JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR, ANANTHAPURAMU

COURSE STRUCTURE FOR DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

M.Tech-CSE- Software Engineering

w.e.f. 2017-18 Admitted Batch onwards

M.Tech I Semester

S.No	Subject Code	Subject	L	T	P	C
1.	17D25101	Object Oriented Software Engineering	4	-	-	4
2.	17D25102	Fundamentals of Data Science	4	-	-	4
3.	17D25103	Software Requirements & Estimations	4	-	-	4
4.	 17D25104 17D25105 17D25106 17D25107	Elective-I a. Agile Methodologies b. Reverse Engineering c. Service Oriented Architecture d. Professional Aspects in Software Engineering	4	-	-	4
5.	 17D25108 17D25109 17D25110 17D25111	Elective-II a. Formal Methods of Software Engineering b. Software Metrics & Reuse c. Component Based Software Engineering d. Protocol Software Engineering	4	-	-	4
6.	17D25112	Object Oriented Software Engineering Lab	-		4	2
7.	17D25113	Data Science Lab	-		4	2
8.	17D25114	Software Requirements & Estimations Lab	-		4	2
		Total	**20**		**12**	**26**

M.Tech II Semester

S.No	Subject Code	Subject	L	T	P	C
1.	17D25201	Advances in Software Testing	4	-	-	4
2.	17D25202	Model Driven Software Development	4	-	-	4
3.	17D25203	Software Patterns	4	-	-	4
4.	 17D25204 17D25205 17D25206 17D25207	Elective-III a. Software Quality Assurance b. Software Reliability c. Internet of Things d. Software Project Management	4	-	-	4
5.	 17D25208 17D25209 17D25210 17D25211	Elective-IV a. Big Data Analytics b. Software Reengineering c. Software Configuration Management d. Secure Software Engineering	4	-	-	4
6.	17D25212	Advances in Testing Lab	-	-	4	2
7.	17D25213	Model Driven Software Development Lab	-	-	4	2
8.	17D25214	Software Patterns Lab	-	-	4	2
		Total	**20**		**12**	**26**

M.Tech III Semester

S.No	Subject Code	Subject	L	T	P	C
1.	17D20301 17D20302 17D20303	Elective-V (Open Elective) 1. Research Methodology 2.Human Values & Professional Ethics 3.Intellectual Property Rights	4	-	-	4
2.	17D25301	Elective-VI (MOOCs)	-	-	-	-
3.	17D25302	Comprehensive Viva-Voice	-	-	-	2
4.	17D25303	Seminar	-	-	-	2
5.	17D25304	Teaching Assignment	-	-	-	2
6.	17D25305	Project work Phase-I	-	-	-	4
		Total	**04**	**-**	**-**	**14**

M.Tech IV Semester

S.No.	Subject Code	Subject	L	T	P	C
1.	17D25401	Project work Phase - II	-		-	12
		Total	**-**		**-**	**12**

Project Viva Voce Grades:

A: Satisfactory
B: Not Satisfactory

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25101) OBJECT ORIENTED SOFTWARE ENGINEERING

UNIT I

Introduction to Software Engineering: Introduction-Software Engineering Failures, What is Software Engineering?, Software Engineering Concepts, Software Engineering Development Activities, Managing Software Development. Modeling with UML: Introduction, Overview of UML, Modeling Concepts, ARENA Case Study

UNIT II

Analysis: Introduction –An Optical Illusion, Overview of Analysis, Analysis Concepts, Analysis Activities, Managing Analysis, ARENA Case Study

UNIT III

System Design: Decomposing the System-Introduction: A Floor Plan Example, Overview of System Design, System Design Concepts, System Design Activities: From Objects to Subsystems

System Design: Addressing Design Goals: Introduction- A Redundancy Example, Overview of System Design Activities, Concepts: UML Deployment Diagrams, Addressing Design Goals, Managing System Design, ARENA Case Study

UNIT IV

Object Design: Specifying Interfaces: Introduction - A Railroad Example, Overview of Interface Specification, Interface Specification Concepts, Interface Specification Activities, Managing Object Design, ARENA Case Study

Mapping Models to Code: Introduction – A Book Example, Overview of Mapping, Mapping Concepts, Mapping Activities, Managing Implementation, ARENA Case Study

UNIT V

Testing: Introduction – Testing the Space Shuttle, Overview of Testing, Testing Concepts, Testing Activities, Managing Testing

TEXT BOOKS

1. Bernd Bruegge & Allen H. Dutoit, “Object-Oriented Software Engineering”, 2009.

2. Ivar Jacobson, “Object-Oriented Software Engineering”, Pearson Education, 2009.

REFERENCES

1. Stephen R. Schach, “Object-Oriented Classical Software Engineering”, Mc Graw Hill, 2010.

2. Yogesh Singh, “Object-Oriented Software Engineering”, 2012

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25102) FUNDAMENTALS OF DATA SCIENCE

UNIT - I

Introduction, What Is Statistical Learning?, Why Estimate f?, How Do We Estimate f?, The Trade-Off Between Prediction Accuracy and Model Interpretability, Supervised Versus Unsupervised Learning, Regression Versus Classification Problems, Assessing Model Accuracy, Measuring the Quality of Fit, The Bias-Variance Trade-of, The Classification Setting, Introduction to R, Basic Commands, Graphics, Indexing Data, Loading Data, Additional Graphical and Numerical Summaries.

UNIT – II

Linear Regression, Simple Linear Regression, Multiple Linear Regression, Other Considerations in the Regression Model, Comparison of Linear Regression with K-Nearest Neighbours, Linear Regression.

UNIT-III

Classification, Logistic Regression, Linear Discriminant Analysis, A Comparison of Classification Methods, Logistic Regression, LDA, QDA, and KNN.

UNIT- IV

Programming for basic computational methods such as Eigen values and Eigen vectors, sparse matrices, QR and SVD, Interpolation by divided differences.

Data Wrangling: Data Acquisition, Data Formats, Imputation, The split-apply-combine paradigm.

UNIT-V

Data Objects and Attribute Types, Basic Statistical Descriptions of Data, Data Visualization, Measuring Data Similarity and Dissimilarity.

Data Warehouse: Basic Concepts, Data Warehouse Modeling: Data Cube and OLAP, Data Warehouse Design and Usage, Data Warehouse Implementation, Data Generalization by Attribute-Oriented Induction.

Text Books:

1. Gareth James Daniela Witten Trevor Hastie, Robert Tibshirani, An Introduction to Statistical Learning with Applications in R, February 11, 2013, web link: www.statlearning.com.
2. Mark Gardener, Beginning R The statistical Programming Language, Wiley, 2015.
3. Han , Kamber, and J Pei, Data Mining Concepts and Techniques, 3rd edition, Morgan Kaufman, 2012.

References:

1. Sinan Ozdemir, Principles of Data Science, Packt Publishing Ltd Dec 2016.
2. Joel Grus, Data Science from Scratch, Oreilly media, 2015.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25103) SOFTWARE REQUIREMENTS AND ESTIMATION

UNIT-I :

Software Requirements: What And Why

Essential Software requirement, Good practices for requirements engineering, Improving requirements processes, Software requirements and risk management.

UNIT II:

Software Requirements Engineering

Requirements elicitation, requirements analysis documentation, review, elicitation techniques, analysis models, Software quality attributes, risk reduction through prototyping, setting requirements priorities, verifying requirements quality.

UNIT-III:

Software Requirements Modeling: Use Case Modeling, Analysis Models, Dataflow diagram, state transition diagram, class diagrams, Object analysis, Problem Frames.

Software Requirements Management: Requirements management Principles and practices, Requirements attributes, Change Management Process, Requirements Traceability Matrix, Links in requirements chain.

UNIT - IV

Software Estimation: Components of Software Estimations, Estimation methods, Problems associated with estimation, Key project factors that influence estimation.

Size Estimation: Two views of sizing, Function Point Analysis, Mark II FPA, Full Function Points, LOC Estimation, Conversion between size measures.

Effort, Schedule and Cost Estimation: What is Productivity? Estimation Factors, Approaches to Effort and Schedule Estimation, COCOMO II, Putnam Estimation Model, Algorithmic models, Cost Estimation.

UNIT-V

Requirements Management Tools: commercial requirements management requirements management automation.

Benefits of using a requirements management tool, tool, Rational Requisite pro, Caliber – RM, implementing

Software Estimation Tools: Desirable features in software estimation tools, IFPUG, USC's COCOMO II, SLIM (Software Life Cycle Management) Tools.

TEXT BOOKS:

1. Software Requirements by Karl E. Weigers,Microsoft Press.

2. Software Requirements and Estimation by *Rajesh Naik and Swapna Kishore*, Tata Mc Graw Hill.

REFERENCES:

1. Managing Software Requirements, Dean Leffingwell & Don Widrig, Pearson Education,2003.

2. Mastering the requirements process, second edition, Suzanne Robertson & James Robertson, Pearson Education, 2006.

3. Estimating Software Costs, Second edition, Capers Jones, Tata McGraw-Hill, 2007.

4. Practical Software Estimation, M.A. Parthasarathy, Pearson Education, 2007.

5. Measuring the software process, William A. Florac & Anita D. Carleton, Pearson Education,1999.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25104) AGILE METHODOLOGIES
(ELECTIVE –I)

UNIT I

Why Agile? , How to be Agile, Understanding XP, Values and Principles, Improve the Process, Eliminate Waste, Deliver Value.

UNIT II

Practicing XP-Thinking, Pair Programming, Energized Work, Informative Workspace, Root-Cause Analysis, Retrospectives, Collaborating, Sit Together, Real Customer Involvement, Ubiquitous Language, Stand-Up Meetings, Coding Standards, Iteration Demo, Reporting.

UNIT III

Releasing-Done Done, No Bugs, Version Control, Ten-Minute Build, Continuous Integration, Collective Code Ownership, Documentation.

UNIT IV

Planning-Vision, Release Planning, Risk Management, Iteration Planning, Stories, Estimating.

UNIT V

Developing-Incremental Requirements, Customer Tests, Test- Driven Development, Refactoring, Incremental Design and Architecture, Spike Solutions, Performance Optimization.

Text Books:

1. James Shore and Shane Warden, " The Art of Agile Development", O'REILLY, 2007.

References:

1. Robert C. Martin, "Agile Software Development, Principles, Patterns, and Practices" , PHI, 2002.

2. Angel Medinilla, "Agile Management: Leadership in an Agile Environment", Springer, 2012.

3. Bhuvan Unhelkar, "The Art of Agile Practice: A Composite Approach for Projects and Organizations", CRC Press.
4. Jim Highsmith, "Agile Project Management", Pearson education, 2004.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25105) REVERSE ENGINEERING
(ELECTIVE –I)

UNIT I

Foundations: What is Reverse Engineering, Software Reverse Engineering, Reverse Applications, Low Level Software, The Reversing Process, The Tools, Is Reversing Legal, Code Samples & Tools.

Object Flow Graph: Abstract Language, Object Flow Graph, Containers, Flow Propagation Algorithm, Object Sensitivity, The elib Program.

Low Level Software: High Level Perspectives, Low Level Perspectives, Assembly Language, A Primer on Compilers and Compilation, Execution Environments.

UNIT II

Reversing Tools: Different Reversing Approaches, Disassemblers, Debuggers, Decompilers, System-Monitoring Tools, Patching Tools, Miscellaneous Reversing Tools.

UNIT III

Beyond the Documentation: Reversing and Interoperability, Laying The Ground Rules, Locating Undocumented APIs, Case Study.

UNIT IV

Class Diagram: Class Diagram Recovery, Declared Vs Actual Types, Containers, The elib Program.

Object Diagram: The Object Diagram, Object Sensitivity, Dynamic Analysis, The elib Program. **Interaction Diagram:** Interaction Diagram, Interaction Diagram, Intreraction Diagram Recovery, Dynamic Analysis, The elib Program.

State Diagram: State Diagram, Abstract Interpretation, State Diagram Recovery, The elib Program.

UNIT V

Package Diagram : Package Diagram Recovery, Clustering, Concept Analysis, The elib Program, Tool Architecture, The elib Program, Perspectives.

Reversing Malware : Types of malware, Sticky software, Future malware, Uses of malware, Malware vulnerability, Polymorphism, Metamorphism, Establishing a secure environment. **Antireversing Techniques :** Why anti reversing?, Basic approaches to anti reversing, Eliminating symbolic information, Code encryption, Active anti debugger techniques, Confusing Disassemblers, Code obfuscation, Control flow transformations, Data transformations.

TEXT BOOKS:

1. Reverse Engineering of Object Oriented Code Paolo Tonella by Alessandra Potrich.
2. Reversing: Secrets of Reverse Engineering by Eldad Eilam.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25106) SERVICE ORIENTED ARCHITECTURE

(Elective-I)

UNIT I

Introduction to SOA

Fundamental SOA; Common Characteristics of contemporary SOA; Common tangible benefits of SOA; An SOA timeline (from XML to Web services to SOA); The continuing evolution of SOA (Standards organizations and Contributing vendors)

UNIT II

Web Services and Primitive SOA

The Web services framework; Services (as Web services); Service descriptions (with WSDL); Messaging (with SOAP).

Web Services and Contemporary SOA

Message exchange patterns; Service activity; Coordination; Atomic Transactions; Business activities; Orchestration; Choreography.

UNIT III

Principles of Service Orientation

Services-orientation and the enterprise; Anatomy of a service-oriented architecture; Common Principles of Service-orientation; How service orientation principles inter-relate; Service-orientation and object-orientation; Native Web service support for service-orientation principles.

UNIT IV

Service Layers- Abstraction, Business and Orchestration Service Layers.

Business Process Design: WS-BPEL language basics; WS-Coordination overview; Service-oriented business process design; WS-addressing language basics; WS-Reliable Messaging language basics.

UNIT V

SOA Platforms: SOA platform basics; SOA support in J2EE; SOA support in .NET; Integration considerations. Amazon web services as an example.

Text Books:

1. Thomas Erl, "Service-Oriented Architecture – Concepts, Technology, and Design", Pearson Education, 2005.

2. Eric Newcomer, Greg Lomow,"Understanding SOA with Web Services", Pearson Education,2005.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25107) PROFESSIONAL ASPECTS IN SOFTWARE ENGINEERING

(Elective-I)

UNIT-I:

Intellectual Property rights Confidential Information, Copyright, Infringement of Copyright, Acts permitted in Relation to Copyright Works, Licensing and Assignment of Copyright, Moral Rights, Designs, Trademarks, The tort of passing off, Domain Names, Patents.

UNIT-II:

Software Licenses, Copyright, Contract, Patent, Free Software and Open Source Software, MIT License, BSD, License, GNU General Public License, GNU Lesser General Public License, Q Public License, Proprietary License, Sun Community License.

UNIT-III:

Software Contracts:

Basics of Software Contracts, Extent of liability, Contract for the supply of custom-built software at a fixed price, other types of software service Contract, Liability for defective software.

UNIT-IV:

Software Crime Prevention

Computing and criminal Activity, Reforms of Criminal Law, Categories of Misuse, Computer Fraud, Obtaining Unauthorized Access to Computer, Unauthorized Alteration or Destruction of Information, Denying Access to an Authorized user, Unauthorized Removal of Information Stored in a Computer.

UNIT-V:

Data Protection Regulations, Data Protection and Privacy, The impact of the Internet, Factors Influencing the Regulation of Data Processing, Convergence of Data Protection Practice, Defamation and the protection of Reputation.

REFERENCES:

1. Andrew M. St. Laurent, “Open Source and Free Software Licensing”, O’Reilly, Publications.

2. Frank Bott, et. al, “Professional Issues in Software Engineering”, Taylor &

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25108) FORMAL METHODS OF SOFTWARE ENGINEERING
(Elective-II)

UNIT I

Introduction: Formal methods, The CICS Experience, The Z notation, The importance of Proof, Abstacion.

Propositional Logic: Proportional logic, Conjunction, Disjunction, Implication, Equivalence, Negation, Tautologies and Contradictions.

Predicate Logic: Predicate calculus, Quantifiers and declarations, Substitution, Universal Introduction and elimination, Existential introduction and elimination, Satisfaction and validity.

Equality and Definite Description: Equality, The one-point rule, Uniqueness and quantity, Definite description.

UNIT II

Sets: Membership and extension, Set comprehension, Power sets, Cartesian products, Union, intersection, and difference, Types.

Definitions: Declarations, Abbreviations, Generic abbreviations, Axiomatic definitions, Generic definitions, Sets and predicates.

Relations: Binary relations, Domain and range, Relational inverse, Relational composition, Closures.

Functions: Partial functions, Lambda notation, Functions on relations, Overriding, Properties of functions, Finite sets.

UNIT III

Sequences: Sequence notation, A model for sequences, Functions on sequences, Structural induction, Bags.

Free Types: The natural numbers, Free type definitions, Proof by induction, Primitive recursion, Consistency.

Schemas: The schema, Schemas as types, Schemas as declarations, Schemas as predicates, Renaming, Generic schemas.

Schema Operators: Conjunction, Decoration, Disjunction, Negation, Quantification and hiding, Composition.

UNIT IV

Promotion: Factoring operations, Promotion, Free and constrained promotion.

Preconditions: The initialisation theorem, Precondition investigation, Calculation and simplification, Structure and preconditions.

A File System: A Programming interface, Operations upon files, A more complete description, A file system, Formal analysis.

Data Refinement: Refinement, Relations and nondeterminism, Data types and data refinement, Simulations, Relaxing and unwinding.

UNIT V

Data Refinement and Schemas: Relations and schema operations, Forwards simulation, Backwards simulation.

Functional Refinement: Retrieve functions, Functional refinement, Calculating data refinements, Refining promotion.

Refinement Calculus : The specification statement, Assignment, Logical constants, Sequence composition, Conditional statements, Iteration.

Text Book:

1. Jim **Woodcock and Jim Davies, "Using Z:** Specification, Refinement, and Proof", Prentice Hall (ISBN 0-13-948472-8), 1996.

Reference Books:

1. Diller, *Z An Introduction to Formal Methods* (2nd ed.), Wiley, 1994.

2. J. M. Spivey, "The Z Notation: A Reference Manual", Second Edition, Prentice Hall, 1992.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25109) SOFTWARE METRICS AND REUSE
(Elective-II)

UNIT - I

Basics of measurement: Measurement in everyday life, measurement in software engineering, scope of software metrics, representational theory of measurement, measurement and models, measurement scales, meaningfulness in measurement, goal-based framework for software measurement, classifying software measures, determining what to measure, software measurement validation.

UNIT - II

Empirical investigation: types of investigation, planning and conducting investigations.

Software-metrics data collection and analysis: What is good data, how to define the data, how to collect the data, how to store and extract data, analyzing software-measurement data, frequency distributions, various statistical techniques.

Measuring internal product attributes: Measuring size, aspects of software size, length, functionality and complexity, measuring structure, types of structural measures, control-flow structure, modularity and information flow attributes, data structures.

UNIT - III

Measuring external product attributes: Modeling software quality, measuring aspects of software quality.

Metrics for object-oriented systems: The intent of object-oriented metrics, distinguishing characteristics of object-oriented metrics, various object-oriented metric suites – LK suite, CK suite and MOOD metrics.

Metrics for component-based systems: The intent of component-based metrics, distinguishing characteristics of component-based metrics, various component-based metrics.

UNIT - IV

Introduction: Software Reuse and Software Engineering, Concepts and Terms, Software Reuse products, Software Reuse processes, Software Reuse paradigms. State of the Art and the Practice: Software Reuse Management, Software Reuse Techniques, Aspects of Software Reuse, Organizational Aspects, Technical Aspects and Economic Aspects.

Programming Paradigm and Reusability: Usability Attributes, Representation and Modeling Paradigms, Abstraction and Composition in development paradigm.

UNIT - V

Object-Oriented Domain Engineering: Abstraction and Parameterization Techniques, Composition Techniques in Object Orientation.

Application Engineering: Component Storage and Retrieval, Reusable Asset Integration. **Software Reuse Technologies:** Component Based Software Engineering, COTS based development, Software Reuse Metrics, Tools for Reusability.

Text books:

1. Norman E. Fenton and Shari Lawrence Pfleeger; Software Metrics – A Rigorous and Practical Approach, Thomson Asia Pte., Ltd, Singapore.

2. Stephen H. Kan; Metrics and Models in Software Quality Engineering, Addison Wesley, New York.

3. Reuse Based Software Engineering Techniques, Organization and Measurement by Hafedh Mili, Ali Mili, Sherif Yacoub and Edward Addy, John Wiley & Sons Inc

4. The Three Rs of Software Automation: Re-engineering, Repository, Reusability by Carma McClure, Prentice Hall New Jersey

References:

1. K. H. Möller and D. J. Paulish; Software Metrics - A Practitioner's Guide to Improved Product Development, Chapman and Hall, London.

2. Mark Lorenz and Jeff Kidd; Object-Oriented Software Metrics, Prentice Hall, New York.

3. McClure, Carma L. Software reuse techniques : adding reuse to the system development process / : Prentice Hall

4. Poulin, Jeffrey S. Measuring software reuse : principles, practices, and economic models / Jeffrey S. Poulin. Reading, Mass. : Addison-Wesley

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25110) COMPONENT BASED SOFTWARE ENGINEERING
(Elective-II)

UNIT I

Component definition - Definition of a Software Component and its elements, The Component Industry Metaphor, Component Models and Component Services, An example specification for implementing a temperature regulator Software Component.

The Case for Components- The Business Case for components, COTS Myths and Other Lessons Learned in Component-Based Software Development.

UNIT II

Planning Team Roles for CBD, Common High-Risk Mistakes, CBSE Success Factors: Integrating Architecture, Process, and Organization.

Software Engineering Practices - Practices of Software Engineering, From Subroutines to

Subsystems: Component-Based Software Development, Status of CBSE in Europe.

UNIT III

The Design of Software Component Infrastructures - Software Components and the UML, Component Infrastructures, Business Components, Components and Connectors, An OPEN process for CBD, Designing Models of Modularity and Integration. Software Architecture, Software Architecture Design Principles, Product-Line Architectures.

UNIT IV

The Management of Component-Based Software Systems - Measurement and Metrics for Software Components, Implementing a Practical Reuse Program for Software Components, Selecting the Right COTS Software, Building instead of Buying, Software Component Project Management, The Trouble with Testing Components, Configuration Management and Component Libraries, The Evolution, Maintenance, and Management of CBS.

UNIT V

Component Technologies - Overview of the CORBA Component Model, Overview of COM+, Overview of the EJB Component Model, Bonobo and Free Software GNOME Components, Choosing between COM+, EJB, and CCM, Software Agents as Next Generation Software Components.

TEXT BOOKS:

1. Component - Based Software Engineering, G.T. Heineman and W.T. Councill, Addison-Wesley, Pearson Education.

REFERENCE BOOKS:

1. Component Software, C.Szyperski, D.Gruntz and S.Murer, Pearson Education.

2. Software Engineering, Roger S. Pressman, 6th edition, Tata McGraw-Hill.

3. Software Engineering, Ian Sommerville, seventh edition, Pearson education, 2004.

4. Software Engineering Principles and Practice, Hans Van Vliet, 3rd edition, Wiley India edition.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
4	0	0	4

(17D25111) PROTOCOL SOFTWARE ENGINEERING
(Elective –II)

UNIT I

Network Reference Model: Layered Architecture, Network Services and Interfaces, Protocol Functions, OSI Model, TCP/IP Protocol Suite, Application Protocols.

Formal Specification: Formal Specification in the Software Process, Sub-system Interface

Specification, Behavioural Specification. Protocol Specification: Components of Protocol

to be Specified, Communication Service Specification, Protocol Entity Specification, Interface Specifications, Interactions, Multimedia Protocol Specifications, Internet Protocol Specifications.

UNIT II

Architectural Design: Architectural Design Decisions, System Organisation, Modular Decomposition Styles, Control Styles, Reference Architectures. Distributed Systems Architectures: Multiprocessor Architectures, Client-server Architectures, Distributed Object Architectures, Inter-organisational Distributed Computing.

UNIT III

Formal Description Testing for Protocol Specification, Extended State Transition Language, Language for temporal Ordering Specfication, Format and Protocol Languages.

SDL: A Protocol Specification Language: SDL, Examples of SDL Based Protocol Specifications, Other Protocol Specificaiton Languages.

Protocol Verification/Validation: Protocol Verification, Verification of a Protocol Using Finite State Machines, Protocol Validation, Protocol Design Errors, Protocol Validation Approaches, SDL Based Protocol Verification, SDL Based Protocol Validation.

UNIT IV

Protocol Conformance Testing: Conformance Testing, Conformance Testing Methodology and Framework, Conformance Test Architectures, Test Sequence Generation Methods, Distributed Architecture by Local Methods, Conformance Testing with TTCN, Conformance Testing in Systems with Semicontrollable Interfaces, Conformance Testing of RIP, Multimedia Applications Testing, SDL Based Tools for Conformance Testing, SDL Based Conformance Testing of MPLS.

UNIT V

Protocol Performance Testing: Performance Testing, SDL Based Performance Testing of TCP, SDL Based Performance Testing of OSPF, Interoperability Testing, SDL Based Interoperability Testing of CSMA/CD and CSMA/CA Protocol Using Bridge, Scalability

Testing. Protocol Synthesis: Protocol Synthesis, Interactive Synthesis Algorithm, Automatic Synthesis Algorithm, Automatic Synthesis of SDL from MSC, Protocol Resynthesis.

Testing Models, PICS and PIX IT, Abstract Test Methods, Simulation Based Evaluation of Conformance Testing Methodologies. Examples include actual implementation like OSINET, based on ESTELLE tools and TTCU, PICS, PIX IT for OSINET.

TEXT BOOKS:

1. Communication Protocol Engineering, Pallapa Venkataram, Sunilkumar S. Manvi, PHI.

2. Protocol Specification for OSI*1 , Gregor V. Bochmann, University of Motreal, Montreal, Quebec, Canada.

3. ASN.1: Communication Between Heterogeneous Systems, Olivier Dubuisson, Morgan Kaufmann.

REFERENCES:

1. Tools for Protocols Driven by Formal Specifications, Harry Rudin.

2. Network Protocols and Tools to help produce them*, Harry Rudin, IBM

Research Division, Zurich Research Laboratory, 8803 Ruschlikon, Switzerland.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
0	0	4	2

(17D25112) OBJECT ORIENTED SOFTWARE ENGINEERING LAB

1. Identifying Requirements from Problem Statements

   Requirements | Characteristics of Requirements | Categorization of Requirements | Functional Requirements | Identifying Functional Requirements | Preparing Software Requirements Specifications

2. Estimation of Project Metrics

   Project Estimation Techniques | COCOMO | Basic COCOMO Model | Intermediate COCOMO Model | Complete COCOMO Model | Advantages of COCOMO | Drawbacks of COCOMO | Halstead's Complexity Metrics.

3. Modeling UML Use Case Diagrams and Capturing Use Case Scenarios

   Use case diagrams | Actor | Use Case | Subject | Graphical Representation | Association between Actors and Use Cases | Use Case Relationships | Include Relationship | Extend Relationship | Generalization Relationship | Identifying Actors | Identifying Use cases | Guidelines for drawing Use Case diagrams

4. E-R Modeling from the Problem Statements

   Entity Relationship Model | Entity Set and Relationship Set | Attributes of Entity | Keys | Weak Entity | Entity Generalization and Specialization | Mapping Cardinalities | ER Diagram | Graphical Notations for ER Diagram | Importance of ER modeling.

5. Identifying Domain Classes from the Problem Statements

   Domain Class | Traditional Techniques for Identification of Classes | Grammatical Approach Using Nouns | Advantages | Disadvantages | Using Generalization | Using Subclasses | Steps to Identify Domain Classes from Problem Statement | Advanced Concepts.

6. Statechart and Activity Modeling

   Statechart Diagrams | Building Blocks of a Statechart Diagram | State | Transition | Action | Guidelines for drawing Statechart Diagrams | Activity Diagrams | Components of an Activity Diagram | Activity | Flow | Decision | Merge | Fork | Join | Note | Partition | A Simple Example | Guidelines for drawing an Activity Diagram

7. Modeling UML Class Diagrams and Sequence Diagrams

Structural and Behavioral Aspects | Class diagram | Class | Relationships | Sequence diagram | Elements in sequence diagram | Object | Life-line bar | Messages

8. Modeling Data Flow Diagrams

   Data Flow Diagram | Graphical notations for Data Flow Diagram | Symbols used in DFD | Context diagram and leveling DFD.

9. Estimation of Test Coverage Metrics and Structural Complexity

   Control Flow Graph | Terminologies | McCabe's Cyclomatic Complexity | Computing Cyclomatic Complexity | Optimum Value of Cyclomatic Complexity | Merits | Demerits

10. Designing Test Suites

    Software Testing | Standards for Software Test Documentation | Testing Frameworks | Need for Software Testing | Test Cases and Test Suite | Types of Software Testing | Unit Testing | Integration Testing | System Testing | Example | Some Remarks.

REFERENCES

1. Bernd Bruegge & Allen H. Dutoit, “Object-Oriented Software Engineering”, 2009.
2. Ivar Jacobson, “Object-Oriented Software Engineering”, Pearson Education, 2009.
3 Stephen R. Schach, “Object-Oriented Classical Software Engineering”, Mc Graw Hill, 2010.
4 Yogesh Singh, “Object-Oriented Software Engineering”, 2012

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	T	P	C
0	0	4	2

(17D25113) DATA SCIENCE LAB

Task1: Basic Statistics, Visualization, and Hypothesis Tests

1. Reload data sets into the R statistical package
2. Perform summary statistics on the data
3. Remove outliers from the data
4. Plot the data using R
5. Plot the data using lattice and ggplot
6. Test a hypothesis about the data

Task 2: Linear Regression

1. Use the R -Studio environment to code OLS models
2. Review the methodology to validate the model and predict the dependent variable for a set of given independent variables
3. Use R graphics functions to visualize the results generated with the model

Task 3: Logistic Regression

1. Use R -Studio environment to code Logistic Regression models
2. Review the methodology to validate the model and predict the dependent variable for a set of given independent variables
3. Use R graphics functions to visualize the results generated with the model

Task 4: Hadoop, HDFS, MapReduce and Pig Purpose

1. Run Hadoop and Hadoop fs and collect help information
2. Run a shell script to perform a word count activity

3. Run a MapReduce job to produce similar output
4. Investigate the UI for MapReduce/HDFS components to track system behavior
5. Run "Pig" statements to execute the same tasks done with MapReduce

REFERENCES

- R Commands - Quick Reference
- Surviving LINUX - Quick Reference
- Hadoop Commands
- HDFS Commands

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech I semester (SE)

L	**T**	**P**	**C**
4	**0**	**0**	**4**

(17D25114) SOFTWARE REQUIREMENTS & ESTIMATION LAB

1	Draw the Work Breakdown Structure for the system to be automated
2	Schedule all the activities and sub-activities Using the PERT/CPM charts
3	Define use cases and represent them in use-case document for all the stakeholders of the system to be automated
4	Identify and analyze all the possible risks and its risk mitigation plan for the system to be automated
5	Define Complete Project plan for the system to be automated using Microsoft Project Tool
6	Define the Features, Vision, Bussiness objectives, Bussiness rules and stakeholders in the vision document
7	Define the functional and non-functional requirements of the system to be automated by using usecases and document them in SRS document
8	Define the following tracebility matrices : 1. Usecase vs Features 2. Functional requirements Vs Usecases
9	Estimate the effort using the following the methods for the system to be automated: 1. Function point metric 2. Usecase point metric
10	Develop a tool which can be used for quantification of all the non-functional requirements

For developing software project plan Microsoft Project or its equivalents may be used

For developing SRS document Rational Requisite Pro Tool or its equivalents may be used