples- Generalizing the Problem - Heap Sort and Priority Queues-Representing Expressions.

UNIT II OPTIMISATION ALGORITHMS 9

Optimization Problems-Graph Search Algorithms-Generic Search-Breadth-First Search-Dijkstra’s Shortest-Weighted-Path -Depth-First Search-Recursive Depth-First Search-Linear Ordering of a Partial Order- Network Flows and Linear Programming-Hill Climbing-Primal Dual Hill Climbing- Steepest Ascent Hill Climbing-Linear Programming-Recursive Backtracking-Developing Recursive Backtracking Algorithm- Pruning Branches-Satisfiability

UNIT III DYNAMIC PROGRAMMING ALGORITHMS 9

Developing a Dynamic Programming Algorithm-Subtle Points- Question for the Little Bird-Subinstances and Subsolutions-Set of Subinstances-Decreasing Time and Space-Number of Solutions-Code. Reductions and NP-Completeness**-**Satisfiability-Proving NP-Completeness- 3-Coloring- Bipartite Matching. Randomized Algorithms**-**Randomness to Hide Worst Cases-Optimization Problems with a Random Structure.

UNIT IV SHARED OBJECTS AND CONCURRENT OBJECTS 9

Shared Objects and Synchronization -Properties of Mutual Exclusion-The Moral- The Producer–Consumer Problem -The Readers–Writers Problem-Realities of Parallelization-Parallel Programming- Principles- Mutual Exclusion-Time- Critical Sections--Thread Solutions-The Filter Lock-Fairness-Lamport’s Bakery Algorithm-Bounded Timestamps-Lower Bounds on the Number of Locations-Concurrent Objects- Concurrency and Correctness-Sequential Objects-Quiescent Consistency- Sequential Consistency-Linearizability- Formal Definitions- Progress Conditions- The Java Memory Model

UNIT V CONCURRENT DATA STRUCTURES 9

Practice-Linked Lists-The Role of Locking-List-Based Sets-Concurrent Reasoning- Coarse-Grained Synchronization-Fine-Grained Synchronization-Optimistic Synchronization- Lazy Synchronization-Non-Blocking Synchronization-Concurrent Queues and the ABA Problem-Queues-A Bounded Partial Queue-An Unbounded Total Queue-An Unbounded Lock-Free Queue-Memory Reclamation and the ABA Problem- Dual Data Structures- Concurrent Stacks and Elimination- An Unbounded Lock-Free Stack- Elimination-The Elimination Backoff Stack.

TOTAL: 45PERIODS

OUTCOMES:

Upon completion of the course, the students will be able to

- Design and apply iterative and recursive algorithms.
- Design and implement optimisation algorithms in specific applications.
- Design appropriate shared objects and concurrent objects for applications.
- Implement and apply concurrent linked lists, stacks, and queues.

REFERENCES:

1. Jeff Edmonds, “How to Think about Algorithms”, Cambridge University Press, 2008.
2. M. Herlihy and N. Shavit, “The Art of Multiprocessor Programming”, Morgan Kaufmann, 2008.
3. Steven S. Skiena, “The Algorithm Design Manual”, Springer, 2008.
4. Peter Brass, “Advanced Data Structures”, Cambridge University Press, 2008.
5. S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, “Algorithms” , McGrawHill, 2008.
6. J. Kleinberg and E. Tardos, "Algorithm Design“, Pearson Education, 2006.
7. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, “Introduction to Algorithms“, PHI Learning Private Limited, 2012.
8. Rajeev Motwani and Prabhakar Raghavan, “Randomized Algorithms”, Cambridge University Press, 1995.
9. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, “The Design and Analysis of Computer Algorithms**”,** Addison-Wesley, 1975.
10. A. V. Aho, J. E. Hopcroft, and J. D. Ullman,”Data Structures and Algorithms”, Pearson,2006.

SE7111 SOFTWARE REQUIREMENTS AND DESIGN LABORATORY L T P C

0 0 4 2

1. The students should develop all the necessary requirements based on IEEE standards or any other standardized standards and should prepare requirement document and design document after completion.
2. Use any open source software for requirements elicitation, requirements analysis and requirements validation.
3. Use any open source software for performing software design based on the requirements obtained in 2 for each system.

1. ONLINE SHOPPING MALL
PROJECT DESCRIPTION:

The Online Shopping Mall (OSM) application enables vendors to set up online shops, customers to browse through the shops, and a system administrator to approve and reject requests for new shops and maintain lists of shop categories. Also on the agenda is designing an online shopping site to manage the items in the shop and also help customers purchase them online without having to visit the shop physically.

The online shopping mall will showcase a complete shopping experience in a small package.

This project envisages bridging the gap between the seller, the retailer and the customer. A very high flexibility is being maintained in the design process so that this project can take the following path: -

- A multiple merchant venue with each merchant having his/her own window which the customer can visit to browse and subsequently buy the products.
- Maintaining the deliverable goods as well as services through single or multiple windows is also on the agenda.

Target Users:

Mall Administrator: The Mall Administrator is the super user and has complete control over all the activities that can be performed. The application notifies the administrator of all shop creation requests, and the administrator can then approve or reject them. The administrator also manages the list of available product categories. The administrator can also view and delete entries in the guestbook.

Shop Owner: Any user can submit a shop creation request through the application. When the request is approved by the Mall Administrator, the requester is notified, and from there on is given the role of Shop Owner. The Shop Owner is responsible for setting up the shop and maintaining it. The job involves managing the sub-categories of the items in the shop. Also, the shop owner can add or remove items from his shop. The Shop Owner can view different reports that give details of the sales and orders specific to his shop. The Shop Owner can also decide to close shop and remove it from the mall.

Mall Customer/Guests: A Mall Customer can browse through the shops and choose products to place in a virtual shopping cart. The shopping cart details can be viewed and items can be removed from the cart. To proceed with the purchase, the customer is prompted to login. Also, the customer can modify personal profile information (such as phone number and shipping address) stored by the application. The customer can also view the status of any previous orders.

EMPLOYEES:

- Purchase department under a Purchase manager to overlook purchasing activities if warehousing needs arise.
- Sales department under a Sales manager who will look after the sale of products and services.
- Accounts department under an Accounts manager to look after the accounting activities of the enterprise.

2. BANKING SYSTEM

PROJECT DESCRIPTION:

A bank has several automated teller machines (ATMs), which are geographically distributed and connected via a wide area network to a central server. Each ATM machine has a card reader, a cash dispenser, a keyboard/display, and a receipt printer. By using the ATM machine, a customer can withdraw cash from either checking or savings account, query the balance of an account, or transfer funds from one account to another. A transaction is initiated when a customer inserts an ATM card into the card reader. Encoded on the magnetic strip on the back of the ATM card are the card number, the start date, and the expiration date.

Assuming the card is recognized, the system validates the ATM card to determine that the expiration date has not passed, that the user-entered PIN (personal identification number) matches the PIN maintained by the system, and that the card is not lost or stolen. The customer is allowed three attempts to enter the correct PIN; the card is confiscated if the third attempt fails. Cards that have been reported lost or stolen are also confiscated.

If the PIN is validated satisfactorily, the customer is prompted for a withdrawal, query, or transfer transaction. Before withdrawal transaction can be approved, the system determines that sufficient funds exist in the requested account, that the maximum daily limit will not be exceeded, and that there are sufficient funds available at the local cash dispenser.

If the transaction is approved, the requested amount of cash is dispensed, a receipt is printed containing information about the transaction, and the card is ejected. Before a transfer transaction can be approved, the system determines that the customer has at least two accounts and that there are sufficient funds in the account to be debited. For approved query and transfer requests, a receipt is printed and card ejected.

A customer may cancel a transaction at any time; the transaction is terminated and the card is ejected. Customer records, account records, and debit card records are all maintained at the server.

3. CAMPUS MANAGEMENT SYSTEM

PROJECT DESCRIPTION:

The Campus Management System; is fully computerized information organization, storage and retrieval system that could provide us any information about an Institute just at the click of a mouse. The most fascinating asset about a computerized College fee Manager is that it enables us to explore any institute related information at any time on demand and that too in an absolutely user friendly environment that could be accessed even by a layman very easily

OBJECTIVES AND GOALS:

- To automate the functions at a Higher Education Institute, the main missions of this software are as under
- To provide user-friendly interface to the college administrator
- To minimize the typing errors during data entry
- To search record of a particular object (course, student, faculty etc.)
- To update the record of an object
- To generate various reports for management
- To print various reports
- To reduce the typing work by keeping maximum information available on the screen

- To reduce the expenditure involving stationery items such as paper, ledgers, fee receipt book etc.
- To provide consistent, updated and reliable data at any time on demand.
- To analyse, plan and forecast the inflation or recession graph of the in college in the near future based on the college's record of revenue sources and expenditure.
- To provide the most important feature of maintaining the valuable back-up of the critical data.
- To be bestowed with the latest security facilities provided by the modern computerized DBMS.

PROJECT BUILDING BLOCKS:

Enrolment Management , Portal management , Admissions/Recruiting , Faculty Information, Student Services, Student Portal, Hostel management, Parking and Security, Student Health , Student Placement ,Campus Incidents, Faculty Portal , Forum Portal , Student Billing ,Alumni Portal.

4. AIR TRAFFIC CONTROL SYSTEM

Air traffic control is a closed loop activity in which pilots state the intent by filing flight plans. Controllers then plan traffic flow based on the total number of flight plans and, when possible, given clearance to pilots to fly according to their plans. When planning conflicts arise, controllers resolve them by clearing pilots to fly alternatives to their plans to avoid the conflicts. If unpredicted atmospheric conditions (e.g., wind speed or direction) or pilot actions cause deviations from conflict-free planned routings, controllers issue clearances for tactical maneuvers that solve any resultant problem, albeit not necessarily in a way that furthers the pilot's goal of reaching the planned destination at a certain time.

PROBLEM FORMULATION:

Design an air traffic control system (ATCS) that is fault tolerant and scalable, according to the specific requirements listed in the following sections. The primary objective of the ATCS is to provide separation services for aircraft that are flying in controlled air space, or where poor visibility prevents from maintaining visual separation. Aircraft are separated from one another and from terrain hazards.

SPECIFIC SOFTWARE REQUIREMENTS:

The requirements of ATCSs include real-time aspects. The ATCS is a "dynamic" real-time system. Its loading will vary significantly over time, and has no upper bound. Loading scenarios can vary significantly, hence the average loading of the ATCS is not a highly useful metric for schedulability and other analyses. Although an upper bound could possibly be imposed artificially, this may not be a cost-effective solution, since pre-allocation of computing resources for such a worst case would lead to very poor resource utilization. A dynamic resource management policy is thus preferred.

5. CAFETERIA ORDERING SYSTEM

The Cafeteria Ordering System is a new system that replaces the current manual and telephone processes for ordering and picking up lunches in the Process Impact cafeteria.

Patron: A Patron is a Process Impact employee at the corporate campus in TidalPark, Chennai, who wishes to order meals to be delivered from the company cafeteria.

There are about 600 potential Patrons, of which an estimated 400 are expected to use the Cafeteria Ordering System .Patrons will sometimes order multiple meals for group events or guests. An estimated 90 percent of orders will be placed using the corporate Intranet, with 10 percent of orders being placed from home. All Patrons have Intranet access from their offices. Some Patrons will wish to set up meal subscriptions, either to have the same meal to be delivered every day or to have the day's meal special delivered automatically. A Patron must be able to override a subscription for a specific day.

Cafeteria Staff: The Process Impact cafeteria currently employs about 20 Cafeteria Staff, who will receive orders from the Cafeteria Ordering System, prepare meals, and package them for delivery, print delivery instructions, and request delivery. Most of the Cafeteria Staff will need to be trained in the use of the computer, the Web browser, and the Cafeteria Ordering System.

Menu Manager: The Menu Manager is a cafeteria employee, perhaps the cafeteria manager, who is responsible for establishing and maintaining daily menus of the food items available from the cafeteria and the times of day that each item is available. Some menu items may not be available for delivery. The Menu Manager will also define the cafeteria's daily specials. The Menu Manager will need to edit the menus periodically to reflect planned food items that are not available or price changes.

Meal Deliverer: As the Cafeteria Staff prepare orders for delivery, they will print delivery instructions and issue delivery requests to the Meal Deliverer, who is either another cafeteria employee or a contractor. The Meal Deliverer will pick up food and delivery instructions for each meal and deliver it to the Patron. The Meal Deliverer's primary interactions with the system will be to reprint the deliveryinstructions on occasion and to confirm that a meal was (or was not) delivered.

TOTAL: 60 PERIODS

SE7112 ADVANCED DATA STRUCTURES LABORATORY L T P C
0 0 4 2

OBJECTIVES:

- To learn to implement iterative and recursive algorithms.
- To learn to design and implement algorithms using hill climbing and dynamic programming techniques.
- To learn to implement shared and concurrent objects.
- To learn to implement concurrent data structures.

LAB EXERCISES:

Each student has to work individually on assigned lab exercises. Lab sessions could be scheduled as one contiguous four-hour session per week or two two-hour sessions per week. There will be about 15 exercises in a semester. It is recommended that all implementations are carried out in Java. If C or C++ has to be used, then the threads library will be required for concurrency. Exercises should be designed to cover the following topics:

- Implementation of graph search algorithms.
- Implementation and application of network flow and linear programming problems.
- Implementation of algorithms using the hill climbing and dynamic programming design techniques.
- Implementation of recursive backtracking algorithms.
- Implementation of randomized algorithms.

- Implementation of various locking and synchronization mechanisms for concurrent linked lists, concurrent queues, and concurrent stacks.
- Developing applications involving concurrency.

TOTAL: 60 PERIODS

OUTCOMES:

Upon completion of the course, the students will be able to

- Design and apply iterative and recursive algorithms.
- Design and implement algorithms using the hill climbing and dynamic programming and recursive backtracking techniques.
- Design and implement optimisation algorithms for specific applications.
- Design and implement randomized algorithms.
- Design appropriate shared objects and concurrent objects for applications.
- Implement and apply concurrent linked lists, stacks, and queues.

REFERENCES:

1. Jeff Edmonds, “How to Think about Algorithms”, Cambridge University Press, 2008.
2. M. Herlihy and N. Shavit, “The Art of Multiprocessor Programming”, Morgan Kaufmann, 2008.
3. Steven S. Skiena, “The Algorithm Design Manual”, Springer, 2008.
4. Peter Brass, “Advanced Data Structures”, Cambridge University Press, 2008.
5. S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, “Algorithms” , McGrawHill, 2008.
6. J. Kleinberg and E. Tardos, "Algorithm Design“, Pearson Education, 2006.
7. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, “Introduction to Algorithms“, PHI Learning Private Limited, 2012.
8. Rajeev Motwani and Prabhakar Raghavan, “Randomized Algorithms”, Cambridge University Press, 1995.
9. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, “The Design and Analysis of Computer Algorithms**”,** Addison-Wesley, 1975.
10. A. V. Aho, J. E. Hopcroft, and J. D. Ullman,”Data Structures and Algorithms”, Pearson,2006.

SE7201 SOFTWARE PROJECT PLANNING AND MANAGEMENT L T P C
3 0 0 3

OBJECTIVES:

- To understand the various software processes.
- To learn format process models.
- To gain knowledge of the overall project activities
- To analyses the various issues in each phase of project management and people management.

UNIT I BASIC CONCEPTS 9

Product, Process and Project – Definition, Software Process Maturity ,Software maturity Framework, Principles of Software Process Change, Software Process Assessment, The Initial Process, The Repeatable Process, The Defined Process, The Managed Process, The Optimizing Process, Product Life Cycle-Project Life Cycle Models.

UNIT II FORMAT PROCESS MODELS AND THEIR USE 9

Definition and format model for a process, The ISO 9001 and CMM models and their relevance to project Management-other emerging models like People CMM.

UNIT III UMBRELLA ACTIVITIES IN PROJECTS 9

Software Project Management -Formal Technical Reviews-Software Quality Assurance-Software Configuration Management-Re-usability Management-Risk analysis and Management -Measurement and Metrics-Document Preparation and Production

UNIT IV IN STREAM ACTIVITIES IN PROJECTS 9

Project Initiation - Project Planning- feasibility study estimation- resource allocation- execution and tracking,-root cause analysis- Project Wind-up-Concept of process/project database.

UNITV ENGINEERING AND PEOPLE ISSUES IN PROJECT MANAGEMENT 9

Phases (Requirements, Design, Development, Testing, maintenance, deployment) - engineering activities and management issues in each phase-Difficulties in people management - Role of Project manager ,Special considerations in project management for India and geographic distribution issues.

OUTCOMES:

- Get the basic knowledge about various processes.
- Emphasize the use of format process models.
- Knowledge gained in usage and application of umbrella activities for project management
- Execute the project development in a systematic manner using tools and techniques
- Issues are analysed in various phases of project management and people management

TOTAL: 45 PERIODS

REFERENCES:

1. Ramesh, " Gopalaswamy: Managing Global Projects ", Tata McGraw Hill, 2001.
2. Humphrey, Watts: “Managing the software process ", Addison Wesley, 1986.
3. Pressman, Roger: “Software Engineering ", A Practitioner’s approach, McGraw Hill, 1997.
4. DeMarco and Lister: “Peopleware ".
5. Wheelwright and Clark: “Revolutionising product development ", The Free Press, 1993.
   Watts Humphrey, “Managing the Software Process “, Pearson Education, New Delhi, 2000
6. PankajJalote, “Software Project Management in practice”, Pearson Education, New Delhi, 2002.

SE7202 SOFTWARE TESTING L T P C
3 0 0 3

OBJECTIVES:

- To know the behavior of the testing techniques to detect the errors in the software
- To understand standard principles to check the occurrence of defects and its removal.
- To learn the functionality of automated testing tools
- To understand the models of software reliability.

UNIT I TESTIING ENVIRONMENT AND TEST PROCESSES 9

World-Class Software Testing Model – Building a Software Testing Environment - Overview of Software Testing Process – Organizing for Testing – Developing the Test Plan – Verification Testing – Analysing and Reporting Test Results – Acceptance Testing – Operational Testing – Post Implementation Analysis

UNIT II TESTING TECHNIQUES AND LEVELS OFTESTING 9

Using White Box Approach to Test design - Static Testing Vs. Structural Testing – Code Functional Testing – Coverage and Control Flow Graphs –Using Black Box Approaches to Test Case Design – Random Testing – Requirements based testing –Decision tables –State-based testing – Cause-effect graphing – Error guessing – Compatibility testing – Levels of Testing - Unit Testing - Integration Testing - Defect Bash Elimination. System Testing - Usability and Accessibility Testing – Configuration Testing - Compatibility Testing - Case study for White box testing and Black box testing techniques.

UNIT III INCORPORATING SPECIALIZED TESTING RESPONSIBILITIES 9

Testing Client/Server Systems – Rapid Application Development Testing – Testing in a Multiplatform Environment – Testing Software System Security - Testing Object-Oriented Software – Object Oriented Testing – Testing Web based systems – Web based system – Web Technology Evolution – Traditional Software and Web based Software – Challenges in Testing for Web-based Software –Testing a Data Warehouse - Case Study for Web Application Testing.

UNIT IV TEST AUTOMATION 9

Selecting and Installing Software Testing Tools - Software Test Automation – Skills needed for Automation – Scope of Automation – Design and Architecture for Automation – Requirements for a Test Tool – Challenges in Automation – Tracking the Bug – Debugging – Case study using Bug Tracking Tool.

UNIT V SOFTWARE TESTING AND QUALITY METRICS 9

Testing Software System Security - Six-Sigma – TQM - Complexity Metrics and Models – Quality Management Metrics - Availability Metrics - Defect Removal Effectiveness - FMEA - Quality Function Deployment – Taguchi Quality Loss Function – Cost of Quality. Case Study for Complexity and Object Oriented Metrics.

TOTAL: 45 PERIODS

OUTCOMES:

- Test the software by applying testing techniques to deliver a product free from bugs
- Evaluate the web applications using bug tracking tools.
- Investigate the scenario and the able to select the proper testing technique
- Explore the test automation concepts and tools
- Deliver quality product to the clients by way of applying standards such as TQM, Six Sigma
- Evaluate the estimation of cost, schedule based on standard metrics

REFERENCES:

1. William Perry, “Effective Methods of Software Testing”, Third Edition, Wiley Publishing 2007
2. Srinivasan Desikan and Gopalaswamy Ramesh, “Software Testing – Principles and Practices”, Pearson Education, 2007.
3. NareshChauhan, “Software Testing Principles and Practices” Oxford University Press, New Delhi, 2010.
4. Dale H. Besterfiled et al., “Total Quality Management”, Pearson Education Asia, Third Edition, Indian Reprint (2006).
5. Stephen Kan, “Metrics and Models in Software Quality”, Addison – Wesley, Second Edition, 2004.
6. LleneBurnstein, “ Practical Software Testing”, Springer International Edition, Chennai, 2003
7. RenuRajani,Pradeep Oak, “Software Testing – Effective Methods, Tools and Techniques”, Tata McGraw Hill,2004.
8. Edward Kit, “Software Testing in the Real World – Improving the Process”, Pearson Education, 1995.

9. Boris Beizer, " Software Testing Techniques" – 2nd Edition, Van Nostrand Reinhold, New York, 1990
10. Adithya P. Mathur, "Foundations of Software Testing – Fundamentals algorithms and techniques", Dorling Kindersley (India) Pvt. Ltd., Pearson Education, 2008.

SE7203 SOFTWARE METRICS AND QUALITY ASSURANCE LT P C
3 0 0 3

OBJECTIVES:

- To understand software metrics and measurement.
- To emphasize the use of product and quality metrics.
- To explain quality assurance and various tools used in quality management.
- To learn in detail about various quality assurance models.
- To understand the audit and assessment procedures to achieve quality.

UNIT I INTRODUCTION TO SOFTWARE METRICS 9

Fundamentals of measurement-Scope of software metrics-Measurement theory-Software measurement validation software metrics data collection – Analysis methods.

UNIT II PRODUCT AND QUALITY METRICS 9

Measurement of internet product attributes-size and structure-external product attributes-measurement of quality- Software quality metrics-product quality-process quality- metrics for software maintenance.

UNIT III FUNDAMENTALS OF SOFTWARE QUALITY ASSURANCE 9

SQA basics-Software quality in business context – Planning for software quality assurance – Product quality and process quality – Software process models -Total Quality Management- 7 QC Tools and Modern Tools.

UNIT IV QUALITY ASSURANCE MODELS 9

Models for Quality Assurance-ISO-9000 – Series- CMM- CMMI-Test Maturity Models, SPICE, Malcolm Baldrige Model- P-CMM.

UNIT V SOFTWARE QUALITY ASSURANCE TRENDS 9

Software Process- PSP and TSP - OO Methodology, Clean-room software engineering, Defect injection and prevention -Internal Auditing and Assessments-Inspections & Walkthroughs.

OUTCOMES:

- Knowledge on how to choose which metrics to collect and use them to make predictions.
- Ken on product and quality metrics.
- Understand how to detect, classify, prevent and remove defects.
- Choose appropriate quality assurance models and develop quality.
- Ability to conduct formal inspections, record and evaluate results of inspections.

TOTAL: 45 PERIODS

REFERENCES:

1. Norman E-Fentor and Share Lawrence Pflieger.” Software Metrics”. International Thomson Computer Press, 1997.
2. Stephen H.Kan,”Metric and Models in software Quality Engineering”, Addison QWesley 1995.
3. S.A.Kelkar,”Software quality and Testing, PHI Learing, Pvt, Ltd., New Delhi 2012.
4. Watts S Humphrey, “Managing the Software Process”, Pearson Education Inc, 2008.
5. Mary Beth Chrissis, Mike Konrad and Sandy Shrum, “CMMI”, Pearson Education(Singapore) Pte Ltd, 2003
6. Philip B Crosby, " Quality is Free: The Art of Making Quality Certain ", Mass Market, 1992.

IF7203 DATA WAREHOUSING AND DATA MINING L T P C
3 0 0 3

OBJECTIVES:

- To expose the students to the concepts of Data warehousing Architecture and Implementation
- To Understand Data mining principles and techniques and Introduce DM as a cutting edge business intelligence
- To learn to use association rule mining for handling large data
- To understand the concept of classification for the retrieval purposes
- To know the clustering techniques in details for better organization and retrieval of data
- To identify Business applications and Trends of Data mining

UNIT I DATA WAREHOUSE 8

Data Warehousing - Operational Database Systems vs. Data Warehouses - Multidimensional Data Model - Schemas for Multidimensional Databases – OLAP Operations – Data Warehouse Architecture – Indexing – OLAP queries & Tools.

UNIT II DATA MINING & DATA PREPROCESSING 9

Introduction to KDD process – Knowledge Discovery from Databases - Need for Data Pre-processing – Data Cleaning – Data Integration and Transformation – Data Reduction – Data Discretization and Concept Hierarchy Generation.

UNIT III ASSOCIATION RULE MINING 8

Introduction - Data Mining Functionalities - Association Rule Mining - Mining Frequent Itemsets with and without Candidate Generation - Mining Various Kinds of Association Rules - Constraint-Based Association Mining.

UNIT IV CLASSIFICATION & PREDICTION 10

Classification vs. Prediction – Data preparation for Classification and Prediction – Classification by Decision Tree Introduction – Bayesian Classification – Rule Based Classification – Classification by Back Propagation – Support Vector Machines – Associative Classification – Lazy Learners – Other Classification Methods – Prediction – Accuracy and Error Measures – Evaluating the Accuracy of a Classifier or Predictor – Ensemble Methods – Model Section.

UNIT V CLUSTERING 10

Cluster Analysis: - Types of Data in Cluster Analysis – A Categorization of Major Clustering Methods – Partitioning Methods – Hierarchical methods – Density-Based Methods – Grid-Based Methods – Model-Based Clustering Methods – Clustering High- Dimensional Data – Constraint-Based Cluster Analysis – Outlier Analysis.

TOTAL :45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able to

- Store voluminous data for online processing
- Preprocess the data for mining applications
- Apply the association rules for mining the data
- Design and deploy appropriate classification techniques
- Cluster the high dimensional data for better organization of the data
- Discover the knowledge imbibed in the high dimensional system
- Evolve Multidimensional Intelligent model from typical system
- Evaluate various mining techniques on complex data objects

REFERENCES:

1. Jiawei Han and MichelineKamber, “Data Mining Concepts and Techniques” Second Edition, Elsevier, Reprinted 2008.
2. K.P. Soman, ShyamDiwakar and V. Ajay, “Insight into Data mining Theory and Practice”, Easter Economy Edition, Prentice Hall of India, 2006.
3. G. K. Gupta, “Introduction to Data Mining with Case Studies”, Easter Economy Edition, Prentice Hall of India, 2006.
4. BERSON, ALEX & SMITH, STEPHEN J, Data Warehousing, Data Mining, and OLAP, TMH Pub. Co. Ltd, New Delhi, 2012
5. Pang-Ning Tan, Michael Steinbach and Vipin Kumar, “Introduction to Data Mining”, Pearson Education, 2007
6. PRABHU Data Warehousing, PHI Learning Private Limited, New Delhi, 2012, ,
7. PONNIAH, PAULRAJ, Data Warehousing Fundamentals, John Wiley & Sons, New Delhi, 2011
8. MARAKAS, GEORGE M, Modern Data Warehousing, Mining, and Visualization, Pearson Education, 2011.

SE7204 BIG DATA ANALYTICS L T P C

3 0 0 3

OBJECTIVES:

- To explore the fundamental concepts of big data analytics
- To learn to analyze the big data using intelligent techniques.
- To understand the various search methods and visualization techniques.
- To learn to use various techniques for mining data stream.
- To understand the applications using Map Reduce Concepts.

UNIT I INTRODUCTION TO BIG DATA 8

Introduction to Big Data Platform – Challenges of Conventional Systems - Intelligent data analysis – Nature of Data - Analytic Processes and Tools - Analysis vs Reporting - Modern Data Analytic Tools - Statistical Concepts: Sampling Distributions - Re-Sampling - Statistical Inference - Prediction Error.

UNIT II DATA ANALYSIS 11

Regression Modeling - Multivariate Analysis – Bayesian Methods – Bayesian Paradigm - Bayesian Modeling - Inference and Bayesian Networks - Support Vector and Kernel Methods - Analysis of Time Series: Linear Systems Analysis - Nonlinear Dynamics - Rule Induction - Fuzzy Logic: Extracting Fuzzy Models from Data - Fuzzy Decision Trees

UNIT III SEARCH METHODS AND VISUALIZATION 9

Search by simulated Annealing – Stochastic, Adaptive search by Evaluation – Evaluation Strategies – Genetic Algorithm – Genetic Programming – Visualization – Classification of Visual Data Analysis Techniques – Data Types – Visualization Techniques – Interaction techniques – Specific Visual data analysis Techniques.

UNIT IV MINING DATA STREAMS 8

Introduction To Streams Concepts – Stream Data Model and Architecture - Stream Computing - Sampling Data in a Stream – Filtering Streams – Counting Distinct Elements in a Stream – Estimating Moments – Counting Oneness in a Window – Decaying Window - Real time Analytics Platform(RTAP) Applications - Case Studies - Real Time Sentiment Analysis, Stock Market Predictions.

UNIT V FRAMEWORKS 9

Map Reduce – Hadoop, Hive, MapR – Sharding – NoSQL Databases - S3 - Hadoop Distributed File Systems– Case Study.

TOTAL: 45 PERIODS

OUTCOMES:

At the end of this course the students will be able to:

- Work with big data platform and its analysis techniques.
- Analyze the big data for useful business applications.
- Select visualization techniques and tools to analyze big data
- Implement search methods and visualization techniques
- Design efficient algorithms for mining the data from large volumes.
- Explore the technologies associated with big data analytics such as NoSQL, Hadoop and Map Reduce.

REFERENCES:

1. Michael Berthold, David J. Hand, “Intelligent Data Analysis”, Springer, 2007.
2. AnandRajaraman and Jeffrey David Ullman, “Mining of Massive Datasets”, Cambridge University Press, 2012.
3. Bill Franks, “Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics”, John Wiley & sons, 2012.
4. Glenn J. Myatt, “Making Sense of Data”, John Wiley & Sons, 2007
5. Pete Warden, “Big Data Glossary”, O’Reilly, 2011.
6. Jiawei Han, MichelineKamber “Data Mining Concepts and Techniques”, Second Edition, Elsevier, Reprinted 2008.
7. Da Ruan,Guoquing Chen, Etienne E.Kerre, Geert Wets, Intelligent Data Mining, Springer,2007
8. Paul Zikopoulos ,Dirk deRoos , Krishnan Parasuraman , Thomas Deutsch , James Giles , David Corrigan, Harness the Power of Big Data The IBM Big Data Platform, Tata McGraw Hill Publications, 2012

9. Michael Minelli (Author), Michele Chambers (Author), AmbigaDhiraj (Author) , Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses,Wiley Publications,2013
10. Zikopoulos, Paul, Chris Eaton,Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data, Tata McGraw Hill Publications, 2011.

SE7211 SOFTWARE TESTING LABORATORY L T P C
0 0 4 2

CASE STUDY 1

Cause Effect Graph Testing for a Triangle Program

Perform cause effect graph testing to find a set of test cases for the following program specification: Write a program that takes three positive integers as input and determine if they represent three sides of a triangle, and if they do, indicate what type of triangle it is. To be more specific, it should read three integers and set a flag as follows:

- If they represent a scalene triangle, set it to 1.
- If they represent an isosceles triangle, set it to 2.
- If they represent an equilateral triangle, set it to 3.
- If they do not represent a triangle, set it to 4.

CASE STUDY 2

Boundary Value Analysis for a Software Unit

The following is a specification for a software unit. The unit computes the average of 25 floating point numbers that lie on or between bounding values which are positive values from 1.0 (lowest allowed boundary value) to 5000.0 (highest allowed boundary value). The bounding values and the numbers to average are inputs to the unit. The upper bound must be greater than the lower bound. If an invalid set of values is input for the boundaries an error message appears and the user is reported. If the boundary values are valid the unit computes the sum and the average of the numbers on and within the bounds. The average and sum are output by the unit, as well as the total number of inputs that lie within the boundaries. Derive a set of equivalence classes for the averaging unit using the specification, and complement the classes using boundary value analysis. Be sure to identify valid and invalid classes.

Design a set of test cases for the unit using your equivalence classes and boundary values. For each test case, specify the equivalence classes covered, input values, expected outputs, and test case identifier. Show in tabular form that you have covered all the classes and boundaries. Implement this module in the programming language of your choice. Run the module with your test cases and record the actual outputs. Save an uncorrected version of the program for future use.

CASE STUDY 3

Cyclomatic Complexity for Binary Search

Draw a control flow graph for the given binary search code and clearly label each node so that it is linked to its corresponding statement. Calculate its cyclomatic complexity.

```
intbinsearch (intx,int v[], int n)
{
int low, high, mid;
low =0;
high  = n-1;
while (low <=high) {
mid =(low+high)/2
if (x < v[mid]
high = mid&ndash;1;
else if (x > v[mid])
low = mid+1;
else /* found match*/
return mid;
}
return–1; /* no match*/
}
```

CASE STUDY 4

Data Flow Testing for Gregorian Calendar

A program was written to determine if a given year in the Gregorian calendar is a leap year. The well-known part of the rule, stipulating that it is a leap year if it is divisible by 4, is implemented correctly in the program. The programmer, however, is unaware of the exceptions: A centenary year, although divisible by 4, is not a leap year unless it is also divisible by 400. Thus, while year 2000 was a leap year, the years 1800 and 1900 were not. Determine if the following test-case selection criteria are reliable or valid.

(a) C1(T) ≡ (T = {1, 101, 1001, 10001})

(b) C2(T) ≡ (T = {t|1995 ≥ t ≥ 2005})

(c) C3(T) ≡ (T = {t|1895 ≥ t ≥ 1905})

(d) C4(T) ≡ (T = {t} ∧ t ∋ {400, 800, 1200, 1600, 2000, 2400})

(e) C5(T) ≡ (T = {t, t + 1, t + 2, t + 3, t + 4} ∧ t ∋ {100, 200, 300, 400,500})

(f) C6(T) ≡ (T = {t, t + 1, t + 2, . . . , t + 399} ∧ t ∋ D)

(g) C7(T) ≡ (T = {t1, t2, t3} ∧ t1, t2, t3 ∋ D)

CASE STUDY 5

State based Testing for an Assembler

Suppose you were developing a simple assembler whose syntax can be described as follows :
<statement_ :: = <label field><op code><address>
<label field> :: = ‘‘none’’ | <identifier> :
<op code> :: = MOVE | JUMP
<address> :: = <identifier> | <unsigned integer>

A stream of tokens is input to the assembler. The possible states for such an assembler are: S1, prelabel; S2, label; S3, valid op code; S4, valid address; S5, valid numeric address. Start, Error, and Done. A table that describes the inputs and actions for the assembler is as follows:

Inputs	Actions
no more tokens	A1: Put the label in the symbol table.
Identifier	A2: Look up the op code and store its binary value in op code field.
MOVE, JUMP	A3: Look up symbol in symbol table and store its value in address field.
colon	A4: Convert number to binary, and store that value in address field.
Integer	A5: Place instruction in the object module, and print a line in the listing.
	A6: Print error message and put all zeroes in the instruction.

Using this information and any assumptions you need to make, develop a state transition diagramfor the assembler. From the state transition diagram develop a set of test cases that will cover allof the state transitions. Be sure to describe the exact sequence of inputs as well as the expected Sequence of state changes and actions.

CASE STUDY 6

Stress Testing of a Map-Aided Vehicle Tracking and Scheduling System

The American package courier and freight business faced the double pressures of consolidation and unstoppable increases in fuel costs. In mid-2008, pump prices were already double those prevailing in early 2007. As well, the recent decision of long-time price leader DHL to “co-locate” dozens of routes with erstwhile competitor UPS revealed just how fragile are market positions built through decades of promotions.

In Omaha, regional freight leader Red Ball Trucking1 was keener than most to maximize operating efficiencies out of its substantial fleet of trucks and vans and thereby maintain margins in the face of low-cost rivals. In March 2008, a brand-new map-based adjunct to the company’s proprietary logistics and routing system neared rollout. Extensive “white box”, line-by-line testing had eliminated most ofthe gross errors but the Red Ball CEO was concerned about the scalability of the program test bed.

Find out whether the map-enhanced vehicle tracking and scheduling system would remain stable at benchmarks of 50, 100 and 1000 concurrent users. Clean up any remaining bugs not caught by in-house.

CASE STUDY 7

Model Based Testing

Design and develop a scientific calculator program using various GUI components and events. Build the test model for the same. Determine the inputs that can be given to the model. Calculate expected output for the model. Run the test cases. Compare the actual output with the expected output. Any model based technique can be used for building the test model.

CASE STUDY 8

Web Application Testing for Student Grade System

With educational organizations under increasing pressure to improve their performance to secure funding for future provision of programmes, it is vital that they have accurate, up-to-date information. For this reason, they have MIS systems to record and track student enrolment and results on completion of a learning programme. In this way they can monitor achievement statistics.All student assignment work is marked and recorded by individual module tutors using a spreadsheet, or similar, of their own design. In the computing department these results are

input into a master spreadsheet to track a student's overall progress throughout their programme of study. This is then made available to students through the web portal used in college. Perform web application testing for this scenario.

TOTAL: 60 PERIODS

SE7212 SOCIALLY RELEVANT MINI PROJECT L T P C
0 0 4 2

- Choose any project of solving social problems
- Team Project with a maximum of three in a team
- Need to concentrate on software development methodologies
- Documentation is based on the standards
- Evaluation pattern is like Lab examination,
- Need to submit a report, presentation with demo.

TOTAL:60 PERIODS

SE7301 SOFTWARE DESIGN PATTERNS L T P C
3 0 0 3

OBJECTIVES:

- How to add functionality to designs while minimizing complexity.
- What code qualities they need to maintain to keep code flexible.
- Understanding the common design patterns.
- Identifying the appropriate patterns for design problems.
- Refactoring the badly designed program properly using patterns.

UNIT I INTRODUCTION 9

Introduction – Design Patterns in Smalltalk MVC – Describing Design patterns –Catalog of Design Patterns- Organizing the Catalog –How Design Patterns Solve Design Problems – How to select a Design Pattern – How to use a Design Pattern – What makes a pattern? – Pattern Categories – Relationship between Patterns – Patterns and Software Architecture

UNIT II DESIGN PATTERNS FROM POSA1 9

Whole Part – Master Slave –Command Processor – View Handler – Forward Receiver – Client Dispatcher Server

UNIT III CREATIONAL AND STRUCTURAL DESIGN PATTERNS 9

Abstract Factory - Factory Method – Prototype - Singleton – Builder Adapter Pattern – Decorator – Façade – Proxy - Bridge

UNIT IV BEHAVIORAL DESIGN PATTERNS AND IDIOMS 9

Chain of Responsibility – Mediator – Observer – Strategy– Memento Idioms – Pattern Systems

UNIT V CASE STUDY **9**

Case Study Designing a Document Editor - What to expect from Design Patterns – A brief History of Design Patterns – The Pattern Community – Where will Patterns Go? – The Past, Present and the Future of Patterns - Anti Patterns

TOTAL: 45 PERIODS

OUTCOMES:

- Be able to Design and implement codes with higher performance and lower complexity
- Be aware of code qualities needed to keep code flexible
- Understand core design principles and be able to assess the quality of a design with respect to these principles.
- Be capable of applying these principles in the design of object oriented systems.
- Demonstrate an understanding of a range of design patterns. Be capable of comprehending adesign presented using this vocabulary.
- Be able to select and apply suitable patterns in specific contexts.
- Understand and apply refactoring techniques in the context of design patterns.

REFERENCES:

1. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, “Design patterns: Elements of Reusable object-oriented software”, Addison-Wesley, 1995.
2. Frank Bachmann, Regine Meunier, Hans Rohnert “Pattern Oriented Software Architecture” – Volume 1, 1996.
3. William J Brown et al., "Anti-Patterns: Refactoring Software, Architectures and Projects in Crisis", John Wiley, 1998.

IF7013 ENERGY AWARE COMPUTING **L T P C**
3 0 0 3

OBJECTIVES:

This course examines the design of power efficient architecture, power and performance tradeoffs, restructuring of software and applications and standards for energy aware Hardware and Software. The objective of this course is:

- To know the fundamental principles energy efficient devices
- To study the concepts of Energy efficient storage
- To introduce energy efficient algorithms
- Enable the students to know energy efficient techniques involved to support real-time systems.
- To study Energy aware applications.

UNIT I INTRODUCTION **9**

Energy efficient network on chip architecture for multi core system-Energy efficient MIPS CPU core with fine grained run time power gating – Low power design of Emerging memory technologies.

UNIT II ENERGY EFFICIENT STORAGE **9**

Disk Energy Management-Power efficient strategies for storage system-Dynamic thermal management for high performance storage systems-Energy saving technique for Disk storage systems

UNIT III ENERGY EFFICIENT ALGORITHMS **9**

Scheduling of Parallel Tasks – Task level Dynamic voltage scaling – Speed Scaling – Processor optimization- Memetic Algorithms – Online job scheduling Algorithms.

UNIT IV REAL TIME SYSTEMS **9**

Multi processor system – Real Time tasks- Energy Minimization – Energy aware scheduling- Dynamic Reconfiguration- Adaptive power management-Energy Harvesting Embedded system.

UNIT V ENERGY AWARE APPLICATIONS **9**

On chip network – Video codec Design – Surveillance camera- Low power mobile storage.

TOTAL: 45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able to

- Design Power efficient architecture Hardware and Software.
- Analyze power and performance trade off between various energy aware storage devices.
- Implement various energy aware algorithms.
- Restructure the software and Hardware for Energy aware applications.
- Explore the Energy aware applications

REFERENCES:

1. Ishfaq Ah mad, Sanjay Ranka, Handbook of Energy Aware and Green Computing, Chapman and Hall/CRC, 2012
2. Chong-Min Kyung, Sungioo yoo, Energy Aware system design Algorithms and Architecture, Springer, 2011.
3. Bob steiger wald ,Chris:Luero, Energy Aware computing, Intel Press,2012.

IF7202 CLOUD COMPUTING **L T P C**
3 0 0 3

OBJECTIVES:

- To introduce the broad perceptive of cloud architecture and model
- To understand the concept of Virtualization
- To be familiar with the lead players in cloud.
- To understand the features of cloud simulator
- To apply different cloud programming model as per need.
- To be able to set up a private cloud.
- To understand the design of cloud Services.
- To learn to design the trusted cloud Computing system

UNIT I CLOUD ARCHITECTURE AND MODEL **9**

Technologies for Network-Based System – System Models for Distributed and Cloud Computing – NIST Cloud Computing Reference Architecture.Cloud Models:- Characteristics – Cloud Services – Cloud models (IaaS, PaaS, SaaS) – Public vs Private Cloud –Cloud Solutions - Cloud ecosystem – Service management – Computing on demand.

UNIT II VIRTUALIZATION **9**

Basics of Virtualization - Types of Virtualization - Implementation Levels of Virtualization - Virtualization Structures - Tools and Mechanisms - Virtualization of CPU, Memory, I/O Devices - Virtual Clusters and Resource management – Virtualization for Data-center Automation.

UNIT III CLOUD INFRASTRUCTURE 9

Architectural Design of Compute and Storage Clouds – Layered Cloud Architecture Development – Design Challenges - Inter Cloud Resource Management – Resource Provisioning and Platform Deployment – Global Exchange of Cloud Resources.

UNIT IV PROGRAMMING MODEL 9

Parallel and Distributed Programming Paradigms – MapReduce , Twister and Iterative MapReduce – Hadoop Library from Apache – Mapping Applications - Programming Support - Google App Engine, Amazon AWS - Cloud Software Environments -Eucalyptus, Open Nebula, OpenStack, Aneka, CloudSim

UNIT V SECURITY IN THE CLOUD 9

Security Overview – Cloud Security Challenges and Risks – Software-as-a-Service Security – Security Governance – Risk Management – Security Monitoring – Security Architecture Design – Data Security – Application Security – Virtual Machine Security - Identity Management and Access Control – Autonomic Security.

TOTAL: 45 PERIODS

OUTCOMES:

- Compare the strengths and limitations of cloud computing
- Identify the architecture, infrastructure and delivery models of cloud computing
- Apply suitable virtualization concept.
- Choose the appropriate cloud player
- Choose the appropriate Programming Models and approach.
- Address the core issues of cloud computing such as security, privacy and interoperability
- Design Cloud Services
- Set a private cloud

REFERENCES:

1. Kai Hwang, Geoffrey C Fox, Jack G Dongarra, “Distributed and Cloud Computing, From Parallel Processing to the Internet of Things”, Morgan Kaufmann Publishers, 2012.
2. John W.Rittinghouse and James F.Ransome, “Cloud Computing: Implementation, Management, and Security”, CRC Press, 2010.
3. Toby Velte, Anthony Velte, Robert Elsenpeter, “Cloud Computing, A Practical Approach”, TMH, 2009.
4. Kumar Saurabh, “Cloud Computing – insights into New-Era Infrastructure”, Wiley India, 2011.
5. George Reese, “Cloud Application Architectures: Building Applications and Infrastructure in the Cloud” O’Reilly
6. James E. Smith, Ravi Nair, “Virtual Machines: Versatile Platforms for Systems and Processes”, Elsevier/Morgan Kaufmann, 2005.
7. Katarina Stanoevska-Slabeva, Thomas Wozniak, SantiRistol, “Grid and Cloud Computing – A Business Perspective on Technology and Applications”, Springer.
8. Ronald L. Krutz, Russell Dean Vines, “Cloud Security – A comprehensive Guide to Secure Cloud Computing”, Wiley – India, 2010.
9. RajkumarBuyya, Christian Vecchiola, S.ThamaraiSelvi, ‘Mastering Cloud Computing”, TMGH, 2013.
10. GautamShroff,Enterprise Cloud Computing,Cambridge University Press,2011
11. Michael Miller, Cloud Computing,Que Publishing,2008
12. Nick Antonopoulos, Cloud computing,Springer Publications,2010

NE7002 MOBILE AND PERVASIVE COMPUTING L T P C
3 0 0 3

OBJECTIVES :

- To understand the basics of Mobile Computing and Personal Computing
- To learn the role of cellular networks in Mobile and Pervasive Computing
- To expose to the concept of sensor and mesh networks
- To expose to the context aware and wearable computing
- To learn to develop applications in mobile and pervasive computing environment

UNIT I INTRODUCTION 9

Differences between Mobile Communication and Mobile Computing – Contexts and Names – Functions – Applications and Services – New Applications – Making Legacy Applications Mobile Enabled – Design Considerations – Integration of Wireless and Wired Networks – Standards Bodies – Pervasive Computing – Basics and Vision – Principles of Pervasive Computing – Categories of Pervasive Devices

UNIT II 3G AND 4G CELLULAR NETWORKS 9

Migration to 3G Networks – IMT 2000 and UMTS – UMTS Architecture – User Equipment – Radio Network Subsystem – UTRAN – Node B – RNC functions – USIM – Protocol Stack – CS and PS Domains – IMS Architecture – Handover – 3.5G and 3.9G a brief discussion – 4G LAN and Cellular Networks – LTE – Control Plane – NAS and RRC – User Plane – PDCP, RLC and MAC – WiMax IEEE 802.16d/e – WiMax Internetworking with 3GPP

UNIT III SENSOR AND MESH NETWORKS 9

Sensor Networks – Role in Pervasive Computing – In Network Processing and Data Dissemination – Sensor Databases – Data Management in Wireless Mobile Environments – Wireless Mesh Networks – Architecture – Mesh Routers – Mesh Clients – Routing – Cross Layer Approach – Security Aspects of Various Layers in WMN – Applications of Sensor and Mesh networks

UNIT IV CONTEXT AWARE COMPUTING & WEARABLE COMPUTING 9

Adaptability – Mechanisms for Adaptation - Functionality and Data – Transcoding – Location Aware Computing – Location Representation – Localization Techniques – Triangulation and Scene Analysis – Delaunay Triangulation and Voronoi graphs – Types of Context – Role of Mobile Middleware – Adaptation and Agents – Service Discovery MiddlewareHealth BAN-Medical and Technological Requirements-Wearable Sensors-Intra-BAN communications

UNIT V APPLICATION DEVELOPMENT 9

Three tier architecture - Model View Controller Architecture - Memory Management – Information Access Devices – PDAs and Smart Phones – Smart Cards and Embedded Controls – J2ME – Programming for CLDC – GUI in MIDP – Application Development ON Android and iPhone

TOTAL:45 PERIODS

OUTCOMES:

At the end of the course the student should be able to

- Design a basic architecture for a pervasive computing environment
- Design and allocate the resources on the 3G-4G wireless networks
- Analyse the role of sensors in Wireless networks
- Work out the routing in mesh network
- Deploy the location and context information for application development
- Develop mobile computing applications based on the paradigm of context aware computing and wearable computing

REFERENCES:

1. Asoke K Talukder, Hasan Ahmed, Roopa R Yavagal, “Mobile Computing: Technology, Applications and Service Creation”, 2nd ed, Tata McGraw Hill, 2010.
2. Reto Meier, “Professional Android 2 Application Development”, Wrox Wiley,2010.
3. .Pei Zheng and Lionel M Li, ‘Smart Phone & Next Generation Mobile Computing’, Morgan Kaufmann Publishers, 2006.
4. Frank Adelstein, ‘Fundamentals of Mobile and Pervasive Computing’, TMH, 2005
5. JochenBurthardt et al, ‘Pervasive Computing: Technology and Architecture of Mobile Internet Applications’, Pearson Education, 2003
6. Feng Zhao and Leonidas Guibas, ‘Wireless Sensor Networks’, Morgan Kaufmann Publishers, 2004
7. UweHansmaan et al, ‘Principles of Mobile Computing’, Springer, 2003
8. Reto Meier, “Professional Android 2 Application Development”, Wrox Wiley, 2010.
9. Mohammad s. Obaidat et al, “Pervasive Computing and Networking”,Johnwiley
10. Stefan Poslad, “Ubiquitous Computing: Smart Devices, Environments and Interactions”, Wiley, 2009.
11. Frank Adelstein Sandeep K. S. Gupta Golden G. Richard III Loren Schwiebert “Fundamentals of Mobile and Pervasive Computing, “, McGraw-Hill, 2005

SE7001 DISTRIBUTED SYSTEM L T P C
3 0 0 3

OBJECTIVE:

- To explore distributed systems principles associated with communication, naming, synchronization, distributed file systems, system design, distributed scheduling, and several case studies
- To cover both foundational concepts and well as practical deployments.

UNIT I COMMUNICATION IN DISTRIBUTED ENVIRONMENT 8

Introduction – Various Paradigms in Distributed Applications – Remote Procedure Call – Remote Object Invocation – Message-Oriented Communication – Unicasting, Multicasting and Broadcasting – Group Communication.

UNIT II DISTRIBUTED OPERATING SYSTEMS 12

Issues in Distributed Operating System – Threads in Distributed Systems – Clock Synchronization – Causal Ordering – Global States – Election Algorithms –Distributed Mutual Exclusion – Distributed Transactions – Distributed Deadlock – Agreement Protocols .

UNIT III DISTRIBUTED RESOURCE MANAGEMENT 10

Distributed Shared Memory – Data-Centric Consistency Models – Client-Centric Consistency Models – Ivy – Munin – Distributed Scheduling – Distributed File Systems – Sun NFS.

UNIT IV FAULT TOLERANCE AND CONSENSUS 7

Introduction to Fault Tolerance – Distributed Commit Protocols – Byzantine Fault Tolerance – Impossibilities in Fault Tolerance.

UNIT V CASE STUDIES 8

Distributed Object-Based System – CORBA – COM+ – Distributed Coordination-Based System – JINI.

TOTAL: 45 PERIODS

OUTCOMES:

The students will understand:

- the concepts underlying distributed systems
- how distributed systems may be constructed using a variety of tools and approaches

Students will be able to design, and implement distributed software systems in Java using:

- sockets
- remote procedure call mechanisms
- JAVA RMI

Students will demonstrate an ability to apply theory and techniques to unseen problems.

REFERENCES:

1. George Coulouris, Jean Dollimore, Tim Kindberg, "Distributed Systems Concepts and Design", Third Edition, Pearson Education Asia, 2002.
2. HagitAttiya and Jennifer Welch, "Distributed Computing: Fundamentals, Simulations and Advanced Topics", Wiley, 2004.
3. MukeshSinghal, "Advanced Concepts In Operating Systems", Mc GrawHill Series in Computer Science, 1994.
4. A.S.Tanenbaum, M.Van Steen, "Distributed Systems", Pearson Education, 2004.
5. M.L.Liu, "Distributed Computing Principles and Applications", Pearson Addison Wesley, 2004.

CP7028 ENTERPRISE APPLICATION INTEGRATION L T P C
3 0 0 3

OBJECTIVES:

- Describe approaches to enterprise application integration
- Understand the integration middleware
- Evaluate the integration approaches suitable for a given problem

UNIT I INTRODUCTION 6

Requirements for EAI - Challenges in EAI – Integration with legacy systems – Integration with partners - Heterogeneous environment – Implementation approaches – Web services, messaging, ETL, direct data integration – Middleware requirements – Approaches to integration – services oriented and messaging.

UNIT II INTEGRATION PATTERNS 6

Introduction to integration patterns – Architecture for application integration – Integration patterns – Point to point, broker, message bus, publish/subscribe, Challenges in performance, security, reliability - Case studies

UNIT III SERVICE ORIENTED INTEGRATION 12

Business process integration - Composite applications-services – Web services – Service choreography and orchestration - Business process modeling - BPMN, Business process execution - BPEL – Middleware infrastructure - Case studies

UNIT IV MESSAGING BASED INTEGRATION 9

Messaging – Synchronous and asynchronous – Message structure – Message oriented middleware – Reliability mechanisms – Challenges – Messaging infrastructure – Java Messaging Services – Case studies

UNIT V ENTERPRISE SERVICE BUS 12

Enterprise Service Bus – routing, scalable connectivity, protocol and message transformations, data enrichment, distribution, correlation, monitoring – Deployment configurations – Global ESB, Directly connected, Federated, brokered ESBs – Application server based – Messaging system based – Hardware based ESBs – Support to SOA, message based and event based integrations - Case studies.

TOTAL: 45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able to

- Describe different approaches to integration enterprise applications
- Analyze specifications and identify appropriate integration approaches
- Develop a suitable integration design for a given problem
- Identify appropriate integration middleware for a given problem
- Evaluate the integration approaches against specified requirements

REFERENCES:

1. George Mentzas and Andreas Frezen (Eds), "Semantic Enterprise Application Integration for Business Processes: Service-oriented Frameworks", Business Science Reference, 2009
2. WaseemRoshen, "SOA Based Enterprise Integration", Tata McGrawHill, 2009.
3. G Hohpe and B Woolf, "Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions", Addison-Wesley Professional, 2003
4. D Linthicum, "Next Generation Application Integration: From Simple Information to Web Services", Addison-Wesley, 2003
5. Martin Fowler, "Patterns of Enterprise Application Architecture", Addison- Wesley, 2003
6. Kapil Pant and MatiazJuric, "Business Process Driven SOA using BPMN and BPEL: From Business Process Modeling to Orchestration and Service Oriented Architecture", Packt Publishing, 2008

SE7010 SOFTWARE ARCHITECTURE L T P C
3 0 0 3

UNIT I INTRODUCTION 9

Architecture business cycle – architectural patterns – reference models – architectural structures, views – Basic Concepts of Software Architecture

UNIT II SOFTWARE ARCHITECTURAL PATTERNS 9

Architectural Patterns – Introduction to Styles – Simple Styles - Distributed and Networked Architectures- Architecture for network based applications – Decentralized Architectures

UNIT III DESIGNING FOR NON FUNCTIONAL PROPERTIES 9

Understanding Quality Attributes – Functionality and Architecture – Architecture and Quality Attributes – System Quality Attributes – Quality attribute Scenarios in Practice - Introducing Tactics – Availability Tactics – Modifiability Tactics – Performance Tactics - Security Tactics – Testability Tactics – Usability Tactics – Relationship of Tactics to Architectural Patterns – Architectural Patterns and Styles

UNIT IV ARCHITECTURE DESCRIPTION DOCUMENTATION AND EVALUATION 9

Early Architecture Description Languages – Domain and Style Specific ADLs – Extensible ADLs - Documenting Software architecture - Architecture Evaluation - ATAM

UNIT V ARCHITECTURE ADAPTATION AND CASE STUDY 9

A Conceptual Framework for Architectural Adaptation – Techniques for supporting architecture centric change- The World Wide Web – A Case Study in Interoperability.

TEXT BOOKS:

1. Len Bass, Paul Clements, Rick Kazman, “Software Architecture in Practice”, Third Edition, Addison-Wesley, 2003.
2. Richard N.Taylor, Nenad Medvidovic and Eric M.Dashofy, “Software Architecture, Foundations, Theory and Practice”, Wiley 2010.

REFERENCES:

1. Frank Buschmann, Regine Meunier, Hans Rohnert, Michael Stal, “Pattern Oriented Software Architecture” ,Volume 1, 1996.
2. Mary shaw and David Garlan, “Software Architecture – Perspectives on an emerging discipline”, Pearson education, 2008.

MU7011 VIDEO COMPRESSION L T P C
3 0 0 3

OBJECTIVES :

- To introduce principles and current technologies of multimedia systems.
- To study the issues in effectively representing, processing and transmitting multimedia data including text, graphics, sound and music, image and video.
- To study the Image, video and audio standards such as JPEG, MPEG, H.26x, Dolby Digital and AAC will be reviewed.
- To study the applications such as video conferencing, multimedia data indexing and retrieval will also be introduced.

UNIT I INTRODUCTION 9

Overview of image compression - important information theory concepts - entropy definition and interpretation - Shannon-Fanon coding - Huffman coding - Adaptive Huffman coding - Lempel-Ziv codec- QM codec, context-based QM coder - examples of lossless compression

UNIT II QUANTIZATION 9

Scalar quantization, optimal scalar quantizer, commander- Vector quantization- Audio and speech compression- JPEG & JPEG-2000 still image compression- Video coding standards (A) MPEG-1, MPEG-2

UNIT III VIDEO PROCESSING 9

Video coding standards H.264/AVC and HEVC- Video coding techniques - motion estimation, rate control algorithms, pre & post processing- Video delivery/streaming over wired and wireless networks

UNIT IV ADVANCED VIDEO CODING TECHNIQUES 9

Mobile multimedia computing- Multimedia content management and protection- Future directions – Multi-view video coding, depth coding and others

UNIT V CONTENT MANAGEMENT 9

Video Compression-Motion Compensation, H.261 standard – FMM-14 Multimedia Applications Content-based retrieval in digital libraries – FMM

TOTAL: 45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able

- To know principles and current technologies of multimedia systems
- To know issues in effectively representing, processing, and retrieving multimedia data
- To know the areas by implementing some components of a multimedia streaming system
- To know the latest web technologies and some advanced topics in current multimedia research

REFERENCES:

1. Handbook of Image and Video processing - Al Bovik (Alan C Bovik), Academic Press, Second Edition, 2005.
2. Digital Image Sequence Processing, Compression, and Analysis - Todd R. Reed, CRC Press, 2004.
3. H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia - Iain E.G. Richardson, Wiley, 2003
4. Digital Video Processing - A. Murat Tekalp, Prentice Hall, 1995
5. Andy Beach, “Real World Video Compression”Pearson Education, 2010.
6. Peter D. Symes ,” Video Compression Demystified”McGraw-Hill, 2001.
7. Yun Q. Shi, Huifang Sun,” Image and Video Compression for Multimedia Engineering Fundamentals, Algorithms, and Standards” 2nd Edition 2008.

SE7002 PATTERN CLASSIFICATION AND ANALYSIS L T P C
3 0 0 3

UNIT I IMAGE FORMATION AND IMAGE PROCESSING 9

Introduction- Geometric primitives and transformations- Photometric image formation- Sampling and aliasing, Compression- Point operators- Linear filtering- More neighbourhood operators- Fourier Transforms- Pyramids and wavelets- Geometric transformations- global optimization.

UNIT II PATTERN RECOGNITION 9

Linear Discriminant Analysis- Bayes' classifier – Neural net- Feed forward, unsupervised learning, Hopfield nets- fuzzy system-optimization techniques in Recognition-Genetic algorithm-Simulated annealing, object detection, Face recognition-Category recognition- Context and scene understanding

UNIT III FEATURE DETECTION AND SEGMENTATION 9

Points and patches-Edges-Lines-Active Contours-Split and merge-Mean shift and mode finding-Normalized cuts-Graph cuts and energy based methods- 2D and 3D feature based alignment-Pose estimation- Geometric intrinsic calibration.

UNIT IV MOTION ESTIMATION 9

Structure from motion- Two-frame structure- Factorization- Bundle adjustment- Constrained structure and motion- Translational alignment- Parametric motion- Spline based motion-Optical flow, Layered motion-Image stitching motion models-Computational photography.

UNIT V OBJECT DETECTION AND TRACKING 9

Object Detection- Neural Network-Based Face Detection- Instance and Category recognition-Freund & Schapire's AdaBoost algorithm – Context and scene understanding- Primitive tracking- Gradient Descent Tracking- Object Tracking with RBF Networks- Mean Shift tracking.

TOTAL: 45 PERIODS

REFERENCES:

1. Richard Szeliski, "Computer Vision: Algorithms and Applications", Springer Publications, 2010.
2. Schalkoff R.J., "Digital Image Processing & Computer vision", John Wiley sons, 1989.
3. Dudar R.O., and Hart P.E.,"Pattern classification and scene Analysis", 2002.
4. Object Detection and Tracking http://www.serc.iisc.ernet.in/~venky/SE263/index.html

CP7012 COMPUTER VISION L T P C
3 0 0 3

OBJECTIVES:

- To review image processing techniques for computer vision
- To understand shape and region analysis
- To understand Hough Transform and its applications to detect lines, circles, ellipses
- To understand three-dimensional image analysis techniques
- To understand motion analysis
- To study some applications of computer vision algorithms

UNIT I IMAGE PROCESSING FOUNDATIONS 9

Review of image processing techniques – classical filtering operations – thresholding techniques – edge detection techniques – corner and interest point detection – mathematical morphology – texture

UNIT II SHAPES AND REGIONS 9

Binary shape analysis – connectedness – object labeling and counting – size filtering – distance functions – skeletons and thinning – deformable shape analysis – boundary tracking procedures – active contours – shape models and shape recognition – centroidal profiles – handling occlusion – boundary length measures – boundary descriptors – chain codes – Fourier descriptors – region descriptors – moments

UNIT III HOUGH TRANSFORM 9

Line detection – Hough Transform (HT) for line detection – foot-of-normal method – line localization – line fitting – RANSAC for straight line detection – HT based circular object detection – accurate center location – speed problem – ellipse detection – Case study: Human Iris location – hole detection – generalized Hough Transform (GHT) – spatial matched filtering – GHT for ellipse detection – object location – GHT for feature collation

UNIT IV 3D VISION AND MOTION 9

Methods for 3D vision – projection schemes – shape from shading – photometric stereo – shape from texture – shape from focus – active range finding – surface representations – point-based representation – volumetric representations – 3D object recognition – 3D reconstruction – introduction to motion – triangulation – bundle adjustment – translational alignment – parametric motion – spline-based motion – optical flow – layered motion

UNIT V APPLICATIONS 9

Application: Photo album – Face detection – Face recognition – Eigen faces – Active appearance and 3D shape models of faces

Application: Surveillance – foreground-background separation – particle filters – Chamfer matching, tracking, and occlusion – combining views from multiple cameras – human gait analysis

Application: In-vehicle vision system: locating roadway – road markings – identifying road signs – locating pedestrians

TOTAL : 45 PERIODS

OUTCOMES:

Upon completion of the course, the students will be able to

- Implement fundamental image processing techniques required for computer vision
- Perform shape analysis
- Implement boundary tracking techniques
- Apply chain codes and other region descriptors
- Apply Hough Transform for line, circle, and ellipse detections
- Apply 3D vision techniques
- Implement motion related techniques
- Develop applications using computer vision techniques

REFERENCES:

1. E. R. Davies, “Computer & Machine Vision”, Fourth Edition, Academic Press, 2012.
2. R. Szeliski, “Computer Vision: Algorithms and Applications”, Springer 2011.
3. Simon J. D. Prince, “Computer Vision: Models, Learning, and Inference”, Cambridge University Press, 2012.

4. Mark Nixon and Alberto S. Aquado, “Feature Extraction & Image Processing for Computer Vision”, Third Edition, Academic Press, 2012.
5. D. L. Baggio et al., “Mastering OpenCV with Practical Computer Vision Projects”, Packt Publishing, 2012.
6. Jan Erik Solem, “Programming Computer Vision with Python: Tools and algorithms for analyzing images”, O’Reilly Media, 2012.

MU7008 USER INTERFACE DESIGN L T P C
3 0 0 3

OBJECTIVES:

- To understand the basics of User Interface Design.
- To design the user interface, design, menu creation and windows creation
- To understand the concept of menus, windows, interfaces, business functions, various problems in windows design with colour, text, Non-anthropomorphic Design.
- To study the design process and evaluations.

UNIT I INTERACTIVE SOFTWARE AND INTERACTION DEVICE 9

Human–Computer Interface – Characteristics Of Graphics Interface –Direct Manipulation Graphical System – Web User Interface –Popularity –Characteristic & Principles.

UNIT II HUMAN COMPUTER INTERACTION 9

User Interface Design Process – Obstacles –Usability –Human Characteristics In Design – Human Interaction Speed –Business Functions –Requirement Analysis – Direct – Indirect Methods – Basic Business Functions – Design Standards – General Design Principles – Conceptual Model Design – Conceptual Model Mock-Ups

UNIT III WINDOWS 9

Characteristics– Components– Presentation Styles– Types– Managements– Organizations– Operations– Web Systems– System Timings - Device– Based Controls Characteristics– Screen – Based Controls –– Human Consideration In Screen Design – Structures Of Menus – Functions Of Menus– Contents Of Menu– Formatting – Phrasing The Menu – Selecting Menu Choice– Navigating Menus– Graphical Menus. Operate Control – Text Boxes– Selection Control– Combination Control– Custom Control– Presentation Control.

UNIT IV MULTIMEDIA 9

Text For Web Pages – Effective Feedback– Guidance & Assistance– Internationalization– Accessibility– Icons– Image– Multimedia – Coloring- Case Study: Addressing usability in E-Commerce sites

UNIT V DESIGN PROCESS AND EVALUATION 9

User Interface Design Process - Usability Testing - Usability Requirements and Specification procedures and techniques- User Interface Design Evaluation

TOTAL: 45 PERIODS

OUTCOMES:

- Knowledge on development methodologies, evaluation techniques and user interface building tools
- Explore a representative range of design guidelines
- Gain experience in applying design guidelines to user interface design tasks.
- Ability to design their own Human Computer

REFERENCES:

1. Wilbent. O. Galitz ,"The Essential Guide To User Interface Design", John Wiley& Sons, 2001.
2. Deborah Mayhew, The Usability Engineering Lifecycle, Morgan Kaufmann, 1999Ben Shneiderman, "Design The User Interface", Pearson Education, 1998.
3. Alan Cooper, "The Essential Of User Interface Design", Wiley – Dream Tech Ltd., 2002. Sharp, Rogers, Preece, 'Interaction Design', Wiley India Edition, 2007
4. Alan Dix et al, " Human - Computer Interaction ", Prentice Hall, 1993.
5. Ben Schneiderman, " Designing the User Interface ", Addison Wesley, 2000.

IF7301 SOFT COMPUTING L T P C
3 0 0 3

OBJECTIVES:

- To learn the key aspects of Soft computing
- To know about the components and building block hypothesis of Genetic algorithm.
- To understand the features of neural network and its applications
- To study the fuzzy logic components
- To gain insight onto Neuro Fuzzy modeling and control.
- To gain knowledge in machine learning through Support vector machines.

UNIT I INTRODUCTION TO SOFT COMPUTING 9

Evolution of Computing - Soft Computing Constituents – From Conventional AI to Computational Intelligence - Machine Learning Basics

UNIT II GENETIC ALGORITHMS 9

Introduction, Building block hypothesis, working principle, Basic operators and Terminologies like individual, gene, encoding, fitness function and reproduction, Genetic modeling: Significance of Genetic operators, Inheritance operator, cross over, inversion & deletion, mutation operator, Bitwise operator, GA optimization problems, JSPP (Job Shop Scheduling Problem), TSP (Travelling Salesman Problem),Differences & similarities between GA & other traditional methods, Applications of GA.

UNIT III NEURAL NETWORKS 9

Machine Learning using Neural Network, Adaptive Networks – Feed Forward Networks – Supervised Learning Neural Networks – Radial Basis Function Networks - Reinforcement Learning – Unsupervised Learning Neural Networks – Adaptive Resonance Architectures – Advances in Neural Networks.

UNIT IV FUZZY LOGIC 9

Fuzzy Sets – Operations on Fuzzy Sets – Fuzzy Relations – Membership Functions-Fuzzy Rules and Fuzzy Reasoning – Fuzzy Inference Systems – Fuzzy Expert Systems – Fuzzy Decision Making

UNIT V NEURO-FUZZY MODELING 9

Adaptive Neuro-Fuzzy Inference Systems – Coactive Neuro-Fuzzy Modeling – Classification and Regression Trees – Data Clustering Algorithms – Rule base Structure Identification – Neuro-Fuzzy Control – Case Studies.

TOTAL: 45 PERIODS

OUTCOMES:

- Implement machine learning through neural networks.
- Write Genetic Algorithm to solve the optimization problem
- Develop a Fuzzy expert system.
- Model Neuro Fuzzy system for clustering and classification.

REFERENCES:

1. Jyh-Shing Roger Jang, Chuen-Tsai Sun, EijiMizutani, “Neuro-Fuzzy and Soft Computing”, Prentice-Hall of India, 2003
2. KwangH.Lee, “First course on Fuzzy Theory and Applications”, Springer–Verlag Berlin Heidelberg, 2005.
3. George J. Klir and Bo Yuan, “Fuzzy Sets and Fuzzy Logic-Theory and Applications”, Prentice Hall, 1995.
4. James A. Freeman and David M. Skapura, “Neural Networks Algorithms, Applications, and Programming Techniques”, Pearson Edn., 2003.
5. David E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning”, Addison Wesley, 2007.
6. Mitsuo Gen and RunweiCheng,”Genetic Algorithms and Engineering Optimization”, Wiley Publishers 2000.
7. Mitchell Melanie, “An Introduction to Genetic Algorithm”, Prentice Hall, 1998.
8. S.N.Sivanandam, S.N.Deepa, “Introduction to Genetic Algorithms”, Springer, 2007.
9. A.E. Eiben and J.E. Smith “Introduction to Evolutionary Computing” Springer, 2003
10. E. Sanchez, T. Shibata, and L. A. Zadeh, Eds., "Genetic Algorithms and Fuzzy Logic Systems: Soft Computing Perspectives, Advances in Fuzzy Systems - Applications and Theory", Vol. 7, River Edge, World Scientific, 1997.
11. ROSS TIMOTHY J, Fuzzy Logic with Engineering Applications, Wiley India Pvt Ltd, New Delhi, 2010

SE7003 MACHINE LEARNING L T P C
3 0 0 3

UNIT I INTRODUCTION 9

Learning Problems – Perspectives and Issues – Concept Learning – Version Spaces and Candidate Eliminations – Inductive bias – Decision Tree learning – Representation – Algorithm – Heuristic Space Search.

UNIT II NEURAL NETWORKS AND GENETIC ALGORITHMS 9

Neural Network Representation – Problems – Perceptrons – Multilayer Networks and Back Propagation Algorithms – Advanced Topics – Genetic Algorithms – Hypothesis Space Search – Genetic Programming – Models of Evaluation and Learning.

UNIT III BAYESIAN AND COMPUTATIONAL LEARNING 9

Bayes Theorem – Concept Learning – Maximum Likelihood – Minimum Description Length Principle –Bayes Optimal Classifier –Gibbs Algorithm – Naïve Bayes Classifier – Bayesian Belief Network – EM Algorithm – Probability Learning – Sample Complexity – Finite and Infinite Hypothesis Spaces – Mistake Bound Model.

UNIT IV INSTANT BASED LEARNING 9

K-Nearest Neighbour Learning – Locally weighted Regression – Radial Bases Functions – Case Based Learning.

UNIT V ADVANCED LEARNING **9**

Learning Sets of Rules – Sequential Covering Algorithm – Learning Rule Set – First Order Rules – Sets of First Order Rules – Induction on Inverted Deduction – Inverting Resolution – Analytical Learning – Perfect Domain Theories – Explanation Base Learning – FOCL Algorithm – Reinforcement Learning – Task – Q-Learning – Temporal Difference Learning.

TOTAL: 45 PERIODS

REFERENCES:

1. Tom M. Mitchell, “Machine Learning”, McGraw-Hill edition, 1997
2. EthemAlpaydin, “Introduction to Machine Learning (Adaptive Computation and Machine Learning)”, The MIT Press 2004
3. T. Hastie, R. Tibshirani, J. H. Friedman, “The Elements of Statistical Learning”, Springer Verlag, 2001
4. Pattern recognition and machine learning by Christopher Bishop, Springer Verlag, 2006.

CP7024 INFORMATION RETRIEVAL TECHNIQUES **L T P C**
3 0 0 3

OBJECTIVES:

- To understand the basics of Information Retrieval with pertinence to modeling, query operations and indexing
- To get an understanding of machine learning techniques for text classification and clustering
- To understand the various applications of Information Retrieval giving emphasis to Multimedia IR, Web Search
- To understand the concepts of digital libraries

UNIT I INTRODUCTION **8**

Motivation – Basic Concepts – Practical Issues - Retrieval Process – Architecture - Boolean Retrieval –Retrieval Evaluation – Open Source IR Systems–History of Web Search – Web Characteristics–The impact of the web on IR ––IR Versus Web Search–Components of a Search engine

UNIT II MODELING **10**

Taxonomy and Characterization of IR Models – Boolean Model – Vector Model - Term Weighting – Scoring and Ranking –Language Models – Set Theoretic Models - Probabilistic Models – Algebraic Models – Structured Text Retrieval Models – Models for Browsing

UNIT III INDEXING **9**

Static and Dynamic Inverted Indices – Index Construction and Index Compression. Searching - Sequential Searching and Pattern Matching. Query Operations -Query Languages – Query Processing - Relevance Feedback and Query Expansion - Automatic Local and Global Analysis – Measuring Effectiveness and Efficiency

UNIT IV CLASSIFICATION AND CLUSTERING **8**

Text Classification and Naïve Bayes – Vector Space Classification – Support vector machines and Machine learning on documents. Flat Clustering – Hierarchical Clustering –Matrix decompositions and latent semantic indexing – Fusion and Meta learning

UNIT V SEARCHING AND RANKING 10

Searching the Web –Structure of the Web –IR and web search – Static and Dynamic Ranking - Web Crawling and Indexing – Link Analysis - XML Retrieval Multimedia IR: Models and Languages – Indexing and Searching Parallel and Distributed IR – Digital Libraries

TOTAL: 45 PERIODS

OUTCOMES:

Upon completion of the course, the students will be able to

- Build an Information Retrieval system using the available tools
- Identify and design the various components of an Information Retrieval system
- Apply machine learning techniques to text classification and clustering which is used for efficient Information Retrieval
- Analyze the Web content structure
- Design an efficient search engine

REFERENCES:

1. Ricardo Baeza – Yates, BerthierRibeiro – Neto, Modern Information Retrieval: The concepts and Technology behind Search (ACM Press Books), Second Edition 2011
2. Christopher D. Manning, PrabhakarRaghavan, HinrichSchutze, Introduction to Information Retrieval, Cambridge University Press, First South Asian Edition 2012
3. Stefan Buttcher, Charles L. A. Clarke, Gordon V. Cormack, Information Retrieval Implementing and Evaluating Search Engines, The MIT Press, Cambridge, Massachusetts London, England, 2010

SE7004 SOFTWARE AGENTS L T P C
3 0 0 3

OBJECTIVES:

- To learn the principles and fundamentals of designing agents
- To study the architecture design of different agents.
- To learn to do detailed design of the agents
- To understand user interaction with agents
- To explore the role of agents in assisting the users in day to day activities

UNIT I INTRODUCTION 9

Agents and Multi Agent Systems- Intelligent Agent- Concepts of Building Agent – Situated Agents – Proactive and Reactive agents- Challenging Agent Environment- Social Agents- Agent Execution Cycle- Prometheus Methodology- Guidelines for using Prometheus- Agent Oriented Methodologies- System Specification – Goal Specification – Functionalities – Scenario Development – Interface Description – Checking for Completeness and Consistency.

UNIT II ARCHITECTURAL DESIGN 9

Agent Types - Grouping Functionalities - Agent Coupling - Develop Agent Descriptors - Interactions - Interaction Diagram from Scenarios- Interaction Protocol from Interaction Diagram- Develop Protocol and Message Descriptors –Architectural Design - Identifying the Boundaries of Agent System – Percepts and Action - Shared Data Objects – System Overview – Checking for Completeness and Consistency.

UNIT III DETAILED DESIGN 9

Capability Diagrams – Sub Tasks - Alternative Programs – Events and Messages – Action and Percept Detailed Design – Data – Develop and Refine Descriptors – Missing or Redundant Items- Consistency between Artifacts – Important Scenarios- Implementing Agent Systems - Agent Platform – JACK

UNIT IV AGENTS AND USER EXPERIENCE 9

Interact with Agents - Agents from Direct Manipulation to Delegation – Interface Agents - Designing Agents - Direct Manipulation versus Agents- Agents for Information Sharing and Coordination- Agents that Reduce Work and Information Overload - KidSim: Programming Agents without a Programming Language.

UNIT V AGENTS FOR INTELLIGENT ASSISTANCE 9

Computer Characters- Software Agents for Cooperative Learning – Integrated Agents- Agent Oriented Programming- KQML as an Agent Communication Language- Agent Based Framework for Interoperability - Agents for Information Gathering - KAoS- Communicative Actions for Artificial Agents – Mobile Agents.

TOTAL: 45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able to:

- Identify and explore the advantages of agents
- Design the architecture for an agent
- Design the agent in details in a view for the implementation
- Design communicative actions with agents.
- Design typical agents using a tool for different types of applications.

REFERENCES:

1. Lin Padgham and Michael Winikoff, “Developing Intelligent Agent Systems: A Practical Guide”, John Wiley & sons Publication, 2004.
2. Jeffrey M. Bradshaw, “Software Agents”, MIT Press , 1997.
3. Steven F. RailsBack and Volker Grimm, “Agent-Based and Individual Based modeling:A Practical Introduction”, Princeton University Press, 2012.

MP7001 XML AND WEBSERVICES L T P C
3 0 0 3

OBJECTIVES:

To provide an in-depth knowledge of XML and Web Services.

- To understand the fundamental concepts of Web services.
- To Understand the fundamental concepts of XML Technology.
- To design Web service Architecture.
- To Study Building Blocks of Web services.
- To understand the XML security issues.

UNIT I WEB FUNDAMENTALS 9

History of Web – Protocols – Web Applications - Web servers-Web Browsers-HTTP-Java Network Programming-HTML-CCS.

UNIT II XML TECHNOLOGY 9

XML-XML DTD-W3C XML Schema-Parsing XML - X path- XML Transformation-Other XML Technologies..

UNIT III ARCHITECTING WEB SERVICES 9

Business motivations for web services – B2B – B2C- Technical motivations –– Service oriented Architecture (SOA) – Architecting web services – Implementation view – web services technology stack – logical view – composition of web services – deployment view – from application server to peer to peer – process view – life in the runtime

UNIT IV WEB SERVICES BUILDING BLOCK 9

Transport protocols for web services – messaging with web services – protocols – SOAP – describing web services – WSDL – Anatomy of WSDL – manipulating WSDL – web service policy – Discovering web services – UDDI – Anatomy of UDDI

UNIT V XML SECURITY 9

Security Overview - Canonicalization - XML Security Framework - XML Encryption - XML Digital Signature - XKMS Structure - Guidelines for Signing XML Documents - XML in Practice.

TOTAL:45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able

- To Know the fundamental elements in Web Technology and XML services.
- To design the Architecture of Web Services.
- To construct building blocks of Web services.
- To analyze security in XML.

REFERENCES:

1. Uttam K.Roy , “Web Technologies”, Oxford University Press,2010
2. Ron schmelzer et al, “XML and Web Services”, Pearson Education, 2002.
3. Sandeep Chatterjee and James Webber, “Developing Enterprise Web Services: An Architect’s Guide”, Prentice Hall, 2004.
4. Frank. P. Coyle, XML, Web Services And The Data Revolution, Pearson Education, 2002
5. Keith Ballinger, “.NET Web Services Architecture and Implementation”, Pearson Education,2003
6. Henry Bequet and Meeraj Kunnumpurath, “Beginning Java Web Services”, Apress, 2004.
7. Russ Basiura and Mike Batongbacal, “Professional ASP.NET Web Services”, Apress2,2001.

SE7005 WEB ENGINEERING AND MANAGEMENT L T P C
3 0 0 3

OBJECTIVES:

- To understand web page site planning, management and maintenance.
- To know the concept of developing advanced HTML pages with the help of frames, scripting languages, and evolving technology like DHTML.
- To develop web sites which are secure and dynamic in nature and writing scripts which get executed on the servers.

UNIT I WEB ENGINEERING AND COMMUNICATION 9

Introduction – Components of web Engineering-Defining framework – Generic Actions an tasks for the WebE Framework – Umbrella activities – Communication – Formulation – Site types and architectures – Navigation Practices

UNIT II PLANNING AND MODELLING ACTIVITY 9

Scope – Refining framework activities – Developing a schedule – Modeling framework-Design goals of Web App and Quality – Types of Model in Web Apps

UNIT III CLIENT SIDE TECHNOLOGIES 8

Client side scripting: XHTML – DHTML– JavaScript– JSON - jQuery and AJAX, Setting up the environment (LAMP server)

UNIT IV SERVER SIDE TECHNOLOGIES 10

PHP, PERL, Reading Data in Web Pages - Embedding PHP within HTML - Establishing connectivity with MySQL database, Servlets, JSP, Struts Architecture, - Understanding struts, Struts Validation Framework, Understanding LAMP and Its Effect on Web Development

UNIT V PRODUCTION, MAINTENANCE AND EVALUATION 9

Web Project Method – Project Clarification - Design and Construction – Testing, Launch and Handover – Maintenance – Review and Evaluation – Case Study.

TOTAL : 45 PERIODS

OUTCOMES:

- Develop project management skills related to web development
- Understand the technical skills required for Web Developers to use W3C standards, HTML, XHTML, Style Sheets, Java
- Understand the concepts in Client and Server-Side Scripting languages such as JavaScript, PERL and PHP.

REFERENCES:

1. Roger.S. Pressman, David Lowe, “Web Engineering – A Practitioner’s Approach”, Tata McGraw Hill, Fourth reprint, 2012. (Unit – 1 &2)
2. Ashley Friedlein, “Web Project Management”, Morgan Kaufmann Publishers, 2001. (Unit -5)
3. Thomas A Powell, Fritz Schneider, “JavaScript: The Complete Reference”, Third Edition, Tata McGraw Hill, 2013 (Unit – 3)
4. H. M. Deitel, P. J. Deitel, A. B. Goldberg, “Internet and World Wide Web – How to Program”, Third Edition, Pearson Education 2004 (Unit - 3)
5. Open Source Development with LAMP: Using Linux, Apache, MySQL, Perl, and PHP, “JamesLee, Brent Ware (Unit -4)
6. Struts: The Complete Reference, "James Holmes", 2nd Edition (Unit 4)
7. Hibernate Quickly, “Patrick Peak and Nick Heudecker, Patrick Peak, Nick Heudecker" (Unit 4
8. Themas A. Powell, “The Complete Reference – Web Design”, Tata McGraw Hill, Third Edition, 2003.

NE7011 MOBILE APPLICATION DEVELOPMENT L T P C
3 0 0 3

OBJECTIVES:

1. Understand system requirements for mobile applications
2. Generate suitable design using specific mobile development frameworks
3. Generate mobile application design
4. Implement the design using specific mobile development frameworks
5. Deploy the mobile applications in marketplace for distribution

UNIT I INTRODUCTION 5

Introduction to mobile applications – Embedded systems - Market and business drivers for mobile applications – Publishing and delivery of mobile applications – Requirements gathering and validation for mobile applications

UNIT II BASIC DESIGN 8

Introduction – Basics of embedded systems design – Embedded OS - Design constraints for mobile applications, both hardware and software related – Architecting mobile applications – User interfaces for mobile applications – touch events and gestures – Achieving quality constraints – performance, usability, security, availability and modifiability.

UNIT III ADVANCED DESIGN 8

Designing applications with multimedia and web access capabilities – Integration with GPS and social media networking applications – Accessing applications hosted in a cloud computing environment – Design patterns for mobile applications.

UNIT IV TECHNOLOGY I - ANDROID 12

Introduction – Establishing the development environment – Android architecture – Activities and views – Interacting with UI – Persisting data using SQLite – Packaging and deployment – Interaction with server side applications – Using Google Maps, GPS and Wifi – Integration with social media applications.

UNIT V TECHNOLOGY II - IOS 12

Introduction to Objective C – iOS features – UI implementation – Touch frameworks – Data persistence using Core Data and SQLite – Location aware applications using Core Location and Map Kit – Integrating calendar and address book with social media application – Using Wifi - iPhone marketplace.

TOTAL : 45 PERIODS

OUTCOMES:

Upon the students will be able to Completion of the course,

1. Describe the requirements for mobile applications
2. Explain the challenges in mobile application design and development
3. Develop design for mobile applications for specific requirements
4. Implement the design using Android SDK
5. Implement the design using Objective C and iOS
6. Deploy mobile applications in Android and iPone marketplace for distribution

REFERENCES:

1. http://developer.android.com/develop/index.html
2. Jeff McWherter and Scott Gowell, "Professional Mobile Application Development", Wrox, 2012
3. Charlie Collins, Michael Galpin and Matthias Kappler, “Android in Practice”, DreamTech, 2012
4. James Dovey and Ash Furrow, “Beginning Objective C”, Apress, 2012
5. David Mark, Jack Nutting, Jeff LaMarche and Frederic Olsson, “Beginning iOS 6 Development: Exploring the iOS SDK”, Apress, 2013.

NE7012 **SOCIAL NETWORK ANALYSIS** **L T P C**
3 0 0 3

OBJECTIVES:

- To understand the components of the social network
- To model and visualize the social network
- To mine the users in the social network
- To understand the evolution of the social network
- To mine the interest of the user

UNIT I INTRODUCTION 9

Introduction to Web - Limitations of current Web – Development of Semantic Web – Emergence of the Social Web – Statistical Properties of Social Networks -Network analysis - Development of Social Network Analysis - Key concepts and measures in network analysis - Discussion networks - Blogs and online communities - Web-based networks.

UNIT II MODELING AND VISUALIZATION 9

Visualizing Online Social Networks - A Taxonomy of Visualizations - Graph Representation - Centrality- Clustering - Node-Edge Diagrams - Visualizing Social Networks with Matrix-Based Representations- Node-Link Diagrams - Hybrid Representations - Modelling and aggregating social network data **–** Random Walks and their Applications –Use of Hadoop and Map Reduce - Ontological representation of social individuals and relationships.

UNIT III MINING COMMUNITIES 9

Aggregating and reasoning with social network data, Advanced Representations - Extracting evolution of Web Community from a Series of Web Archive - Detecting Communities in Social Networks - Evaluating Communities – Core Methods for Community Detection & Mining - Applications of Community Mining Algorithms - Node Classification in Social Networks.

UNIT IV EVOLUTION 9

Evolution in Social Networks – Framework - Tracing Smoothly Evolving Communities - Models and Algorithms for Social Influence Analysis - Influence Related Statistics - Social Similarity and Influence - Influence Maximization in Viral Marketing - Algorithms and Systems for Expert Location in Social Networks - Expert Location without Graph Constraints - with Score Propagation – Expert Team Formation - Link Prediction in Social Networks - Feature based Link Prediction - Bayesian Probabilistic Models - Probabilistic Relational Models

UNIT V TEXT AND OPINION MINING 9

Text Mining in Social Networks -Opinion extraction – Sentiment classification and clustering - Temporal sentiment analysis - Irony detection in opinion mining - Wish analysis - Product review mining – Review Classification – Tracking sentiments towards topics over time.

TOTAL : 45 PERIODS

OUTCOMES:

Upon Completion of the course, the students will be able to

- Work on the internals components of the social network
- Model and visualize the social network
- Mine the behaviour of the users in the social network
- Predict the possible next outcome of the social network
- Mine the opinion of the user

REFERENCES:

1. Charu C. Aggarwal, "Social Network Data Analytics", Springer; 2011
2. Peter Mika, "Social Networks and the Semantic Web", Springer, 1st edition2007.
3. BorkoFurht, "Handbook of Social Network Technologies and Applications", Springer, 1st edition, 2010.
4. GuandongXu , Yanchun Zhang and Lin Li, "Web Mining and Social Networking – Techniques and applications", Springer, 1st edition, 2011.
5. Giles, Mark Smith, John Yen, "Advances in Social Network Mining and Analysis", Springer, 2010.
6. Ajith Abraham, Aboul Ella Hassanien, VáclavSnášel, "Computational Social Network Analysis: Trends, Tools and Research Advances", Springer, 2009.
7. Toby Segaran, "Programming Collective Intelligence", O'Reilly, 2012

SE7006 SOFTWARE RELIABILITY LT P C
3 0 0 3

UNIT I INTRODUCTION TO SOFTWARE RELIABILITY 7

Basic Concepts – Failure and Faults – Environment – Availability –Modeling –uses.

UNIT II SOFTWARE RELIABILITY MODELING 12

Concepts – General Model Characteristic – Historical Development of models – Model Classification scheme – Markovian models – General concepts – General Poisson Type Models – Binomial Type Models – Poisson Type models – Fault reduction factor for Poisson Type models.

UNIT III COMPARISON OF SOFTWARE RELIABILITY MODELS 10

Comparison Criteria – Failure Data – Comparison of Predictive Validity of Model Groups – Recommended Models – Comparison of Time Domains – Calendar Time Modeling – Limiting Resource Concept – Resource Usage model – Resource Utilization – Calendar Time Estimation and confidence Intervals.

UNIT IV ARCHITECTURE AND DESIGN TECHNIQUES 8

Fault Tolerance Techniques- Error Handling for Safe Systems – Environment Independence Techniques- Development Techniques- Components- Minimization of Faults – Fault Avoidance Techniques – Key Design Principles- Rejuvenation Techniques.

UNIT V MEASUREMENT AND TOOLS 8

Software Reliability Measurement Experience- Measurement Based Analysis of Software Reliability- Tools – CASRE – SoftRel.

TOTAL : 45 PERIODS

REFERENCES:

1. John D. Musa, Anthony Iannino, KazuhiraOkumoto, "Software Reliability – Measurement, Prediction, Application, Series in Software Engineering andTechnology", McGraw Hill, 1987.
2. John D. Musa, "Software Reliability Engineering", Tata McGraw Hill, 1999.
3. Jalote, "Fault Tolerance in Distributed Systems", Prentice Hall, 1998.
4. Michael R. Lyu, "Handbook of Software Reliability Engineering", McGraw Hill, 1996.

SE7007 **SOFTWARE DOCUMENTATION** **L T P C**
3 0 0 3

OBJECTIVES:

- To understand task orientation and process of software documentation
- to gain expertise in designing tutorials, various view types
- to understand document planning, conduct review and usability test
- to realize the software architecture and document interfaces
- to expertise in layout pages and screens graphics usage

UNIT I UNDERSTANDING TASK ORIENTATION 9

Principles of Software Documentation – Definition of Task Orientation – Theory – Forms of Software Documentation – Procedural – Reference – Process of Software of Documentation – Seven Rules for Sound Documentation.

UNIT II FORMS AND MODULE OF SOFTWARE DOCUMENTATION 9

Designing Tutorials – Elaborative Approach – Minimalist Approach – Writing to Guide – Procedure – Writing to Support – Reference – Module View type - Styles of Module View Type – Component and Connector View type – Styles of Component and Connector View type – Allocation View type and styles.

UNIT III PROCESS OF SOFTWARE DOCUMENTATION 9

Analyzing your Users – Planning and Writing your Documents – Getting Useful Reviews – Conducting Usability Tests – Editing and Fine Tuning.

UNIT IV SOFTWARE ARCHITECTURE DOCUMENTATION IN PRACTICE 9

Chunking Information: View Packets, Refinement and Descriptive Completeness – using Context Diagrams – Combined views – Documenting Variability and Dynamism – Documenting Software Interfaces.

UNIT V TOOLS OF SOFTWARE DOCUMENTATION 9

Designing for Task Orientation – Laying out Pages and Screens – Getting the Language Right – Using Graphics effectively – Designing Indexes.

TOTAL : 45 PERIODS

OUTCOMES:

- Students gain knowledge of task orientation and process of software documentation
- Students know-how to design tutorials, types of view etc.
- Students are trained to document planning and conduct review and usability test
- Students get expertise in documenting interfaces, designing layout pages, screens and graphics usage

REFERENCES:

1. Paul Clements, Felix Bachman, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, Judith Stafford, “Documenting Software Architectures: Views and Beyond”, Second Edition, Addison Wesley, 2010.
2. Thomas T.Barker, “Writing Software Documentation: A Task Oriented Approach”, Second Edition, Pearson Education, 2008.

SE7008 **SOFTWARE REFACTORING** **L T P C**
3 0 0 3

OBJECTIVES:

- To learn about Refactoring, need for refactoring and the problems with refactoring
- To gain expertise in building test, refactoring, composing methods, removing parameters
- To understand organizing data, replace, remove, preserve objects etc
- To apprehend generalization, extract and replace field, class, object , perform big refactoring
- To skill in refactoring reuse and tools for refactoring

UNIT I **9**

Refactoring, a First Example - The Starting Point - The First Step in Refactoring - Decomposing and Redistributing the Statement Method - Replacing the Conditional Logic on Price Code with Polymorphism - Principles in Refactoring- Defining Refactoring - Why Should You Refactor? - When Should You Refactor? - What Do I Tell My Manager? - Problems with Refactoring - Refactoring and Design - Refactoring and Performance - Where Did Refactoring Come From? - Bad Smells in Code - Duplicated Code - Long Method - Large Class - Long Parameter List - Divergent Change - Shotgun Surgery - Feature Envy - Data Clumps - Primitive Obsession - Switch Statements - Parallel Inheritance Hierarchies - Lazy Class - Speculative Generality - Temporary Field - Message Chains - Middle Man - Inappropriate Intimacy - Alternative Classes with Different Interfaces - Incomplete Library Class - Data Class - Refused Bequest

UNIT II **9**

Building Tests - The Value of Self-testing Code - The JUnit Testing Framework - Adding More Tests - Toward a Catalog of Refactorings - Format of the Refactorings - Finding References - How Mature Are These Refactorings? - Composing Methods - Extract Method - Inline Method - Inline Temp - Replace Temp with Query - Introduce Explaining Variable - Split Temporary Variable - Remove Assignments to Parameters - Replace Method with Method Object - Substitute Algorithm - Moving Features Between Objects - Move Method - Move Field - Extract Class - Inline Class - Hide Delegate - Remove Middle Man - Introduce Foreign Method - Introduce Local Extension

UNIT III **9**

Organizing Data - Self Encapsulate Field - Replace Data Value with Object - Change Value to Reference - Change Reference to Value - Replace Array with Object - Duplicate Observed Data - Change Unidirectional Association to Bidirectional - Change Bidirectional Association to Unidirectional - Replace Magic Number with Symbolic Constant - Encapsulate Field - Encapsulate Collection - Replace Record with Data Class - Replace Type Code with Class - Replace Type Code with Subclasses - Replace Type Code with State/Strategy - Replace Subclass with Fields - Simplifying Conditional Expressions - Decompose Conditional - Consolidate Conditional Expression - Consolidate Duplicate Conditional Fragments - Remove Control Flag - Replace Nested Conditional with Guard Clauses - Replace Conditional with Polymorphism - Introduce Null Object - Introduce Assertion - Making Method Calls Simpler - Rename Method - Add Parameter - Remove Parameter - Separate Query from Modifier - Parameterize Method - Replace Parameter with Explicit Methods - Preserve Whole Object - Replace Parameter with Method - Introduce Parameter Object - Remove Setting Method - Hide Method - Replace Constructor with Factory Method - Encapsulate Downcast - Replace Error Code with Exception - Replace Exception with Test

UNIT IV **9**

Dealing with Generalization - Pull Up Field - Pull Up Method - Pull Up Constructor Body - Push Down Method - Push Down Field - Extract Subclass - Extract Superclass - Extract Interface - Collapse Hierarchy - Form Template Method - Replace Inheritance with Delegation - Replace Delegation with Inheritance - Big Refactorings - The Nature of the Game - Why Big Refactorings Are Important - Four Big Refactorings - Tease Apart Inheritance - Convert Procedural Design to Objects - Separate Domain from Presentation - Extract Hierarchy

UNIT V **9**

Refactoring, Reuse, and Reality - Why Are Developers Reluctant to Refactor Their Programs? - Resources and References for Refactoring - Implications Regarding Software Reuse and Technology Transfer - Refactoring Tools - Refactoring with a Tool - Technical Criteria for a Refactoring Tool - Practical Criteria for a Refactoring Tool - Putting It All Together.

TOTAL : 45 PERIODS

OUTCOMES:

- Students be trained about Refactoring, need and problems with refactoring
- Students are capable of building test, composing methods and removing parameters
- Students identify and know how to organizing data, replace, remove, preserve objects
- Students able to capture about generalization, able to replace field, class, object and perform big refactoring
- Students gain proficiency in refactoring reuse and tools for refactoring

REFERENCES:

1. Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts ," Refactoring : Improving the Design of Existing Code , Addison Wesley, 2011
2. Martin Lippert, Stephen Roock,"Refactoring in Large Software Projects: Performing Complex Restructurings ", John Wiley and sons ltd, 2006
3. William C. Wake ,"Refactoring workbook", Pearson Education Inc. 2004
4. William J. Brown,"AntiPatterns: refactoring software, architectures, and projects in crisis",
5. Joshua Kerievsky," Refactoring to Patterns" ,Addison-Wesley, 2005

CP7015 MODEL CHECKING AND PROGRAM VERIFICATION **L T P C**
3 0 0 3

OBJECTIVES:

- To understand automata for model checking
- To understand LTL, CTL, and CTL*
- To understand timed automata, TCTL, and PCTL
- To understand verification of deterministic and recursive programs
- To understand verification of object-oriented programs
- To understand verification of parallel, distributed, and non-deterministic programs

UNIT I AUTOMATA AND TEMPORAL LOGICS **9**

Automata on finite words – model checking regular properties – automata on infinite words – Buchi automata – Linear Temporal Logic (LTL) – automata based LTL model checking – Computational Tree Logic (CTL) – CTL model checking – CTL* model checking

UNIT II TIMED AND PROBABILISTIC TREE LOGICS 9

Timed automata – timed computational tree logic (TCTL) – TCTL model checking – probabilistic systems – probabilistic computational tree logic (PCTL) – PCTL model checking – PCTL* - Markov decision processes

UNIT III VERIFYING DETERMINISTIC AND RECURSIVE PROGRAMS 9

Introduction to program verification – verification of "while" programs – partial and total correctness – verification of recursive programs – case study: binary search – verifying recursive programs with parameters

UNIT IV VERIFYING OBJECT-ORIENTED AND PARALLEL PROGRAMS 9

Partial and total correctness of object-oriented programs – case study: Insertion in linked lists – verification of disjoint parallel programs – verifying programs with shared variables – case study: parallel zero search – verification of synchronization – case study: the mutual exclusion problem

UNIT V VERIFYING NON-DETERMINISTIC AND DISTRIBUTED PROGRAMS 9

Introduction to non-deterministic programs – partial and total correctness of non-deterministic programs – case study: The Welfare Crook Problem – syntax and semantics of distributed programs – verification of distributed programs – case study: A Transmission Problem – introduction to fairness

TOTAL: 45 PERIODS

OUTCOMES

Upon Completion of the course, the students will be able to

- Perform model checking using LTL
- Perform model checking using CTL
- Perform model checking using CTL*
- Perform model checking using TCTL and PCTL
- Verify deterministic and recursive programs
- Verify object-oriented programs
- Verify parallel, distributed, and non-deterministic programs

REFERENCES:

1. C. Baier, J.-P. Katoen, and K. G. Larsen, "Principles of Model Checking", MIT Press, 2008.
2. E. M. Clarke, O. Grumberg, and D. A. Peled, "Model Checking", MIT Press, 1999.
3. M. Ben-Ari, "Principles of the SPIN Model Checker", Springer, 2008.
4. K. R. Apt, F. S. de Boer, E.-R. Olderog, and A. Pnueli, "Verification of Sequential and Concurrent Programs", Third Edition, Springer, 2010.
5. M. Huth and M. Ryan, "Logic in Computer Science --- Modeling and Reasoning about Systems", Second Edition, Cambridge University Press, 2004.
6. B. Berard et al., "Systems and Software Verification: Model-checking techniques and tools", Springer, 2010.
7. J. B. Almeida, M. J. Frade, J. S. Pinto, and S. M. de Sousa, "Rigorous Software Development: An Introduction to Program Verification", Springer, 2011.

CP7006 PARALLEL PROGRAMMING PARADIGMS L T P C
3 0 0 3

OBJECTIVES:

- To understand models of and issues in concurrency in computing
- To develop message-passing parallel programs using MPI
- To develop shared-memory parallel programs using Pthreads
- To develop shared-memory parallel programs using OpenMP
- To use GPU for parallel programming using OpenCL and CUDA

UNIT I FOUNDATIONS OF PARALLEL PROGRAMMING 9

Motivation for parallel programming - Concurrency in computing – basics of processes, multiprocessing, and threads – cache – cache mappings – caches and programs – virtual memory – instruction level parallelism – hardware multi-threading – SIMD – MIMD – interconnection networks – cache coherence – shared-memory model – issues in shared-memory model – distributed-memory model – issues in distributed-memory model – hybrid model – I/O – performance of parallel programs – parallel program design

UNIT II MESSAGE PASSING PARADIGM 9

Basic MPI programming – MPI_Init and MPI_Finalize – MPI communicators – SPMD programs – message passing – MPI_Send and MPI_Recv – message matching – MPI I/O – parallel I/O – collective communication – MPI_Reduce – MPI_Allreduce – broadcast – scatter – gather – allgather – derived types – remote memory access – dynamic process management – MPI for grids – performance evaluation of MPI programs

UNIT III SHARED MEMORY PARADIGM: PTHREADS 9

Basics of Pthreads – thread synchronization – critical sections – busy-waiting – mutexes – semaphores – barriers and condition variables – read-write locks – Caches, cache coherence and false sharing – thread safety – Pthreads case study

UNIT IV SHARED MEMORY PARADIGM: OPENMP 9

Basic OpenMP constructs – scope of variabls – reduction clause – parallel for directive – loops in OpenMP – scheduling loops – synchronization in OpenMP – Case Study: Producer-Consumer problem – cache issues – threads safety in OpenMP – OpenMP best practices

UNIT V GRAPHICAL PROCESSING PARADIGMS: OPENCL AND CUDA 9

Introduction to CUDA – CUDA programming examples – CUDA execution model – CUDA memory hierarchy – CUDA case study - introduction to OpenCL – OpenCL programming examples – Programs and Kernels – Buffers and Images – Event model – OpenCL case study.

TOTAL : 45 PERIODS

OUTCOMES:

Upon completion of the course, the students will be able to

- Explain models of parallel programming
- Explain hardware level support for concurrency
- Explain issues in parallel programming
- Develop message-passing parallel programs using MPI framework
- Develop shared-memory parallel programs using Pthreads
- Develop shared-memory parallel programs using OpenMP
- Develop CUDA programs
- Develop OpenCL programs

REFERENCES:

1. Peter S. Pacheco, “An introduction to parallel programming”, Morgan Kaufmann, 2011.
2. M. J. Quinn, “Parallel programming in C with MPI and OpenMP”, Tata McGraw Hill, 2003.
3. W. Gropp, E. Lusk, and R. Thakur, “Using MPI-2: Advanced features of the message passing interface”, MIT Press, 1999.
4. W. Gropp, E. Lusk, and A. Skjellum, “Using MPI: Portable parallel programming with the message passing interface”, Second Edition, MIT Press, 1999.
5. B. Chapman, G. Jost, and Ruud van der Pas, “Using OpenMP”, MIT Press, 2008.
6. D. R. Butenhof, “Programming with POSIX Threads”, Addison Wesley, 1997.
7. B. Lewis and D. J. Berg, “Multithreaded programming with Pthreads”, Sun Microsystems Press, 1998.
8. A. Munshi, B. Gaster, T. G. Mattson, J. Fung, and D. Ginsburg, “OpenCL programming guide”, Addison Wesley, 2011.
9. Rob Farber, “CUDA application design and development”, Morgan Haufmann, 2011.

SE7009 SOFTWARE PROCESS MODELS L T P C
3 0 0 3

UNIT I PROCESS AND BASIC PROCESS MODELS 9

Process Definition – Process for Software Development and Maintenance – Process Models – Waterfall – Prototypes – Throwaway – Evolutionary – Incremental.

UNIT II ADVANCED PROCESS MODELS 9

Spiral – Rapid Application Development – Unified Process Models- Agile – Extreme Programming (XP) – Adaptive Software Development (ASD) – DSDM – Scrum – Crystal – Feature Driven Development (FDD) – Comparison of Different Models.

UNIT III REUSE ORIENTED SOFTWARE ENGINEERING 9

Systematic reuse- COTS- Process stages- Benefits and Problems with reuse- Reusable components- Approaches for Software reuse- Reuse planning factors.

UNIT IV PROCESS IMPROVEMENT MODELS 9

Need for Process Improvement – ISO 9000: 2000 – SPICE- Six Sigma – CMMI.

UNIT V EMERGING TRENDS AND NEW DIRECTIONS 9

Software process simulation- Web-based software process models and process engineering- Software process and business process reengineering- Understanding, capturing, and operationalizing process models.

TOTAL: 45 PERIODS

REFERENCES:

1. PankajJalote, “An Integrated Approach to Software Engineering”, Second Edition, Springer Verlag, 1997.
2. Roger S. Pressman, “Software Engineering: A Practitioner’s Approach”, Fifth Edition, McGraw Hill, 2001.
3. Ian Sommerville, “Software Engineering”, Sixth Edition, Addison Wesley, 2000.
4. Kent Beck, “eXtreme Programming explained: EMBRACE CHANGE”, First Edition, Pearson Education Asia, 1999.
5. Philippe Kruchten, “The Rational Unified Process, an introduction”, Second Edition, Addison Wesley, 2000.
6. Humphrey Watts S, “Managing the Software Process”, Addison Wesley, 1989.
7. Alan C. Gillies, “Software Quality - Theory and Management”, Second Edition,
8. International Thomson Computer Press, 1999.
9. David Hoyle, “ISO 9000 Quality Systems Handbook”, Fourth Edition, Butterworth – Heinemann, 2001.
10. Peter S. Pande, Larry Holpp, Pete Pande, Lawrence Holpp,” What Is Six Sigma?” McGraw-Hill Trade, 2001.
11. John J. Marciniak, “Encyclopedia of Software Engineering, Wiley-Interscience, 2001.