CC503: Software Project Management

(4 Credits, 3L + 2T)

Objectives: To provide basic project management skills with a strong emphasis on issues and problems associated with delivering successful IT projects and how the software is tested using various techniques to improve the quality of software. This course is designed to provide an understanding of the particular issues encountered in handling IT projects and to offer students methods, techniques and 'hands-on' experience in dealing with them.

Learning Outcomes: At the end of this course, student should be able to

- Understand and practice the process of project management and its application in delivering successful IT projects;
- Evaluate a project to develop the scope of work, provide accurate cost estimates and to plan the various activities;
- Identify the resources required for a project and to produce a work plan and resource schedule;
- Understand diff. types of testing and how it is conducted.
- Practice the automated tools available for testing

Text Book(s):

A) Information Technology Project Management" Kathy schwalbe, International student edition, THOMSON course Technology, 2003

B) "Software project management "Bob Hughes and Mike Cottrell, Third edition, Tata McGraw-Hill

C) "Microsoft office Project 2003 Bible", Elaine Marmel, Wiley publishing Inc.

Software Requirement: Microsoft project 2003

UNIT1:

Introduction to project management: (5L)

Project, project management, Importance, characteristics of project how software projects are diff. than other projects, Problems with software projects, Phases: Initiation phase, planning phase, execution phase, monitoring and controlling phase, and closing phase. All parties involved in project, Role of Project Manager, Project management framework, Software tool for project management

UNIT2:

Project planning: (6L)

Integration management: What is integration management, plan development and execution, What is scope management, methods for selecting project, scope statement, Work Breakdown Structure, main steps in Project planning: identify project scope and objective, identify project infrastructure, analyze project characteristics, identify project products and activities, estimate effort for each activity, identify risk activity, allocate resources, review plan, execute plan. Use of software (Microsoft Project) to assist in project planning activities

UNIT3:

Project scheduling: (6L)

Time management: importance of Project schedules, schedules and activities, sequencing and scheduling activities, Network Planning models, duration estimation and schedule development, Critical path analysis, PERT, Use of software(Microsoft project) to assist in project scheduling

UNIT4:

Project cost management: (6L)

Importance and principles of project cost management, Resource planning, Attributes to be considered in cost estimation, factors affecting the cost, various costs involved in it. Traditional method: Estimation by analogy, Expert judgment, Parkinson, price to win, top down, bottom up. COCOMO Model, Function point analysis,
Function point analysis, Cost control, Use of software(Microsoft project) to assist in cost management

UNIT5:

Project quality management: (5L)

Quality of information technology project, Stages of software quality management, PMBOK, Quality standards, Tools and techniques for quality control.

UNIT6:

Project risk management: (5L)

The importance, Top risk in projects, Common sources of risk in IT projects, elements in risk mgt., Risk identification, Risk quantification, Risk response development and control, using software to assist in project risk management

UNIT7:

Fundamentals of Testing: (12L)

Fundamentals of Software quality, quality views, People challenges in testing, Principles of Verification and validation, Techniques of verification, V model, Testing process, Unit testing, Integration testing, System Testing and Acceptance testing, Testing new product versions, Testing planning: test plan, test plan template, risk analysis, Test Design, good test case, test case template, test case mistakes, Testing Execution: objectives, execution considerations, test execution activities, executing test, Defect management: what is defect, defect life cycle, defect management process, Test Metrics: purpose, characteristics of good metrics, metrics, Functional testing tools, Unit testing tools, Test management tools.

CC501 SOFT COMPUTING

(Credit 4 , L-3, T-2)

objectives

- To introduce the ideas of fuzzy sets, fuzzy logic and use of heuristics based on human experience
- To become familiar with neural networks that can learn from available examples and generalize to form appropriate rules for inferencing systems
- To provide the mathematical background for carrying out the optimization associated with neural network learning
- To familiarize with genetic algorithms and other random search procedures useful while seeking global optimum in self-learning situations
- To introduce case studies utilizing the above and illustrate the intelligent behavior of programs based on soft computing

Learning Outcomes: To introduce the techniques of soft computing and adaptive neuro-fuzzy inferencing systems which differ from conventional AI and computing in terms of its tolerance to imprecision and uncertainty.

Text Books:

1. S. Rajsekaran & G.A. Vijayalakshmi Pai, "Neural Networks,Fuzzy Logic and Genetic Algorithm:Synthesis and Applications" Prentice Hall of India.

2. N.P.Padhy,"Artificial Intelligence and Intelligent Systems" Oxford University Press.

Reference Books:

3. Siman Haykin,"Neural Netowrks"Prentice Hall of India

4. Timothy J. Ross, "Fuzzy Logic with Engineering Applications" Wiley India.

5. Kumar Satish, "Neural Networks" Tata Mc Graw Hill

6. J.S.R.Jang, C.T.Sun and E.Mizutani, "Neuro-Fuzzy and Soft Computing", PHI, 2004, Pearson Education 2004.

7. Timothy J.Ross, "Fuzzy Logic with Engineering Applications", McGraw-Hill, 1997.

8.. Davis E.Goldberg, "Genetic Algorithms: Search, Optimization and Machine Learning", Addison Wesley, N.Y., 1989.

9. S. Rajasekaran and G.A.V.Pai, “Neural Networks, Fuzzy Logic and Genetic Algorithms”, PHI, 2003.
10.. R.Eberhart, P.Simpson and R.Dobbins, “Computational Intelligence - PC Tools”, AP Professional, Boston, 1996.

Syllabus:

Unit-I

Neural Networks-1(Introduction & Architecture)

Neuron, Nerve structure and synapse, Artificial Neuron and its model, activation functions, Neural network architecture: single layer and multilayer feed forward networks, recurrent networks. Various learning techniques; perception and convergence rule, Auto-associative and hetro-associative memory. (10)

Unit-II

Neural Networks-II (Back propagation networks)

Architecture: perception model, solution, single layer artificial neural network, multilayer perception model; back propagation learning methods, effect of learning rule co-efficient ;back propagation algorithm, factors affecting back propagation training, applications. (10)

Unit-III

Fuzzy Logic

Basic concepts of fuzzy logic, Fuzzy sets and Crisp sets, Fuzzy set theory and operations, Properties of fuzzy sets, Fuzzy and Crisp relations, Fuzzy to Crisp conversion. Membership functions, interference in fuzzy logic, fuzzy if-then rules, Fuzzy implications and Fuzzy algorithms, Fuzzyfications & Defuzzificataions, Fuzzy Controller, Industrial applications. (10)

Unit-IV

Rough Sets

Introduction, Indisnibility Relations, Decernibility Matrix, Lower Approximation, Upper Approximation, Boundary, Accuracy of Approximation, Rule Induction, Case Study

Unit-V

Genetic Algorithm (GA)

Basic concepts, working principle, procedures of GA, flow chart of GA, Genetic representations, (encoding) Initialization and selection, Genetic operators, Mutation, Generational Cycle, applications. (9)

CC502: Finite Automata and Grammars

(4 Credits, 3L + 2T, Level 3)

Objectives: The course introduces some fundamental concepts in automata theory and formal languages including grammar, finite automaton, regular expression, formal language, pushdown automaton, and Turing machine. Not only do they form basic models of computation, they are also the foundation of many branches of computer science, e.g. compilers, software engineering, concurrent systems, etc. The properties of these models will be studied and various rigorous techniques for analyzing and comparing them will be discussed, by using both formalism and examples. To introduce students the basic concepts in theoretical computer science, and the formal relationships among machines, languages and grammars.

Learning Outcomes: Upon successful completion of this course, students will be able to:

Sr.No.	Learning Outcomes
	Knowledge
1.	Explain the basic concepts of deterministic and non-deterministic finite automata, regular language, context-free language, Turing machines, Church's thesis, halting problem, computability and complexity
2.	**Describe the formal relationships among machines, languages and grammars**
	Professional Skill
3.	**Perceive the power and limitation of a computer**
4.	**Solve the problems using formal language**
	Attitude
5.	**Develop a view on the importance of computational theory**

Textbook: J. Hopcroft, R. Motwani, and J. Ullman. Introduction to Automata Theory, Languages, and Computation, 3rd edition, 2006, Addison-Wesley.

Reference Books:

(1) P. Linz. Introduction to Formal Languages and Automata, 5th edition, 2011 (or 4th or 3rd edition), Jones and Barlett;
(2) Michael Sipser, Introduction to the Theory of Computation, First Edition, 1997, PWS Publishing Company.

Syllabus:

UNIT-1:
Basic concepts of finite automata and languages, deterministic finite automaton, nondeterminism, equivalence between DFA and NFA, regular expression and equivalence to FA

UNIT-2:
Algebraic laws for regular expressions pumping lemma and applications, properties of regular languages, minimization of automata and applications.

UNIT-3:
Context-free grammars and languages, parsing (or derivation) and parse trees, ambiguity of grammar and language, Chmosky normal form of CFG, pumping lemma, properties of CFLs

UNIT-4:
Pushdown automaton (PDA), various forms of PDA, equivalence between CFG and PDA, equivalence between CFG and PDA,

UNIT-5:
Turing machines and (un)decidability