JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COURSE STRUCTURE AND SYLLBI FOR
M.Tech - 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.	17D58102	Fundamentals of Data Science	4	-	-	4
3.	17D25102	Software Requirements & Estimation	4	-	-	4
4.	 17D25103 17D25104 17D25105 17D25106	Elective-I a. Agile Methodologies b. Reverse Engineering c. Service Oriented Architecture d. Professional Aspects in Software Engineering	4	-	-	4
5.	 17D25107 17D25108 17D25109 17D25110	Elective-II a. Formal Methods of Software Engineering b. Software Metrics & Reuse c. Component Based Software Engineering d. Protocol Software Engineering	4	-	-	4
6.	17D25111	Object Oriented Software Engineering Lab	-		4	2
7.	17D25112	Data Science Lab	-		4	2
8.	17D25113	Software Requirements & Estimation 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 Engineering	4	-	-	4
3.	17D58103	Software Patterns	4	-	-	4
4.	 17D25203 17D25204 17D58203 17D25205	Elective-III a. Software Quality Assurance b. Software Reliability c. Internet of Things d. Software Project Management	4	-	-	4
5.	 17D58201 17D25206 17D25207 17D25208	Elective-IV a. Big Data Analytics b. Software Reengineering c. Software Configuration Management d. Secure Software Engineering	4	-	-	4
6.	17D25209	Advances in Software Testing Lab	-	-	4	2
7.	17D25210	Model Driven Software Engineering Lab	-	-	4	2
8.	17D58112	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

(17D58102) 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

(17D25102) 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

(17D25103) 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

(17D25104) 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

(17D25105) 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

(17D25106) 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

(17D25107) 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

(17D25108) SOFTWARE METRICS & 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

(17D25109) 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

(17D25110) 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

(17D25111) 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

(17D25112) 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
0	0	4	2

(17D25113) 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

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25201) ADVANCES IN SOFTWARE TESTING

Course objectives:

- Study the significance of testing
- Study the testing to be done at various levels
- Understand the procedure for designing test cases

Course Outcomes:

- Ability to systematically test the applications
- Ability to write test cases
- Ability to use testing tools effectively

UNIT I

Control flow graph – basic blocks, flow graphs, paths, basic paths, path conditions and domains, Dominators and post-dominators; Program dependence graph – data dependence, control dependence, call graph,

Tests generation - Test selection Problem, equivalence partitioning, Equivalence class partitioning, boundary value analysis and category partitioning method.

UNIT II

Finite state machines (FSM) - properties of FSM, Conformance testing, test generation, test optimization, Fault detection. **Combinatorial designs** – combinatorial test design process. **Pairwise design**: Binary factors and multi-valued factors. **Orthogonal arrays** and multi level orthogonal arrays.

UNIT III

Test Adequacy: Basics, measurement of test adequacy, infeasibility and test adequacy. Adequacy criteria based control – statement, block, conditions and decisions coverage techniques. Basics of Junit tool for Java.

Metrics

Importance of Metrics in Testing - Effectiveness of Testing – Defect Density – Defect Leakage Ratio – Residual Defect Density – Test Team Efficiency – Test Case Efficiency.

UNIT IV

Regression Testing

What is Regression Testing? Regression test process. Regression test selection techniques: Test all, Random selection, modification traversing tests, using execution trace. Test minimization and prioritization.

UNIT V

Non-functional testing

Load testing, performance testing, GUI testing, Security testing techniques and tools.

Automation: Case studies functional test automation using Selenium.

Text Books:

1. Aditya P Mathur, Foundations of software testing, 2nd edition, Pearson , 2013.
2. Boris Beizer, “Software Testing Techniques”, 2nd Edition, Dream tech press, 2003.

Reference Books:

1. M G Limaye, “Software Testing – Principles, Techniques and Tools”, Tata McGraw Hill, 2009.
2. Edward Kit, “Software Testing in the Real World - Improving the Process”, Pearson Education, 2004.
3. William E. Perry, “Effective methods for software testing”, 2nd Edition, John Wiley, 2000.

AWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25202) MODEL DRIVEN SOFTWARE ENGINEERING

Objectives:

Develop enabling technologies for supporting model driven engineering approaches to software development

Develop improved techniques and tool support for using executable specifications and model-based testing to better capture, manage and test software against its requirements

Better integrate social networking tools and techniques into the software development process to improve the efficiency of collaborative and community development of software

Better support "early phase" decision making by providing tools and techniques to assess nonfunctional requirement adherence at early stages in the software development process.

Outcomes:

- Explain the role and importance of modelling in software development
- Make and defend decisions regarding the appropriate use of modelling throughout the software development life-cycle
- Demonstrate the practical application of several general purpose modeling languages
- Design and demonstrate the practical application of domain specific modeling languages
- Integrate a set of models to form effective software specifications
- Describe concepts involved in the verification and translation of specifications
- Demonstrate the translation of specifications to form executable software

UNIT I

MDSD Basic Terminology

Goals of MDSD, MDSD Approach, Overview of MDA concepts, Architecture-Centric MDSD, Common MDSD concepts and terminology, Model-Driven Architecture, Generative Programming, Software factories, Model-Integrated computing, Language-Oriented Programming, Domain specific modeling.

UNIT II

Metamodeling

What is Metamodeling?, Metalevels vs. Level of Abstraction, MOF and UML, Extending UML, UML profiles, Metamodeling and OCL, Examples, Tool-supported Model validation, Metamodeling and Behavior, Pitfalls in Metamodeling, MDSD classification.

UNIT III

Model Transformation with QVT

History, M2M language requirements, Overall Architecture, An Example Transformation, The OMG standardization Process and Tool Availability, Assesment.

MDSD Tools:Roles, Architecture, Selection Criteria, and Pointers

Role of Tools in the Development Process, Tool Architecture and selection criteria, pointers.

The MDA Standard: Goals, Core concepts

UNIT IV

MDSD Process Building Blocks and Best Practices

Introduction, Separation between Application and domain Architecture Development, Two track Iterative Development, Target Architecture Development Process, Product-line Engineering.

Testing

Test Types, Tests in Model-driven Application Development, Testing the Domain Architecture

Versioning

What is Versioned? Projects and Dependencies, The structure of Application Projects, Version management and Build Process for mixed files, Modeling in a team and versioning of partial models

UNIT V

Quality : Quality in Model Driven Engineering

Case study: Embedded Component Infrastructures

Overview, Product-Line Engineering, Modeling, Implementation of Components, Generator Adaptation, Code Generation.

TEXT BOOKS:

1. Model-Drievn Software Development-Technology, Engineering, Management by Thomos Stahl, Markus Volter, jul 2006, John Wiley & Sons.

2. Model-Driven Software Development: Integrating Quality Assurance by Jorg Rech, Christian Bunse,2008,Information Science Publishing.

REFERENCE BOOKS :

1. Model-Driven Software Development by Sami Beydeda Matthias Book , Volker Gruhn, Springer.

2. Model Driven Systems Development with Rational Products By Brian Nolan, Barclay Brown, Dr. Laurent Balmelli, Et Al Tim Bohn, 2008,IBM.

3. Model Driven Development with Executable UML by Dragan Milicev, 2009,Wilei India pvt Ltd.

4. Model Driven Software Development by Kevin Lano, Apr 2009, Ci Business Press.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D58103) SOFTWARE PATTERNS

COURSE OBJECTIVES:

- Introduction to the fundamentals of software architecture.
- To understand various architectural patterns of software systems.
- To understand design patterns and their underlying object oriented concepts.
- Software architecture and quality requirements of a software system
- Identifying the appropriate patterns for design problems.
- To understand design patterns and their underlying object oriented concepts.
- To understand implementation of design patterns and providing solutions to real world software design problems.
- To understand patterns with each other and understanding the consequences of combining patterns on the overall quality of a system.

COURSE OUTCOMES:

The student will be able to:

- Design and motivate software architecture for large scale software systems
- Recognize major software architectural patterns, design patterns, and frameworks
- Know the underlying object oriented principles of design patterns.
- Understand the context in which the pattern can be applied.
- Understand how the application of a pattern affects the system quality and its tradeoffs.

UNIT I

Envisioning Architecture - What is Software Architecture, Architectural patterns, reference models, reference architectures, architectural structures and views and the Architecture Business Cycle.

Creating an Architecture - Quality Attributes, Achieving qualities, designing the Architecture, Documenting software architectures, Reconstructing Software Architecture.

UNIT II

Introduction to Patterns - What is a Pattern? What makes a Pattern? Pattern Categories, Relationships between Patterns, Pattern Description, Patterns and Software Architecture.

Architectural Patterns

Layers, Pipes and Filters, Blackboard, Broker, Microkernel, MVC, PAC, Reflection.

UNIT III

What is Design Pattern, Organizing catalogs, Role in solving design problems, Selection and Usage, **Creational Patterns** - Abstract factory, builder, factory method, prototype, singleton,

UNIT IV

Structural Patterns - Adapter, bridge, composite, decorator, façade, flyweight, Proxy, Decorator, façade, flyweight, Proxy.

UNIT V

Behavioral Patterns - Chain of responsibility, command, Interpreter, iterator, mediator, memento, observer, state, strategy, template method, and visitor.

Case Studies – Designing a Document Editor - Design issues of Lexi Editor in Deign Patterns, The World Wide Web - a case study in interoperability

TEXT BOOKS:

1. Software Architecture in Practice, second edition, Len Bass, Paul Clements & Rick Kazman, Pearson Education,2003.
2. Pattern-Oriented Software Architecture", A System of Patterns, Frank Buschmann Regine Meunier, Hans Rohnert, Peter Sommerlad and Michael Stal, WlLEY.
3. Design Patterns: Elements of Reusable Object-Oriented Software, Erich Gamma, Pearson Education.

REFERENCE BOOKS:

1. AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, by William J. Brown, Raphael C. Malveau, Hays W. "Skip" McCormick , Thomas J. Mowbray (Author) 1st Edition,
2. Java testing patterns, John Thomas etc, wiley.
3. Software architecture, David M. Dikel, David Kane and James R. Wilson, Prentice Hall PTR,2001
4. Head First Design patterns, Eric Freeman & Elisabeth Freeman, O'REILLY, 2007.
5. Design Patterns in Java, Steven John Metsker & William C. Wake, Pearson education, 2006

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25203) SOFTWARE QUALITY ASSURANCE

Elective-III

Course Objectives:.

The student should be able to:

Understand quality assurance as a fundamental component of software

Define the scope of quality assurance projects

Estimate cost of a quality assurance project and manage budgets
Prepare schedules for a quality assurance project
Develop testing & quality assurance project staffing requirements
Effectively manage a testing & quality assurance project

Course Outcomes:

1. Critically evaluate different software development environments and contexts with respect to the application of appropriate standards and models,
2. Critically evaluate leading edge approaches in software development and attendant quality assurance methodologies, presenting the research using Harvard referencing.
3. Understand and apply key quality assurance techniques tailored for specific software development environments.

UNIT I

FUNDAMENTALS OF SOFTWARE QUALITY ASSURANCE

The Role of SQA – SQA Plan – SQA considerations – SQA people – Quality Management – Software Configuration Management

UNIT II

MANAGING SOFTWARE QUALITY

Managing Software Organizations – Managing Software Quality – Defect Prevention – Software Quality Assurance Management

UNIT III

SOFTWARE QUALITY ASSURANCE METRICS

Software Quality – Total Quality Management (TQM) – Quality Metrics – Software Quality Metrics Analysis

UNIT IV

SOFTWARE QUALITY PROGRAM

Software Quality Program Concepts – Establishment of a Software Quality Program – Software Quality Assurance Planning – An Overview – Purpose & Scope.

UNIT V

SOFTWARE QUALITY ASSURANCE STANDARDIZATION

Software Standards–ISO 9000 Quality System Standards - Capability Maturity Model and the Role of SQA in Software Development Maturity – SEI CMM Level 5 – Comparison of ISO 9000 Model with SEI's CMM

TEXT BOOKS:

1. Mordechai Ben-Menachem / Garry S Marliss, "Software Quality", Vikas Publishing House, Pvt, Ltd., New Delhi.(UNIT III to V)

2. Watts S Humphrey, " Managing the Software Process", Pearson Education Inc.(UNIT I and II)

REFERENCES:

1. Gordon G Schulmeyer, "Handbook of Software Quality Assurance", Third Edition, Artech House Publishers 2007

2. Nina S Godbole, "Software Quality Assurance: Principles and Practice", Alpha Science International, Ltd, 2004.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25204) SOFTWARE RELIABILITY

Elective-III

Objectives:

To discuss the problems of reliability specification and measurement
To introduce reliability metrics and to discuss their use in reliability specification
To show how reliability predications may be made from statistical test results.

Course Outcome;

Master attributes and assessment of quality, reliability and security of software
To describe the statistical testing process

UNIT I:

Introduction: The Need for Reliable Software, Software Reliability Engineering Concepts, Basic definitions, Software practitioners biggest problem, software reliability engineering approach, software reliability engineering process, defining the product.

The Operational Profile: Reliability concepts, software reliability and hardware reliability, developing operational profiles, applying operational profiles, learning operations and run concepts.

UNIT II:

Software Reliability Concepts: Defining failure for the product, common measure for all associated systems, setting system failure intensity objectives, determining develop software failure intensity objectives, software reliability strategies, failures, faults and errors, availability, system and component reliabilities and failure intensities, predicting basic failure intensity.

UNIT III:

Software Reliability Modeling Survey: Introduction, Historical Perspective and Implementation, Exponential Failure Time Class of Models, Weibull and Gamma Failure Time Class of Models, Infinite Failure Category Models, Bayesian Models, Model Relationship, Software Reliability Prediction in Early Phases of the Life Cycle.

UNIT IV:

Software Metrics for Reliability Assessment: Introduction, Static Program Complexity, Dynamic Program Complexity, Software Complexity and Software Quality, Software Reliability Modeling.

Software Testing and Reliability: Introduction, Overview of Software Testing, Operational profiles, Time/Structure Based Software Reliability Estimation.

UNIT V:

Best Practice of SRE: Benefits and approaches of SRE, SRE during requirements phase, SRE during implementation phase, SRE during Maintenance phase.

Neural Networks for Software Reliability: Introduction, Neural Networks, Neural Networks for software reliability, software reliability growth modeling.

Text Books

1. Handbook of Software Reliability Engineering Edited by Michael R. Lyu, published by IEEE Computer Society Press and McGraw-Hill Book Company.
2. Software Reliability Engineering John D. Musa, second edition Tata McGraw-Hill.

Reference Books

1. Practical Reliability Engineering, Patric D. T. O connor 4th Edition, John Wesley & Sons, 2003.
2. Fault tolerance principles and Practice, Anderson and PA Lee, PHI, 1981.
3. Fault tolerant computing-Theory and Techniques, Pradhan D K (Ed.): Vol 1 and Vol 2, Prentice hall, 1986.
4. Reliability Engineering E. Balagurusamy, Tata McGrawHill, 1994.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D58203) INTERNET OF THINGS

Elective-III

Course Objectives:

- Students will be explored to the interconnection and integration of the physical world and the cyber space. They are also able to design & develop IoT Devices

Course Outcomes:

- Able to understand the application areas of IoT
- Able to realize the revolution of Internet in Mobile Devices, Cloud & Sensor Networks
- Able to understand building blocks of Internet of Things and characteristics.

UNIT 1

Introduction - Internet of Things – **Design Principles for Connected Devices** – Web Thinking for Connected Devices – **Internet Principles** – IP – TCP – IP Protocol Suite – UDP – IP Address – MAC Address – TCP and UDP Ports – Application Layer Protocols.

UNIT 2

Prototyping – Prototypes and Production – Cloud – Open Source vs Closed Source – Tapping into the Community **– Prototyping Embedded Devices** – Electronics – Embedded Computing Basics – Ardunio – Raspberry Pi – Beagle Bone Black – Electronic Imp.

UNIT 3

Prototyping thePhysicalDesign – Laser Cutting – 3D Printing – CNC Milling – Repurposing and Recycling – **Prototyping Online Components** – New API – Real Time Reactions – Other Protocols.

UNIT 4

Techniques for writing Embedded Code – Memory Management – Performance and Battery life – Libraries – Debugging – **Business Models** – Models – Funding an Internet of Things Startup.

UNIT 5

Moving to Manufacture – Designing Kits – Designing Printed Circuit Boards – Manufacturing Printed Circuit Boards – Mass Producing the case and other Fixtures – Scaling up Software – **Ethics** – Characterizing the Internet of Things – Control – Environment – Solutions.

Text Books:

1. Adrian Mcewen and HakinCassimally, "Designing The Internet of Things" Wiley Publications , 2015.

Reference Books:

1. Vijay Madisetti and ArshdeepBahga, "Internet of Things (A Hands-on-Approach)", 1stEdition, VPT, 2014.
2. Francis daCosta, "Rethinking the Internet of Things: A Scalable Approach to Connecting Everything", 1st Edition, Apress Publications, 2013

CunoPfister, "Getting Started with the Internet of Things", O"Reilly Media, 2011

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25205) SOFTWARE PROJECT MANAGEMENT

Elective-III

Course Outcomes:

- Describe the principles, techniques, methods & tools for model based management of software projects, assurance of product quality and process adherence (quality assurance), as well as experienced based creation and improvements of models (process management) .
- Understanding the basic steps of project planning, project management, quality assurance, and process management and their relationships.
- To provide basic project management skills with a strong emphasis on issues and problems associated with delivering successful IT projects.

Course Outcomes:

- Demonstrate effective project execution and control techniques that result in successful projects.
- Conduct project closure activities and obtain formal project acceptance.
- Demonstrate a strong working knowledge of ethics and professional responsibility.
- Demonstrate effective organizational leadership and change skills for managing projects, project teams, and stakeholders.

UNIT I : Project Evaluation And Project Planning

Importance of Software Project Management, Activities Methodologies, Categorization of Software Projects , Setting objectives , Management Principles, Management Control, Project portfolio Management, Cost-benefit evaluation technology, Risk evaluation, Strategic program Management, Stepwise Project Planning.

UNIT II : Project Life Cycle And Effort

Software process and Process Models, Choice of Process models, mental delivery, Rapid Application development, Agile methods, Extreme Programming, SCRUM, Managing interactive processes, Basics of Software estimation, Effort and Cost estimation techniques, COSMIC Full function points, COCOMO II A Parametric Productivity Model, Staffing Pattern.

UNIT III : Activity Planning And Risk Management

Objectives of Activity planning, Project schedules, Activities, Sequencing and scheduling, Network Planning models, Forward Pass & Backward Pass techniques, Critical path (CRM) method, Risk identification, Assessment, Monitoring, PERT technique, Monte Carlo simulation, Resource Allocation, Creation of critical patterns, Cost schedules.

UNIT IV : Project Management And Control

Framework for Management and control, Collection of data Project termination, Visualizing progress, Cost monitoring, Earned Value Analysis- Project tracking, Change control- Software Configuration Management, Managing contracts, Contract Management.

UNIT V : Staffing In Software Projects Managing people, Organizational behavior, Best methods of staff selection, Motivation, The Oldham-Hackman job characteristic model, Ethical and Programmed concerns, Working in teams, Decision making, Team structures, Virtual teams, Communications genres, Communication plans.

Text Books:

1. Bob Hughes, Mike Cotterell and Rajib Mall: Software Project Management – Fifth Edition, Tata McGraw Hill, New Delhi, 2012.

References Books:

1. Robert K. Wysocki "Effective Software Project Management" – Wiley Publication,2011.
2. Walker Royce: "Software Project Management"- Addison-Wesley, 1998.
3. Gopalaswamy Ramesh, "Managing Global Software Projects" – McGraw Hill Education (India), Fourteenth Reprint 2013.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D58201) BIG DATA ANALYTICS

Elective-IV

Objectives:

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

Outcomes:

On completion of this course the student will able to

- Analyze the big data analytics techniques for useful business application.
- Design efficient algorithms for mining the data from large volumes.
- Analyze the HADOOP and Map Reduce technologies associated with big data analytics.
- Explore on big data applications using Pig and Hive.

UNIT-I

Introduction to Big Data

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

UNIT- II

Mining Data Streams

Introduction To Stream Concepts – Stream Data Model and Architecture - Stream Computing – Sampling Data in a Stream – Filtering Stream – 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 – III

Hadoop

History of Hadoop- The Hadoop Distributed File System – Components of Hadoop – Analyzing the Data with Hadoop – Scaling Out – Hadoop Streaming – Design of HDFS- Java interfaces to HDFSBasics- Developing a Map Reduce Application – How Map Reduce Works – Anatomy of a Map Reduce Job run – Failures – Job Scheduling – Shuffle and Sort – Task Execution – Map Reduce Types and Formats – Map Reduce Features.

UNIT – IV

Hadoop Environment

Setting up a Hadoop Cluster – Cluster specification – Cluster Setup and Installation –Hadoop Configuration – Security in Hadoop – Administering Hadoop – HDFS – Monitoring – Maintence – Hadoop Benchmarks – Hadoop in the Cloud

UNIT –V

Frameworks

Applications on Big Data Using Pig and Hive – Data Processing operators in Pig – Hive Services – HiveQL – Querying Data in Hive – fundamentals of HBase and Zookeeper – IBM Info Sphere Big Insights and Streams. Visualization - Visual data analysis techniques, interaction techniques; Systems and applications.

Text Books:

1. Michael Berthold, David J.Hand, Intelligent Data Analysis, Spingers, 2007.
2. Tom White, Hadoop: The Definitive Guide Third Edition, O'reilly Media, 2012.
3. Chris Eaton, Dirk DeRoos, Tom Deutsch, George Lapis, Paul Zikopoulos, Uderstanding Big Data : Analytics for Enterprise Class Hadoop and Streaming Data, McGrawHill Publishing, 2012.
4. AnandRajaraman and Jeffrey David UIIman, Mining of Massive Datasets Cambridge University Press, 2012.

Reference Books:

1. Bill Franks, Taming the big Data tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics, John Wiley & sons, 2012.
2. Glenn J. Myatt, Making Sense of Data , John Wiley & Sons, 2007 Pete Warden, Big Data Glossary, O'Reilly, 2011.
3. Jiawei Han, MichelineKamber, Data Mining Concepts and Techniques, Second Edition.
4. Elsevier, Reprinted 2008. Da Ruan, Guoquing Chen, Etienne E.Kerre, Geert Wets, Intelligent Data Mining, Springer, 2007.

5. 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.
6. Michael Minelli (Author), Michele Chambers (Author), AmbigaDhirraj (Author), Big Data, BigSnalytics.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25206) SOFTWARE REENGINEERING
Elective-IV

Objectives:

To explain why software re-engineering is a cost-effective option for system evolution
To describe the activities involved in the software re-engineering process

To distinguish between software and data re-engineering and to explain the problems of data re-engineering.

Outcomes:

- Assess which parts should be reengineered first;
- Identify the risks and opportunities for a given re-engineering project;
- Extract coarse-grained and fine-grained design models;
- select the most appropriate migration strategy;
- Solve the typical problems of an object-oriented re-engineering project.

UNIT I:

Software, Software Evolution and Maintenance: Software, Legacy software, Well designed software, Software evolution challenges, Lehman's laws, Software deterioration curve.

Software maintenance: Software change, Types of change encountered during the support phase, Maintenance costs, Why is software maintenance expensive?, Factors affecting maintenance, Maintenance process, Change and maintenance prediction.

Software Quality Factors, Quality and Maintainability Metrics: Internal and external attributes, McCall's quality factors, ISO 9126 quality factors, Need and importance of quality and maintainability metrics, Metric for software correctness (Defects/KLOC), Metric for software integrity, Software reliability (MTBF), Metrics for maintainability (Mean-time-to-change (MTTC), Spoilage metric, Software maturity index, McCabe and Halstead metrics).

UNIT II:

Design maintainability: Cohesion, Coupling, Understandability and Adaptability.

Legacy software structure, Software reengineering process model: Software change strategies include: Software maintenance, Architectural transformation, Software reengineering. Legacy software structure and distribution: Ideal structure, Real structure, Layered distribution model, Legacy software distribution, Architectural problems.

Business process reengineering: Business processes, A BPR Model, Software reengineering and its importance, Goals of reengineering, A software reengineering process model, Software reengineering activities.

UNIT III:

Design Extraction: Reverse Engineering: Goals of reverse engineering, Why design extraction is needed?, Reverse engineering process, Reverse engineering to understand processing, Code duplication detection, Reverse engineering to understand data, Reverse engineering user interfaces, Design extraction with UML, Heuristics to extract the design, Tools for reverse engineering.

Restructuring (In Traditional context): Code restructuring: Characteristics of unstructured code, Characteristics of structured code, Spaghetti logic, Structured control logic, Restructuring problems, Flow graph restructuring, Warnier's logical simplification techniques, Some basic code restructuring methods: Interchange, Transposition, Combination, Resolution, Substitution.

UNIT IV:

Data restructuring (Data reengineering): Data reengineering process, Data problems, Approaches: Data cleanup, Data extension, Data migration. Tools for restructuring.

Refactoring (Restructuring in object oriented context): What is refactoring?, Principles in refactoring: Why should you refactor?, When should you refactor?, Problems with refactoring, Refactoring and design, Refactoring and performance. Refactoring opportunities, Top ten of code bad smells, Different refactorings and their use, Refactoring tools.

UNIT V:

Forward Engineering: What is forward engineering ? Goals of forward engineering, Forward engineering for client/server applications, Forward engineering for object oriented architectures, Forward engineering user interfaces, Tools for forward engineering.

Reengineering Metrics, Repositories, and Economics:

Metrics in Reengineering: Why metrics in Reengineering?, Metrics as a reengineering tool, Which metrics to collect ?(Goal Question Metric (GQM) paradigm), Reengineering repositories: Why repositories?, Taxonomy (Functionality + Integration options), Issues, Reengineering economics.

TEXT BOOKS:

1. Software Reengineering, Ed. Robert S. Arnold, IEEE Computer Society, 1993.
2. Software Evolution, Tom Mens, Serge Demeyer, Springer publication company, 2008.

REFERENCES

1. Software Engineering, Ian Sommerville, Addison-Wesley, 6th Edition.

2. Software Engineering, A Practitioner's Approach, Roger S. Pressman, 6th Edition.

3. Refactoring: Improving the Design of Existing Code, Martin Fowler, K.Beck, J.Brant, W.Opdyke, D.Roberts, Addison- Wesley, NY, 1999.

4. Software Reengineering, Georg Abfalter, VDM Verlag, Germany, 2008.

5. Successful Software Reengineering, Salvatore Valenti, IRM Press, 2002.

6. Logical construction of programs, J.D.Warnier,Van Nostrand-Reinhold,1974.

7. Tutorial on Software Restructuring, Robert E.Arnold, IEEE Computer Society, 1986.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25207) SOFTWARE CONFIGURATION MANAGEMENT

Elective-IV

Course Objectives:

- To learn the changing nature of software and need for change management.
- To study the different phases involved in software configuration management.
- To learn about the SCM plans, audits and reviews
- To study the various SCM tools and implementation techniques
- To study the SCM different scenarios and future directions

Course Outcomes:

- Identifying items that need to be controlled for changes.
- Systematically controlling changes to them.
- Establishing & maintaining integrity of these items and providing accurate status of items to relevant stakeholders (like developers, end users, and customers) throughout the Software Development Lifecycle.

UNIT I

OVERVIEW THE SOFTWARE CONFIGURATION MANAGEMENT

SCM: Concepts and definitions – SCM Plan – Software development life cycle models – SDLC Phases – Need and importance of Software configuration management – Increased complexity and demand – Changing nature of software and need for change management – Lower maintenance costs and better quality assurance – Faster problem identification and bug fixes - SCM: Basic concepts – Baselines – Check-in and Check-out- Versions and Variants – System Building – Releases.

UNIT II

DIFFERENT PHASES OF SOFTWARE CONFIGURATION MANAGEMENT

Different Phases Of Scm – SCM System design - SCM Plan preparation – SCM Team organization – SCM Infrastructure organization – SCM Team training – Project team training – Configuration identification – Configuration Control –Configuration status accounting – Configuration audits.

UNIT III

CONFIGURATION AUDITS AND MANAGEMENT PLANS

When, what and who of auditing - Functional Configuration audit – Physical Configuration audit – Auditing the SCM System – Role of SCM Team in configuration audits – SCM plan and the incremental approach – SCM Plan and SCM Tools – SCM Organization.

UNIT IV

SOFTWARE CONFIGURATION MANAGEMENT TOOLS AND IMPLEMENTATION

Advantages of SCM tools – Reasons for the increasing popularity of SCM tools – SCM Tools and SCM Functions – SCM tool selection – Role of Technology –Selection criteria – Tool implementation – SCM implementation plan –implementation strategy – SCM Implementation team.

UNIT V

TRENDS IN SCM: FUTURE DIRECTIONS

SCM in different scenarios – SCM and project size – SCM in integrated development environments – SCM In distributed environments – SCM and CASE Tools - Trends in SCM - Hardware and Software Management – Better integration with IDE'S and CASE environments – Customization – Better decision making capabilities – Reduction in SCM Team size – Market snapshot.

REFERENCES

1. Jessica Keyes, Software Configuration Management, Auerbach Publications, 2008.

2. Alexis Leon, Software Configuration Management Handbook, Artech Print on Demand; 2 edition, 2009.

3. Robert Aiello and Leslie Sachs Configuration Management Best Practices: Practical Methods that work in Real World, , Addison-Wesley Professional; 1 edition, 2010.

4. Stephen P. Berczuk, Brad Appleton and Kyle Brown , "Software Configuration Management Patterns: Effective Teamwork and Practical Integration", Addison-Wesley , 2003.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
4	0	0	4

(17D25208) SECURE SOFTWARE ENGINEERING

Elective-IV

Course Objectives:

Students will demonstrate knowledge of the distinction between critical and non-critical systems.

Students will demonstrate the ability to manage a project including planning, scheduling and risk assessment/management.

Students will demonstrate an understanding of the proper contents of a software requirements document for secure software engineering.

Students will author a formal specification for secure software systems.

Students will demonstrate an understanding of distributed system architectures and application architectures.

Students will demonstrate an understanding of the differences between real-time and non-real time systems.

Course Outcomes:

Ability to identify specific components of a software design that can be targeted for reuse.

Ability to learn software testing plan and metrics for secure software engineering.

UNIT I

Why Is Security a Software Issue?

Introduction, The problem, Software assurance and software security, Threats to software security, Sources of software insecurity, the benefits of detecting software security defects early, managing secure software development.

What Makes Software Secure?

Defining properties of secure software, How to influence the security properties of software,

How to assert and specify desired security properties.

UNIT II

Requirements Engineering for Secure Software

Introduction, Misuse and Abuse Cases, The SQUARE process model: SQUARE sample outputs, Requirements elicitation, Requirements Prioritization.

Secure Software Architecture and Design

Introduction, Software security practices for architecture and design: Architectural risk analysis. Software security knowledge for architecture and design: Security principles, Security guidelines, and Attack patterns.

UNIT III

Considerations for Secure Coding and Testing

Introduction, Code analysis, Coding practices, Software security testing, Security testing considerations throughout the SDLC.

Security and Complexity: System Assembly Challenges

Introduction, Security failures, Functional and attacker perspectives for security analysis, System complexity drivers and security, Deep technical problem complexity.

UNIT IV

Governance, and Managing for More Secure Software

Introduction, Governance and security, Adopting an enterprise software security framework, How much security is enough?, Security and project management, maturity of practice.

UNIT V

Security Metrics

Defining security metrics, Diagnosing problems and measuring technical security, Analysis

Techniques, Organize, aggregate, and analyze data to bring out key insights.

TEXT BOOKS

1. Software Security Engineering: A Guide for Project Managers, by Julia H. Allen, Sean Barnum, Robert J. Ellison, Gary McGraw, Nancy R. Mead, Addison-Wesley , 1st edition, 2008.

2. Security Metrics: Replacing Fear, Uncertainty, and Doubt , by Andrew Jaquith, Addison-Wesley , 1st edition , 2007.

REFERENCES

1. Integrating Security and Software Engineering: Advances and Future Vision, by Haralambos Mouratidis, Paolo Giorgini, IGI Global, 2006.
2. Software Security: Building Security In , by Gary McGraw , Addison-Wesley, 2006
3. The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities, by Mark Dowd, John McDonald, Justin Schuh, Addison-Wesley, 1st edition, 2006

4. Building Secure Software: How to Avoid Security Problems the Right Way by John Viega,Gary McGraw, Addison-Wesley, 2001

5. Writing Secure Code, by M. Howard, D. LeBlanc, Microsoft Press, 2nd Edition, 2003.

6. Exploiting Software: How to break code, by G. Hoglund, G. McGraw, Addison Wesley, 2004.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
0	0	4	2

(17D25209) ADVANCES IN SOFTWARE TESTING LAB

Course Objectives:

To learn to use the following (or Similar) automated testing tools to automate testing:

- Win Runner/QTP for functional testing.
- Load Runner for Load/Stress testing.
- Test Director for test management.
- JUnit, HTMLUnit, CPPUnit.

- To study state-of-art tools for software testing and Middleware technologies

Course Outcomes:

- Test the software applications using standard tools available in the market

1. Write programs in C Language to demonstrate the working of the following constructs: i) do...while ii) while….do iii) if…else iv) switch v) for
2. A program written in C language for Matrix Multiplication fails. Introspect the causes for its failure and write down the possible reasons for its failure.
3. Consider ATM System and Study its system specifications and report the various bugs.
4. Write the test cases for Banking application.
5. Create test plan document for Library Management System.
6. Create test cases for Railway Reservation.
7. Create test plan document for Online Shopping.

Working with Tool's:

Understand the Automation Testing Approach, Benefits, Workflow, Commands and Perform

Testing on one application using the following Tool's.

1. Win runner Tool for Testing.

2. Load runner Tool for Performance Testing.
3. Selenium Tool for Web Testing.
4. Bugzilla Tool for Bug Tracking.
5. Test Director Tool for Test Management.
6. Test Link Tool for Open Source Testing.

References:

1. M G Limaye, “Software Testing – Principles, Techniques and Tools”, Tata McGraw Hill, 2009.
2. Edward Kit, “Software Testing in the Real World - Improving the Process”, Pearson Education, 2004.
3. William E. Perry, “Effective methods for software testing”, 2nd Edition, John Wiley, 2000.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II semester (SE)

L	T	P	C
0	0	4	2

(17D25210) MODEL DRIVEN SOFTWARE ENGINEERING LAB

Course Objectives:

- Speed time to build large applications
- Lower the risk of large applications
- Simplify development
- Lower the required skill level needed to work on large applications.

Course Outcomes:

- Make and defend decisions regarding the appropriate use of modelling throughout the software development life-cycle.
- Realize design practically, using an appropriate software engineering methodology.
- Able to use modern engineering tools for specification, design, implementation, and testing.

List of Experiments:

1. Practicing the different types of case tools such as Rational Rose / other Open Source can be used for all the phases of Software development life cycle.
2. Data modeling for following applications
    a. Library System
    b. Student Marks Analyzing System
    c. Text Editor.
    d. Create a dictionary.

e. Telephone directory.

3. Generate the source code for following applications

   a. Library System

   b. Student Marks Analyzing System

   c. Text Editor.

   d. Create a dictionary.

   e. Telephone directory.

4. Drive Development with unit test using Test Driven Development.

5. Drive Development with regression test using Test Driven Development.

6. Implement model integrity in EMF with MDT OCL.

References:

1. Model-Driven Software Development by Sami Beydeda Matthias Book , Volker Gruhn, Springer.

2. Model Driven Systems Development with Rational Products By Brian Nolan, Barclay Brown, Dr. Laurent Balmelli, Et Al Tim Bohn, 2008,IBM.

3. Model Driven Development with Executable UML by Dragan Milicev, 2009,Wilei India pvt Ltd.

4. Model Driven Software Development by Kevin Lano, Apr 2009, Ci Business Press.

*****

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech II sem (SE)

L	T	P	C
0	0	4	2

(17D58112) SOFTWARE PATTERNS LAB

Course Objectives:

Construct UML diagrams for static view and dynamic view of the system.

Generate creational patterns by applicable patterns for given context.

Create refined model for given Scenario using structural patterns.

Construct behavioral patterns for given applications.

Construct architectural patterns for given applications.

Course Outcomes:

Understand the Case studies and design the Model..

Understand how design patterns solve design problems.

Develop design solutions using creational patterns.

Construct design solutions by using structural, behavioral and architectural patterns

Student is expected to complete the following experiments as a part of laboratory work.

1. Identify the application where you can use single pattern and implement it.
2. Using UML design one of the architectural patterns.
3. Using UML design one of the creational patterns.
4. Using UML design one of the structural patterns.
5. Using UML design one of the behavioral patterns.
6. User gives a print command from a word document. Design to represent this chain of responsibility design pattern.

7. User gives a print command from a word document. Design to represent this Singleton design pattern.
8. Identify the application where you can use multiple creational patterns and implement it.
9. Identify the application where you can use multiple structural patterns and implement it.
10. Identify the application where you can use multiple behavioral patterns and implement it.
11. Identify the application where you can use architectural patterns and implement it.

References:

1. AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, by William J. Brown, Raphael C. Malveau, Hays W. "Skip" McCormick , Thomas J. Mowbray (Author) 1st Edition,
2. Java testing patterns, John Thomas etc, wiley.
3. Software architecture, David M. Dikel, David Kane and James R. Wilson, Prentice Hall PTR,2001
4. Head First Design patterns, Eric Freeman & Elisabeth Freeman, O'REILLY, 2007.
5. Design Patterns in Java, Steven John Metsker & William C. Wake, Pearson education, 2006

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech III semester (SE)

L	T	P	C
4	0	0	4

(17D20301) RESEARCH METHODOLOGY

(Elective V-OPEN ELECTIVE)

UNIT I

Meaning of Research – Objectives of Research – Types of Research – Research Approaches – Guidelines for Selecting and Defining a Research Problem – research Design – Concepts related to Research Design – Basic Principles of Experimental Design.

UNIT II

Sampling Design – steps in Sampling Design –Characteristics of a Good Sample Design – Random Sampling Design.

Measurement and Scaling Techniques-Errors in Measurement – Tests of Sound Measurement – Scaling and Scale Construction Techniques – Time Series Analysis – Interpolation and Extrapolation.

Data Collection Methods – Primary Data – Secondary data – Questionnaire Survey and Interviews.

UNIT III

Correlation and Regression Analysis – Method of Least Squares – Regression vs Correlation – Correlation vs Determination – Types of Correlations and Their Applications

UNIT IV

Statistical Inference: Tests of Hypothesis – Parametric vs Non-parametric Tests – Hypothesis Testing Procedure – Sampling Theory – Sampling Distribution – Chi-square Test – Analysis of variance and Co-variance – Multi-variate Analysis.

UNIT V

Report Writing and Professional Ethics: Interpretation of Data – Report Writing – Layout of a Research Paper – Techniques of Interpretation- Making Scientific Presentations in Conferences and Seminars – Professional Ethics in Research.

Text Books:

1. Research Methodology:Methods And Techniques – C.R.Kothari, 2nd Edition,New Age International Publishers.
2. Research Methodology: A Step By Step Guide For Beginners- Ranjit Kumar, Sage Publications (Available As Pdf On Internet)
3. Research Methodology And Statistical Tools – P.Narayana Reddy And G.V.R.K.Acharyulu, 1st Edition,Excel Books,New Delhi.

REFERENCES:

1. Scientists Must Write - Robert Barrass (Available As Pdf On Internet)

2. Crafting Your Research Future –Charles X. Ling And Quiang Yang (Available As Pdf On Internet)

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech III semester (SE)

L	T	P	C
4	0	0	4

(17D20302) HUMAN VALUES AND PROFESSIONAL ETHICS

(Elective V-OPEN ELECTIVE)

Unit I:

HUMAN VALUES:Morals, Values and Ethics-Integrity-Work Ethic-Service learning – Civic Virtue – Respect for others – Living Peacefully – Caring – Sharing – Honesty - Courage- Co Operation – Commitment – Empathy –Self Confidence Character – Spirituality.

Unit II:

ENGINEERING ETHICS: Senses of Engineering Ethics- Variety of moral issues – Types of inquiry – Moral dilemmas – Moral autonomy –Kohlberg‟s theory- Gilligan‟s theory- Consensus and controversy – Models of professional roles- Theories about right action- Self interest - Customs and religion –Uses of Ethical theories – Valuing time –Co operation – Commitment.

Unit III :

ENGINEERING AS SOCIAL EXPERIMENTATION: Engineering As Social Experimentation – Framing the problem – Determining the facts – Codes of Ethics – Clarifying Concepts – Application issues – Common Ground - General Principles – Utilitarian thinking respect for persons.

UNIT IV:

ENGINEERS RESPONSIBILITY FOR SAFETY AND RISK: Safety and risk – Assessment of safety and risk – Risk benefit analysis and reducing riskSafety and the Engineer- Designing for the safety- Intellectual Property rights(IPR).

UINIT V:

GLOBAL ISSUES: Globalization – Cross culture issues- Environmental Ethics – Computer Ethics – Computers as the instrument of Unethical behavior – Computers as the object of Unethical acts – Autonomous Computers- Computer codes of Ethics – Weapons Development - Ethics .

Text Books :

1. “Engineering Ethics includes Human Values” by M.Govindarajan, S.Natarajan and V.S.SenthilKumar-PHI Learning Pvt. Ltd-2009.

2. “Engineering Ethics” by Harris, Pritchard and Rabins, CENGAGE Learning, India Edition, 2009.

3. “Ethics in Engineering” by Mike W. Martin and Roland Schinzinger – Tata McGrawHill– 2003.

4. “Professional Ethics and Morals” by Prof.A.R.Aryasri, Dharanikota Suyodhana-Maruthi Publications.

5. “Professional Ethics and Human Values” by A.Alavudeen, R.Kalil Rahman and M.Jayakumaran , Laxmi Publications.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

M.Tech III semester (SE)

L	T	P	C
4	0	0	4

(17D20303) INTELLECTUAL PROPERTY RIGHTS

(Elective V-OPEN ELECTIVE)

UNIT – I

Introduction To Intellectual Property: Introduction, Types Of Intellectual Property, International Organizations, Agencies And Treaties, Importance Of Intellectual Property Rights.

UNIT – II

Trade Marks : Purpose And Function Of Trade Marks, Acquisition Of Trade Mark Rights, Protectable Matter, Selecting And Evaluating Trade Mark, Trade Mark Registration Processes.

UNIT – III

Law Of Copy Rights : Fundamental Of Copy Right Law, Originality Of Material, Rights Of Reproduction, Rights To Perform The Work Publicly, Copy Right Ownership Issues, Copy Right Registration, Notice Of Copy Right, International Copy Right Law.

Law Of Patents : Foundation Of Patent Law, Patent Searching Process, Ownership Rights And Transfer

UNIT – IV

Trade Secrets : Trade Secrete Law, Determination Of Trade Secrete Status, Liability For Misappropriations Of Trade Secrets, Protection For Submission, Trade Secrete Litigation.

Unfair Competition : Misappropriation Right Of Publicity, False Advertising.

UNIT – V

New Development Of Intellectual Property: New Developments In Trade Mark Law ; Copy Right Law, Patent Law, Intellectual Property Audits.

International Overview On Intellectual Property, International – Trade Mark Law, Copy Right Law, International Patent Law, International Development In Trade Secrets Law.

TEXT BOOKS & REFERENCES:

1. Intellectual Property Right, Deborah. E. Bouchoux, Cengage Learing.

2. Intellectual Property Right – Nleashmy The Knowledge Economy, Prabuddha Ganguli,

Tate Mc Graw Hill Publishing Company Ltd.,