Department of Computer Science & Engineering

School of Engineering and Computer Science

Independent University, Bangladesh

Curriculum

for

M Sc. in Software Engineering

Version: 3.2

Revision record

Version	Status	Date	Note
1.0	Approved by UGC	Dec 07, 2005	Initial version prepared for program approval
2.0	Approved by Academic council	Mar 13, 2013	Change is optional course and allow student to take graduate project
3.0	Under review	Jul 13, 2017	Update document for peer review along with outcome based education
3.1	Under review	Dec 14, 2017	Incorporation of mapping with PLO and CO
3.2	Under review	Mar 19, 2017	Incorporation of departmental vision, mission along with program Vision, Mission and graduate profile

TABLE OF CONTENTS

LIST OF TABLES:

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (CSE)

INTRODUCTION:

Bangladesh has seen remarkable growth in the ICT sector in recent times and IUB's Computer Science & Engineering (CSE) Department is an active participant in this growth. The Department of Computer Science & Engineering (CSE) at IUB, under the School of Engineering and Computer Science (SECS), is one of the most dynamic and versatile departments in the university. Our "Application Oriented Learning" philosophy not only teaches students the fundamental principles of computing but also provides them with hands-on training of the various applications of this knowledge. University Grants Commission (UGC) of Bangladesh recognizes this contribution and potential and has provided financial support in excess of Four Crore Taka to develop advanced research facilities. We have arguably the best IT research setup in Bangladesh. 11 Tera Flops GPU server for High-Performance Computing research lab (Center for Cognitive Skill Enhancement), digital fabrication facilities enabled research lab (Fab Lab, IUB), state-of-the-art Robotics and Networking Labs, Electronics and Electrical Circuits laboratories for smart circuit design research, to mention just a few of our facilities.

We offer comprehensive Computer Science and Engineering based curriculums both at the undergraduate and graduate level. Students can mound their education according to their desired research or industry track. The CSE department provides the opportunity to explore specialized topics in Machine Learning, Computer Vision, Data Science, Big Data, Robotics, Software Engineering, Network Administration, Game Development, Cyber Security, and follow the latest and greatest trends in the IT world.

The Department of Computer Science and Engineering offers a 4 (four) year Bachelors of Science program in Computer Science. The degree is designed to provide a deep and broad knowledge of the theory and practical application related to computer systems. The university is distinctive in its emphasis on a broad based liberal arts curriculum based on a North American model. The mission of the university is to deliver a high quality education which will help foster thinking across disciplines, encourage tolerance and understanding of diverse cultural and social traditions, nurture essential values and prepare students for a fast changing world. Department of CSE challenged the students to think critically, encouraged to imagine boldly and guided to implement efficiently.

VISION:

Inspired by the mission of the university, the department of Computer Science and Engineering is striving to create a breed of students who can imagine boldly, think clearly, argue precisely and implement efficiently. Our mission is to perform high impact research leading to sustainable innovation in the area of computer science and engineering thus contributing to the economic growth of the country through a meaningful partnership between the Community, the Industry and the University.

MISSION:

At the very heart of science and technology education lie comprehension, imagination and implementation. The objective of our programs is to contribute in creating a globally recognized and locally sustainable knowledge eco-system through which students will graduate to provide leadership in industry, enterprise, public service and welfare; to encourage and support innovative research; create enthusiasm in learning through hands on training; while creating knowledge throughout the process. We also provide a platform for lifelong learning through graduate programs and continuing education endeavors.

The curriculum objectives for the undergraduate education are to teach communicative skills (oral and written communication) with a strong humanities background through a thorough liberal arts foundation. To have students have a comprehensive understanding of Computer Science and Engineering related subjects (i.e. a major), and a sub-specialization (minor) in a subject of student's own interest and appreciate their inter-connectivity with founding principles. Focus is maintained on outcome based learning. Outcome is quantifiable through students' enhanced skill in comprehension, analytical capability and decision making skill.

The undergraduate programs equip students with the tools needed to become innovators and global leaders in computing and equip them with the background necessary to pursue higher studies. Consequently our programs are designed to have a high impact outcome of contributing to the sustainable development of knowledge-based economy of our nation.

SEMESTER/TERM

There are three academic semesters namely, Autumn, Spring and Summer. The duration of each semester is about four (4) months.

The admission procedure for Autumn (the beginning of the academic year) starts within the first week of July and ends with the commencement of classes in the second /third week of August. The semester ends by the end of December. The admission procedure for Spring starts within the first week of December and the commencement of classes in the January. The semester ends on April. The Summer class begins on May and ends on August.

MEDIUM OF INSTRUCTION

English is the medium of instruction at the university.

TEACHING STRATEGY

The Department of CSE believes in extra-ordinary quality teaching in a local university with a global connection. The Department is concern about the variance among the same-level teachers taking similar courses at undergraduate level, hence focuses on eliminating these differences and discriminations as much as possible that have negative implications on students' learning. It also tries to equate the learning outputs from the same courses and make a unitary approach toward tertiary-level education by conducting trainings and workshops on assessment, measurement, evaluation and other pedagogical instruments. In general, the faculty members use four broad styles, i.e. formal authority, demonstrator or personal model, facilitator and delegator. In next few lines some commonly practiced teaching strategies have been listed:

- Lecture
- Demonstration
- Reading Assignment
- Individual Project/Assignment
- Group Assignment
- Small Group Discussion
- Panel Discussion
- Video demonstration
- Debate
- Simulation Game
- Brainstorming
- Case Studies
- Role Playing
- Seminar/Workshop
- Cooperative Learning
- Problem-based Learning (PBL)
- Inquiry-based Learning (IBL)

ASSESSMENT STRATEGY:

At CSE, two types of assessments are in practice; formative assessment for continuous improvement of the students' learning, summative assessment for grading of students' performance. Generally, CSE believes in assessment for learning, not assessment of learning. Principally the assessment is the responsibility of the individual faculty members according to the standard of North American academia. However, for courses with multiple sections a faculty coordinator mediates the process of assessment. Each course is assessed over a total of 100 numeric marks which is divided into two categories. Continuous assessment, such as quizzes, presentation, class test, etc. covers 40% to 60% while the rest of the marks are allocated for the mid-term and final exams. In some courses, rigorous assignments may replace other forms of assessments. There are no specific criteria for assessing the performance of the students; however, the following criteria are often used for different courses:

Table 1: General evaluation criteria

Criteria	Weight
Class Tests	20
Assignments	20
Midterm	25
Final	35
Total	100

*Student must attend at least 75% classes to get a valid grade.

GRADING SYSTEM

The grading scale in CSE is in line with the IUB's grading system. However, the numeric score for the corresponding letter grade is not a fixed structure, rather a guideline. Following is the prescribed grading structure at CSE:

Table 2: Grading System

Numerical Scores	Letter Grade	Grade Point	Explanation
100-85	A	4.0	Excellent
84-80	A-	3.7	Excellent
79-75	B+	3.3	Good
74-70	B	3.0	Good
69-65	B-	2.7	Good
64-60	C+	2.3	Passing
59-55	C	2.0	Passing
54-50	C-	1.7	Passing
49-45	D+	1.3	Deficient Passing
44-40	D	1.0	Deficient Passing
Below	F	0.0	Fail

COURSE CONTACT HOURS

For each 3 credits course students attend 1 class of 3 hours each week.

MASTERS OF SCIENCE IN SOFTWARE ENGINEERING

The M.Sc. in Software Engineering degree program constitutes preparation for one of the largest and fastest-growing professions: software engineer. The degree program is founded on the principle that developers must be thoroughly practiced in understanding new requirements in diverse fields and, acting as team members or leaders, fully competent to deliver secure, reliable software.

MISSION

The mission is to provide quality graduate education in the core disciplines in both the theoretical and applied foundations of Software Engineering. The goal is to prepare graduates for research and development by conducting research to advance the state of the art in Software Engineering program and by integrating research results and innovations into other programs. The program also targets to induce ethical values and spirit of social commitment by supporting society through participating in and encouraging technology transfer, sharing knowledge and expertise to benefit the country, the region, and beyond, and inspiring people to engage in Software Engineering field.

PROGRAM EDUCATIONAL OBJECTIVES (PEO):

PEO01 THINK CRITICALLY:

Our education system is more geared towards making the students acquiring knowledge but fails to capture their imagination. We want to change that by introducing the students to the exciting world of logic, rationalization and decision making through games and apps. We will introduce our students to various development, analysis and synthesis tools and will engage them in research early in their studies.

PEO2: DESIGN CONSCIENTIOUSLY

A broad liberal arts curriculum covering computer science as well as related technical subjects will provide students with a deeper understanding of the engineering issues and trade-offs that cross disciplines. Courses in humanities and social sciences will equip them with the knowledge and awareness to become responsible members of society who understand the social, political, economic and environmental impact of their decisions. Graduates will become ethical professionals with the ability to work individually or in multidisciplinary teams to design sustainable solutions to complex real-world problems under real-world constraints.

PEO3: IMPLEMENT EFFICIENTLY

A strong emphasis is placed on developing the students' ability to identify and analyze a problem, and to then define the computing requirements appropriate to its solution with regard to factors such as efficiency, sustainability, scalability, available resources and constraints. We ensure that our students are kept up-to-date with knowledge and practice of the most current tools and techniques, and that they develop an appreciation for continued improvement through self-training or formal graduate study. Graduates will possess the ability to design and implement efficient and future-proof software solutions to problems of varying sizes and complexities using modern software development principles and practices.

PROGRAM LEARNING OUTCOME (PLO):

The learning outcome of the degree program supports all the outcomes suggested in the ABET criteria.

1. Knowledge: An ability to select and apply the knowledge, techniques, skills, and modern tools of the computer science and engineering discipline;

2. Requirement Analysis: An ability to identify, analyze, and solve a problem by defining the computing requirements of the problem through effectively gathering of the actual requirements;

3. Problem Analysis: An ability to select and apply the knowledge of mathematics, science, engineering, and technology to computing problems that require the application of principles and applied procedures or methodologies;

4. Design: An ability to design computer based systems, components, or processes to meet the desire requirement;

5. Problem Solving: An ability to apply mathematical foundations, simulation, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.

6. Implementation: An ability to apply design and development principles in the construction of software systems of varying complexity

7. Experiment and Analysis: An ability to conduct standard tests and measurements; to conduct, analyze, and interpret experiments; and to apply experimental results to improve solutions (products or processes);

8. Community engagement and Engineering: An ability to appreciate human behavior, culture, interaction and organization through studies in the humanities and social sciences. A knowledge of the impact of computing solutions in a local and global context;

9. Teamwork: An ability to function effectively as a member or leader of a technical team to accomplish common goals;

10. Communication: An ability to apply written and oral communication in both technical and non-technical environments; an ability to communicate with a range of audience; and an ability to identify and use appropriate available technical literature;

11. Self-directed: Recognition of the need for and an ability to engage in self-directed continuing professional development; be prepared to enter a top-ranked graduate program in Computer Science and Engineering.

12. Ethics: An understanding of and a commitment to address professional, ethical, legal, security, social issues and responsibilities including a respect for diversity;

13. Process Management: A commitment to quality, timeliness, and continuous improvement.

PEO AND PLO MAPPING:

Table 3: PEO and PLO mapping

	PEO01 Thinking Critically	PEO02 Design Conscientiously	PEO03 Implement Efficiently
PLO01 Knowledge	x	x	x
PLO02 Requirement Analysis	x	x	
PLO03 Problem Analysis	x	x	
PLO04 Design	x	x	
PLO05 Problem Solving	x	x	x
PLO06 Implementation			x
PLO07 Experiment and Analysis		x	x
PLO08 Community engagement and Engineering	x	x	x
PLO09 Teamwork	x	x	x
PLO10 Communication	x	x	x
PLO11 Self-directed	x	x	x
PLO12 Ethics	x	x	x
PLO13 Process Management	x	x	x

GRADUATE PROFILE:

The followings are the graduate profile of the department of Computer Science and Engineering.

1. The ability to understand and analyze business and technical requirements.
2. The confidence to work comfortably with a software development and project team.
3. A high degree of competence in producing applications that satisfy requirements.
4. The skills to produce secure, robust, and reliable designs and code.
5. The capability to manage and participate in multi-location, multi-language development projects.

STRUCTURE OF THE PROGRAM

In order to obtain a B.Sc. in Computer Science and Engineering student must complete a minimum of 143 credits of coursework. This includes 42 credits of Foundation studies. The minimum requirement for the major is 86 credit hours including 6 credit hours for internship/senior project.

Table 4: Summarized Program structure

SL	Course Component	Credits
02	Core Courses	21
03	Optional Courses	6
04	Project/Thesis	6
	Total	**33**

CURRICULUM LAYOUT:

Below we provide the detail break-down of the program structure as curriculum layout.

Table 5: Detail Program structure

MSc. in Software Engineering		**33**
Core Courses		**21**
SEN545	Software Engineering Concepts	3
SEN565	Software Architecture and Component-Based Design	3
SEN570	Software Requirement Engineering	3
SEN645	Software Project Management	3
SEN647	Software Engineering Process Management	3
SEN650	Software Marketing	3
SEN653	Software Quality Assurance, Testing and Reliability	3
Optional Courses (Any two/Three)		**6**
SEN 503	Design & Analysis of Algorithm	3
SEN 505	User Interface Design and Development	3
SEN 551	Data Mining and Warehousing	3
SEN 555	Advanced Internet Computing	3
SEN 601	Distributed Database System	3
SEN 605	Analysis and Organization of Information Systems	3
SEN 620	Image Analysis and Pattern Recognition	3
SEN 614	Internet Technology	3
SEN 616	Advanced Internet Computing	3
SEN 620	Computer Graphics	3
SEN 630	Advanced Topics in Computer Networking	3
SEN 646	Software Technology and Innovation Management	3
SEN 649	Business Process Reengineering (BPR) and Software	3
SEN 651	Embedded and Real-time Software Engineering	3
SEN 684	Computer Animation and Virtual Reality	3
SEN 655	Computational Linguistics and Machine Translation	3
Thesis or Graduate Project		**6**
SEN 695	Student Seminar	1
SEN 697	Graduate Project	2
SEN 699	Graduate Thesis	6

COURSE DESCRIPTION (CORE COURSE)

COURSE 1: SOFTWARE ENGINEERING CONCEPTS		
Course No: SEN 545	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
Introduction to the principles of software engineering. This course introduce the concept of Software Marketing, Business process reengineering, requirement engineering, Software process, Project management and planning; tracking and scheduling; risk analysis and quality assurance techniques; Configuration management. Software testing techniques: black box and white box techniques. Testing strategy: unit, integration and system testing. The course focuses on real world problem and try to develop solutions to help the software firms and individuals to grow.

Course Objectives:
The main objective of this course is to introduce students with the software engineering area and provides fundamental knowledge of multiple discipline of Software Engineering domain. However, the key objectives are following:

- To understand the background of Software Engineering
- To understand the software process and process management
- To be familiar with the concept of software marketing
- To understand the concept of requirement analysis
- To understand the concept of software architecture
- To understand the concept of Software testing and quality issues.
- To understand the concept of Software project management.

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

- Software Engineering: A Practitioner's Approach, Roger S Pressman.
- Software Engineering, Ian Sommerville.

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
Introduction Introduction to Software Engineering Software Process: Generic view	1. Students will be able to know each other 2. Students will learn about the course policy 3. Students will be able to plan for the exams 4. Students will learn the basic concept of Software Engineering.	Lecture & Cooperative Learning	N/A
Software Process: Process Models	1. Students will understand the concepts of the software process and software process models. 2. Students will be introduced to three generic software process models and when they might be used. 3. Students will learn about the fundamental process activities of software requirements engineering, software development, and evolution. 4. Students will learn why process should be organized to cope with changes in software requirement and design. 5. Students will learn how the Rational Unified Process integrates good software engineering practices to create adaptable software process	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Software Marketing	1. To understand basics of software marketing 2. To learn how to conduct market research. 3. Understand and analyze your market environment and determine your target market. 4. Determine position of product. 5. To educate the 4 P's (product, price, place & promotion) and how to apply them to marketing. 6. To show you the basics on how to create marketing plan.	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Software Requirement Analysis Requirements	1. Students will learn the concepts of user and system requirements and why these requirements should be written in different ways.	Lecturer, Discussing, Case Study	Formative Assessment, Presentation

Engineering	2. Students will learn the differences between functional and nonfunctional software requirements. 3. Students will learn how requirements maybe organized in a software requirement document. 4. Students will learn the principal requirement engineering activities of elicitation, analysis and validation and the relationship between these activities.		& Written Test
Architectural Design Object Oriented Design and Software Reuse	1. Students will learn how graphical models can be used to represent software systems. 2. Students will learn why different types of model are required and the fundamental system modeling perspectives of context, interaction, structure and behavior.	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Verification and Validation	1. Students will learn the difference between validation and Verification. 2. Students will learn why verification is important during the software development process.	Lecturer, Discussing, Case Study	Written Test, Case Study
Software Testing	1. Students will learn about the stages of testing during development to acceptance testing by system customers. 2. Students will learn techniques that will help them to choose test cases that are geared to discovering program defects. 3. Students will learn about test-first development, where they will design tests before writing code and run these tests automatically. 4. Students will know about the important difference between component, system and release testing and be aware of user testing process and techniques.	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Project Management	1. Students will learn about the principal tasks of software project managers. 2. Students will be introduced to the notion of risk management and some of the risks that can arise in software projects. 3. Students will learn about key issues that influences team working, such as team composition, organization and communication. 4. Students will learn about the fundamentals of software development planning process. 5. Students will be learning what sections should be included in a project plan that is created within a plan-driven development process. 6. Students will be able to quality management process and will know why it's important. 7. Students will learn why the software quality is effected by the software development process used. 8. Students will learn about the importance of standards in the quality management process and how standards are used in quality assurance.	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course outcome (CO):

C01: Explain and apply the main aspects of software engineering

CO2: Evaluate requirements for a software system and apply the process of analysis and design using the object-oriented approach.

CO3: Employ group working skills including general organization, planning and time management and inter-group negotiation.

CO4: Translate a requirements specification into an implementable design, following a structured and organized process.

CO5: Formulate a testing strategy for a software system, employing techniques such as unit testing, test driven development and functional testing.

Table 6: SEN545 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3	CO 4	CO 5
PLO 1: Knowledge	✓	✓			
PLO 2: Requirement Analysis		✓		✓	
PLO 3: Problem Analysis		✓			
PLO 4: Design		✓		✓	
PLO 5: Problem Solving		✓			✓
PLO 6: Implementation				✓	✓
PLO 7: Experiment and Analysis					✓
PLO 8: Community Engagement & Engg.				✓	
PLO 9: Teamwork			✓		
PLO 10: Communication			✓		
PLO 11: Self-directed			✓		
PLO 12: Ethics			✓		
PLO 13: Process Management			✓	✓	✓

COURSE 2: SOFTWARE ARCHITECTURE AND COMPONENT-BASED DESIGN		
Course No: SEN565	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
High-level architectural designs of software systems and products, strengths and weaknesses of each design style, component-based design, cohesion, interconnection and complexity, middleware, performance analysis and simulation, and COTS components; commonly-used software system structures, techniques for designing and implementing these structures, models and formal notations for characterizing and reasoning about architectures, tools for generating specific instances of an architecture, and case studies of actual system architectures Object-oriented design, design patterns, and UML; application of domain analysis, impact of platform dependence and independence, relation of software architecture to requirements, domain analysis and the architectural design process, and products in a real-world context.

Course Policy:
Regular class attendance is mandatory.
All course materials will be available at Piazza, the online forum, for students to follow.
Students should take tutorials with the instructor during the office hours, otherwise prior appointment is required.
Students must join the common online forum (detail instructions are provided in the class) within the first week of the semester.
Plagiarism – that is, the presentation of another person's project, homework, ideas, thoughts or words as though they were the student's own – must be strictly avoided. Cheating and plagiarism on exam and assignments are unacceptable and will be penalized.
Late assignments are discouraged and will be penalized.
Without proper excuse, no make-up quizzes will be arranged for the students who will fail to attend the quizzes on the scheduled dates.
Students must maintain the IUB code of conduct and ethical guidelines offered by the School of Engineering and Computer Science

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

- Software Architecture in Practice, 3rd Edition; Bass, Clements & Kazman
- Object-Oriented Software Engineering Using UML, Patterns, and Java, 3rd Edition Bernd Bruegge, Allen H. Dutoit

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
Introduction to Software Architecture	What is Software Architecture? Architectural Structures and Views Architectural Patterns What Makes a "Good" Architecture?	Lecture & Cooperative Learning	N/A
System Design: System Decomposition	Why is Software Architecture Important? Contexts of Software Architecture Architecture in a Technical Context Architecture in a Project Life-Cycle Context Architecture in a Business Context Architecture in a Professional Context Stakeholders How Is Architecture Influenced? What Do Architectures Influence? Overview of Sys Overview of System Design Design Goals Subsystem Decomposition, Architectural Styles tem Design Design Goals Subsystem Decomposition, Architectural Styles	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Patterns and Tactics	What is a Pattern? Pattern Catalogue Module patterns Component and Connector Patterns Allocation Patterns Relation Between Tactics and Patterns Using tactics together	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Architecture and system quality	Availability Interoperability	Lecturer, Discussing,	Formative Assessment,

attributes	Modifiability Performance Security Testability Usability	Case Study	Presentation & Written Test
Documenting Software Architectures	Uses and Audiences for Architecture Documentation Notations for Architecture Documentation Views Choosing the Views Combining Views Building the Documentation Package Documenting Behavior Architecture Documentation and Quality Attributes Documenting Architectures That Change Faster Than You Can Document Them Documenting Architecture in an Agile Development Project	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Architectures in Agile Projects Architecture and Requirements Designing an Architecture	How Much Architecture? Agility and Architecture Methods A Brief Example of Agile Architecting Guidelines for the Agile Architect Architecture and Requirements Gathering ASRs from Requirements Documents Gathering ASRs by Interviewing Stakeholders Gathering ASRs by Understanding the Business Goals Capturing ASRs in a Utility Tree Tying the Methods Together Designing an Architecture Design Strategy The Attribute-Driven Design Method The Steps of ADD	Lecturer, Discussing, Case Study	Written Test, Case Study
Architecture, Implementation, and Testing & Architecture Reconstruction and Conformance	Architecture, Implementation, and Testing Architecture and Implementation Architecture and Testing Architecture Reconstruction and Conformance Architecture Reconstruction Process Raw View Extraction Database Construction View Fusion Architecture Analysis: Finding Violations Guidelines	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Architecture Evaluation Architecture and Product Lines	Evaluation Factors The Architecture Tradeoff Analysis Method (ATAM) Lightweight Architecture Evaluation An Example of Product Line Variability What Makes a Software Product Line Work? Product Line Scope The Quality Attribute of Variability The Role of a Product Line Architecture Variation Mechanisms Evaluating a Product Line Architecture Key Software Product Line Issues	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Architectures for the Cloud	Basic Cloud Definitions Service Models and Deployment Options Economic Justification Base Mechanisms Sample Technologies Architecting in a Cloud Environment	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course Outcome (CO):

CO1: Use well-understood paradigms for designing new systems.

CO2: Generate architectural alternatives for a problem and selection among them.

CO3: Describe a software architecture using various documentation approaches and architectural description languages.

CO4: Recognize major software architectural styles, design patterns, and frameworks.

CO5: Design and motivate software architecture for large-scale software systems.

CO6: Argue the importance and role of software architecture in large-scale software systems.

CO7: Discuss and evaluate the current trends and technologies such as model-driven and service-oriented architectures.

Table 7: SEN565 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3	CO 4	CO 5	CO 6	CO 7
PLO 1: Knowledge	✓		✓	✓		✓	✓
PLO 2: Requirement Analysis	✓	✓			✓		✓
PLO 3: Problem Analysis		✓			✓		✓
PLO 4: Design		✓	✓	✓	✓		
PLO 5: Problem Solving		✓			✓		
PLO 6: Implementation	✓	✓			✓		
PLO 7: Experiment and Analysis		✓					✓
PLO 8: Community Engagement & Engg.						✓	
PLO 9: Teamwork		✓					
PLO 10: Communication			✓			✓	
PLO 11: Self-directed			✓				
PLO 12: Ethics						✓	
PLO 13: Process Management	✓					✓	

COURSE 3: SOFTWARE REQUIREMENT ENGINEERING		
Course No: SEN570	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
Role of requirements in system development and maintenance, goals of the requirements phase, essential difficulties of specifying requirements, effective methods, tools and techniques, techniques for formally modeling and specifying software requirements, process of identifying stakeholders, capturing, analyzing, reviewing and verifying their requirements for new or extended software products, optimization of software requirements and business benefits, the role of prototyping in validating requirements; process of requirements management, configuration management, change management, impact estimation due to requirements change..

Course Policy:
Regular class attendance is mandatory.
All course materials will be available at Piazza, the online forum, for students to follow.
Students should take tutorials with the instructor during the office hours, otherwise prior appointment is required.
Students must join the common online forum (detail instructions are provided in the class) within the first week of the semester.
Plagiarism – that is, the presentation of another person's project, homework, ideas, thoughts or words as though they were the student's own – must be strictly avoided. Cheating and plagiarism on exam and assignments are unacceptable and will be penalized.
Late assignments are discouraged and will be penalized.
Without proper excuse, no make-up quizzes will be arranged for the students who will fail to attend the quizzes on the scheduled dates.
Students must maintain the IUB code of conduct and ethical guidelines offered by the School of Engineering and Computer Science

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

- Karl, Wiegers. "Software requirements." *Microsoft Pr* (2003)..
- Hull, Elizabeth, Ken Jackson, and Jeremy Dick. *Requirements engineering*. Springer Science & Business Media, 2010.

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
Introduction to Software Requirement Engineering	Introduction of Requirement Engineering Introduction to Systems Engineering Requirements and Quality Requirements and the Lifecycle Requirements Traceability Requirements and Modelling Requirements and Testing Requirements in the Problem and Solution Domains	Lecture & Cooperative Learning	N/A
Requirement Engineering Process and methodologies	Developing Systems Generic Process Context Generic Process Introduction Generic Process Information Model Generic Process Details	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Software Requirement types and its analysis framework	Software Requirement and types Identify the rational of the types Reason to categorization of the types Requirement analysis framework	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Software Requirement Specification (SRS) and its content	Requirements for Requirements Structuring Requirements Documents Key Requirements Key attricbutes Ensuring Consistency Across Requirements Value of a Requirement The Language of Requirements Granularity of Requirements Criteria for Writing Requirements Statements	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
CMMI DEV 1.3 Practice and Work product for Requirement Development	What is Requirement Development-RD Activities and Steps of RD CMMI Goals and Practices for RD	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Concept of	What is change of requirement?	Lecturer,	Written Test,

Requirement change and its management	How to handle change requirement? Agility of the requirement? Change impact analysis Change requirement management	Discussing, Case Study	Case Study
Requirement Traceability and its application	Elementary Traceability Satisfaction Arguments Requirements Allocation Reviewing Traceability The Language of Satisfaction Arguments Rich Traceability Analysis Rich Traceability for Qualification Implementing Rich Traceability Design Documents Metrics for Traceability	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Validating the requirements	Introduction to validation Requirement validation Methodology Steps of requirement validation	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Requirement Reuse	Concept of reuse How to reuse the requirements? Capture and store requirements in cluster Application of requirement reuse	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Software Requirement and risk management	Introduction to risk Management Requirements Management Problems associated with risk management Managing Requirements in an Acquisition Organization Supplier Organizations Product Organizations	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course Outcome (CO):

CO1: Understand the requirement and its importance

CO2: Process of Requirement.

CO3: Analyze requirement.

CO4: Creating software requirement specification.

CO5: Requirement Development and CMMI.

CO6: Change requirement management.

CO7: Requirement traceability and verification.

Table 8: SEN 570 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3	CO 4	CO 5	CO 6	CO 7
PLO 1: Knowledge	✓		✓	✓		✓	✓
PLO 2: Requirement Analysis	✓	✓			✓		✓
PLO 3: Problem Analysis		✓			✓		✓
PLO 4: Design		✓	✓	✓	✓		
PLO 5: Problem Solving		✓			✓		
PLO 6: Implementation	✓	✓			✓		
PLO 7: Experiment and Analysis		✓					✓
PLO 8: Community Engagement & Engg.						✓	
PLO 9: Teamwork		✓					
PLO 10: Communication			✓			✓	
PLO 11: Self-directed			✓				
PLO 12: Ethics				✓		✓	
PLO 13: Process Management	✓	✓				✓	✓

COURSE 4: SOFTWARE PROJECT MANAGEMENT		
Course No: SEN645	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
Overview of Project Management. Project tracking and scheduling. Risk management and analysis. Cost estimation models. Project metrics. Function Point Estimation. Software quality assurance. Program verification and validation techniques Software testing techniques, black-box and white-box techniques. .Testing of various areas: unit, domain, path, equivalent class based portion, component, aggregation, system testing, requirement based testing, acceptance testing. Software reuse and maintenance; Industrial practices in software engineering. . ISO certification standards for software quality assurance; Software capability maturity model and its impact. The course focuses on taking a group development project from beginning to end.

Course Policy:
Regular class attendance is mandatory.
All course materials will be available at Piazza, the online forum, for students to follow.
Students should take tutorials with the instructor during the office hours, otherwise prior appointment is required.
Students must join the common online forum (detail instructions are provided in the class) within the first week of the semester.
Plagiarism – that is, the presentation of another person's project, homework, ideas, thoughts or words as though they were the student's own – must be strictly avoided. Cheating and plagiarism on exam and assignments are unacceptable and will be penalized.
Late assignments are discouraged and will be penalized.
Without proper excuse, no make-up quizzes will be arranged for the students who will fail to attend the quizzes on the scheduled dates.
Students must maintain the IUB code of conduct and ethical guidelines offered by the School of Engineering and Computer Science

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

- Gray, C.F. and Larson, E.W., (2003), Project Management – The Managerial Process, 2nd Ed., McGraw Hill Higher Education, NY.
- A Guide to the Project Management Body of Knowledge, 3rd Ed., PMBOK® Guide (2004), Project Management Institute, Global Standard.

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
Introduction to Project Management	Process, Project Triple Constraints Project Lifecycle	Lecture & Cooperative Learning	N/A
Project Initiation	Project Charter, Scope, Definition Aligning Projects with Strategic Planning	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Project Organization, Structure and Culture	Functional vs project vs matrix organization	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Estimating Project Times & Costs	Cost Estimating Life Cycle Costing Learning Curves Work Breakdown Structure	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Project Planning	Network diagram Case study understanding	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Resource Scheduling & Budgeting	Resource Constraints Resource Allocation Methods Splitting / Multitasking	Lecturer, Discussing, Case Study	Written Test, Case Study
Reducing Project Duration	Work with Gantt chart WBS control Resource allocation	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Project Control	Project Control Cycle Earned Value Approach Variance Analysis	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Managing Risk	Risk Identification	Lecturer,	Written Test,

	Risk Assessment Response Development Response Control	Discussing, Case Study	Case Study, Assignment
Leadership & Project Team	Leadership Team development Project team management	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Project Closeout & Review	Managing the Close-Out Process Project Audit Process Evaluations	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course Outcome (CO):

CO1: Understanding of Project management

CO2: Phases of project management

CO3: Project control parameters and issues

Table 9: SEN 645 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3
PLO 1: Knowledge	✓		
PLO 2: Requirement Analysis	✓	✓	
PLO 3: Problem Analysis		✓	
PLO 4: Design		✓	✓
PLO 5: Problem Solving	✓		
PLO 6: Implementation			✓
PLO 7: Experiment and Analysis			✓
PLO 8: Community Engagement & Engg.			
PLO 9: Teamwork		✓	
PLO 10: Communication	✓	✓	✓
PLO 11: Self-directed			✓
PLO 12: Ethics			
PLO 13: Process Management	✓	✓	✓

COURSE 5: SOFTWARE ENGINEERING PROCESS		
Course No: SEN647	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
Challenge of producing and maintaining complex software-intensive system; predictability and improved cost; team work in software development; quality assurance; process centric software engineering practices; software engineering process framework developed by Software Engineering Institute (SEI); capability of each process area; framework to meet challenges; characteristics of software products and processes, its quantification, analysis, prediction, control, and guidelines to achieve both business and technical goals.

Course Policy:
Regular class attendance is mandatory.
All course materials will be available at Piazza, the online forum, for students to follow.
Students should take tutorials with the instructor during the office hours, otherwise prior appointment is required.
Students must join the common online forum (detail instructions are provided in the class) within the first week of the semester.
Plagiarism – that is, the presentation of another person's project, homework, ideas, thoughts or words as though they were the student's own – must be strictly avoided. Cheating and plagiarism on exam and assignments are unacceptable and will be penalized.
Late assignments are discouraged and will be penalized.
Without proper excuse, no make-up quizzes will be arranged for the students who will fail to attend the quizzes on the scheduled dates.
Students must maintain the IUB code of conduct and ethical guidelines offered by the School of Engineering and Computer Science

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

-

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
		Lecture & Cooperative Learning	N/A
		Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
		Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
		Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
		Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
		Lecturer, Discussing, Case Study	Written Test, Case Study
		Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
		Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
		Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
		Lecturer,	Written Test,

		Discussing, Case Study	Case Study, Assignment
		Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course Outcomes (CO):

CO1: Cooperate in and contribute to a team environment, develop team dynamics, work according to an agreed team protocol, and resolve/manage conflict issues.

CO2: Identify, analyze, compare and contrast different processes and their assistive tools for selected phases of the software engineering life cycle.

CO3: Utilize various software engineering processes and their tools as required for best-practice development of software systems.

CO4: Plan, identify and apply processes, standards and tools for phases of a software engineering life cycle for a substantial software development project.

Table 10: SEN 647 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3	CO 4
PLO 1: Knowledge		✓		
PLO 2: Requirement Analysis		✓		✓
PLO 3: Problem Analysis		✓		✓
PLO 4: Design			✓	✓
PLO 5: Problem Solving			✓	
PLO 6: Implementation			✓	✓
PLO 7: Experiment and Analysis		✓		
PLO 8: Community Engagement & Engg.		✓		
PLO 9: Teamwork	✓			✓
PLO 10: Communication	✓			
PLO 11: Self-directed	✓			
PLO 12: Ethics	✓		✓	
PLO 13: Process Management	✓			✓

COURSE 6: SOFTWARE MARKETING		
Course No: SEN650	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
Introduction to marketing and marketing communication, service marketing, marketing challenges of technology products and services; methodology to identify target buyer's perceptions and behaviors; marketing software product: market research, product positioning, determining value proposition of the product, pricing, distribution, promotions, intellectual property management, advertising and product management; marketing customized software development services: target market determination, educating clients, response to request for proposal, basics of software effort and cost estimation, client relationship management, business value determination of software solutions, base line data collection, determining return on software investment (ROI), impact assessment due to software use, and organization change management; marketing software engineering services to offshore clients: determining offshore market opportunities, understanding constraints in penetrating those markets, developing strategies, establishing linkages with complementary partners and packaging and promoting software engineering service capability to targeted markets; Software quality assurance: basic of software quality assurance (SQA), business value of SQA, and clients role in SQA. Introduction to ISO and SEI's capability maturity model for software SQA. Case studies. Contractual and legal issues.

Course Policy:
Regular class attendance is mandatory.
All course materials will be available at Piazza, the online forum, for students to follow.
Students should take tutorials with the instructor during the office hours, otherwise prior appointment is required.
Students must join the common online forum (detail instructions are provided in the class) within the first week of the semester.
Plagiarism – that is, the presentation of another person's project, homework, ideas, thoughts or words as though they were the student's own – must be strictly avoided. Cheating and plagiarism on exam and assignments are unacceptable and will be penalized.
Late assignments are discouraged and will be penalized.
Without proper excuse, no make-up quizzes will be arranged for the students who will fail to attend the quizzes on the scheduled dates.
Students must maintain the IUB code of conduct and ethical guidelines offered by the School of Engineering and Computer Science

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

- Software That Sells: A Practical Guide to Developing and Marketing Your Software Project by Edward Hasted
- Just Enough Requirements Management: Where Software Development Meets Marketing by Alan Mark Davis

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
Introduction to software Marketing	1. Students will be able to know each other 2. Students will learn about the course policy 3. Students will be able to plan for the exams 4. Students will learn the basic concept of Software marketing	Lecture & Cooperative Learning	N/A
What is marketing	1. To understand basics of software marketing 2. To educate the 4 P's (product, price, place & promotion) and how to apply them to marketing. 3. Demands, wants and needs 4. Goods services and ideas	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Conducting Market Research	1. Source of Data 2. Secondary Data 3. Primary Data 4. Research Approaches	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Analyzing the market environment	1. Analyzing company 2. distribution Channel 3. Analyze customer 4. Competitive forces 5. SEPTE Analysis	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Selecting your target market	1. Consumer and business 2. Geographic location, demographic variables. 3. Psychographics of market 4. VALS	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Market Positioning	1. Introduction of Positioning	Lecturer,	Written Test,

	2. How to positioning?	Discussing, Case Study	Case Study
Designing product and services	1. Define product 2. Product mix 3. Product life cycle	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Pricing Product and Services	1. Objective 2. decision 3. pricing Strategies 4. Factors	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Placing Product and Services	1. Placing Decision. 2. Market coverage 3. Channel 4. Supply chain	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Promoting Product and Services	1. Promotional Mix 2. Various promotional mix 3. Publicity Tactics.	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Creating a Market Plan	1. Prepare plan 2. SWOT Analysis	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course Outcomes (CO):

CO1: Appreciate the global nature of software marketing and take appropriate measures to operate effectively in international settings.

CO2: Develop marketing strategies based on product, price, place and promotion objectives.

CO3: Comprehend the social, legal, ethical and technological forces behind software marketing decision-making.

CO4: Formulate software marketing strategies that incorporate psychological and sociological factors which influence consumers.

CO5: Understand and use the standardized models for software quality assurance.

Table 11: SEN 650 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3	CO 4	CO 5
PLO 1: Knowledge	✓		✓		✓
PLO 2: Requirement Analysis	✓	✓			
PLO 3: Problem Analysis	✓	✓			
PLO 4: Design		✓		✓	
PLO 5: Problem Solving		✓		✓	✓
PLO 6: Implementation		✓		✓	✓
PLO 7: Experiment and Analysis				✓	
PLO 8: Community Engagement & Engg.			✓	✓	
PLO 9: Teamwork					
PLO 10: Communication			✓		
PLO 11: Self-directed					
PLO 12: Ethics			✓	✓	
PLO 13: Process Management		✓			

COURSE 7: SOFTWARE QUALITY ASSURANCE, TESTING AND RELIABILITY		
Course No: SEN653	**Credit: 3**	**Contact Hours: 180 mins (per class)**

Course Description:
Software quality assurance (SQA), review of SQA practices, quality management, the role of SQA, the SQA program planning, launching and management, independent verification and validation; software inspections, basic principles, reviews, reporting and tracking, managing inspections and reviews; principles of software testing, testing types, test planning, development, execution and reporting; real-time testing and test organization; basic concepts of reliability, modeling software reliability from test results, techniques for analyzing, predicting, designing, and engineering the required and expected reliability of software systems.

Course Policy:
Regular class attendance is mandatory.
All course materials will be available at Piazza, the online forum, for students to follow.
Students should take tutorials with the instructor during the office hours, otherwise prior appointment is required.
Students must join the common online forum (detail instructions are provided in the class) within the first week of the semester.
Plagiarism – that is, the presentation of another person's project, homework, ideas, thoughts or words as though they were the student's own – must be strictly avoided. Cheating and plagiarism on exam and assignments are unacceptable and will be penalized.
Late assignments are discouraged and will be penalized.
Without proper excuse, no make-up quizzes will be arranged for the students who will fail to attend the quizzes on the scheduled dates.
Students must maintain the IUB code of conduct and ethical guidelines offered by the School of Engineering and Computer Science

Required Text:
The course will be based mostly on the following books [some other books and journals may be referred time to time]:

- Software Quality Assurance from Theory to Implementation by Daniel Galin
- Software Quality Engineering: Testing, Quality Assurance, and Quantifiable Improvement by Jeff Tian

Topics	Learning Outcomes	Teaching Strategies	Assessment Strategies
Software Quality Control & Software Testing	Understand software quality Software Quality Control Software Testing Software Quality Assurance	Lecture & Cooperative Learning	N/A
Quality factors and SQA	Software Quality Factors; Components of SQA System; Contract Review; Development of Quality Plans	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
Testing life-cycle	Integrating quality activities in the project life cycle; Cost of Software Quality	Lecturer, Discussing, Case Study	Formative Assessment Presentation & Written Test
SQA	Software Quality Infrastructure Components: Procedures and Work Instructions; Supporting Quality Devices; Staff Training and Certification; Corrective and Preventive Actions; Configuration Management; Documentation Control	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
QA management	Organizing for Quality Assurance: Management and its Role in SQA; The SQA unit and Other Actors in the SQA	Lecturer, Discussing, Case Study	Formative Assessment, Presentation & Written Test
Software Testing	Testing: Concepts, Issues and Techniques	Lecturer, Discussing, Case Study	Written Test, Case Study
Software Testing	Test Activities, Management & Test Automation; Coverage & Usage testing based on Checklists and partitions;	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment
Software Testing	Testing Technique: Input Domain Partitioning & Boundary Testing; Coverage & Usage testing based on Finite-State Machines and Markov Chain;	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Software Testing	Testing Technique: Control Flow, Data Dependency and Interaction Testing; Testing Techniques: Adaptation, Specialization & Integration	Lecturer, Discussing, Case Study	Written Test, Case Study, Assignment

Course Outcomes (CO):

CO1: Manage incidents and risks within a project.

CO2: Create test strategies and plans, design test cases, prioritize and execute them.

CO3: Apply modern software testing processes in relation to software development and project management.

CO4: Contribute to efficient delivery of software solutions and implement improvements in the software development processes.

Table 12: SEN 653 Program Learning Outcome and Course Outcome Mapping

	CO 1	CO 2	CO 3	CO 4
PLO 1: Knowledge			✓	
PLO 2: Requirement Analysis		✓		
PLO 3: Problem Analysis		✓		
PLO 4: Design		✓		
PLO 5: Problem Solving		✓	✓	
PLO 6: Implementation		✓	✓	✓
PLO 7: Experiment and Analysis		✓	✓	
PLO 8: Community Engagement & Engg.			✓	
PLO 9: Teamwork	✓			
PLO 10: Communication	✓			
PLO 11: Self-directed	✓			
PLO 12: Ethics	✓			✓
PLO 13: Process Management	✓		✓	✓

COURSE DESCRIPTION (OPTIONAL)

SEN 505 USER INTERFACE DESIGN AND DEVELOPMENT:

Human-computer interaction and the importance of good interface design. Interface quality and methods of evaluation. Prototyping and implementation techniques. Task analysis and iterative design cycle. Dialog techniques, basic computer graphics, use of colour and sound. I/O device. Menus and their use. Command languages. Screen formatting. Natural language facilities. Case studies and project.

SEN 551 DATA MINING AND WAREHOUSING

Basic concept of data mining, issues and techniques. Data warehouse and OLTP technologies for data mining, Classification of data mining techniques and models, data pre-processing, data mining primitives, query languages and system architecture, characterization and comparison. Mining association rules in large database. Cluster analysis, multidimensional analysis and descriptive mining of complex data object. Data mining in distributed heterogeneous database systems. Data mining applications and future research issues.

SEN 555 ADVANCED INTERNET COMPUTING

Introduction to Internet Technology, web servers and HTTP, URLs, Forms and CGI, JavaScript, Cookies, Java and Servlets, Databases and ASP, JDBC. Markup languages - SGML, HTML, DHTML, XML, WML, their standards, Publishing information in XML and WML, Extracting product information and application development with XML or WML. Active server pages, IIS and PWS environment, ASP variables and control structures, data storage and access, ASP object models, Advanced data handling techniques. Application development using Java Scripts, Java applets, Java Servlets, Java Database Connectivity (JDBC). Introduction to PHP programming, variables and control structures, Database connectivity and Application development with PHP

SEN 601 DISTRIBUTED DATABASE SYSTEM

A detailed study of advanced topics related to relational database theory, query processing and optimisation, recovery techniques, concurrency control. Crash recovery. Distributed database systems: security and integrity. Other database paradigms such as deductive and object oriented issues. Heterogeneous databases

SEN 605 ANALYSIS AND ORGANIZATION OF INFORMATION SYSTEMS

Organization and representation of information and access to information; categorization, indexing and content analysis; use of codes, formats and standards. Analysis and evaluation of search and navigation techniques; project management and scheduling; analysis of information needs and systems requirements; design of alternatives. Quantitative methods and tools for decision making. Documentation management. Social and behavioural aspects of information production

SEN 620 IMAGE ANALYSIS AND PATTERN RECOGNITION

Digital image fundamentals; advanced image transformations: FFT, DTF, sine/cosine, Walsh, Hadamard, Haar, Slant, Wavelet transforms, Karhünen-Louève transformation; image enhancement: Low pass and high pass filters, homomorphic filters: image restoration: constrained least square restoration; time-varying image analysis: dynamic imagery algorithm, image motion detection, two dimension and three dimension motions; image compression: error-free compression, image compression standard; image segmentation: edge linking and boundary detection, region oriented segmentation: colour based image processing: colour perception, chromacity diagram, filtering colour images, colour tone adjustment, neural networks for colour classification; pattern recognition: pattern classes, decision theoretic models, neural networks, knowledge- based interpretation

SEN 630 ADVANCED TOPICS IN COMPUTER NETWORKING

Overview of OSI model, TCP/IP, Circuit switching and Packet switching, multiplexing, routing and congestion control and deadlock prevention. Driving forces for high speed networking, High speed LANs, Fast Ethernet and Gigabit Ethernet, FDDI and DQDB, Frame relay architecture, Standards and protocols, Switched Multi Megabit Data Services, ATM standards protocols, ATM LANs, Optical Communication and SONET/SDH, Broadband access technologies, x-DSL. BISDN protocol and architecture, Broadband service aspects and access architecture, Broadband transmission networks, Broadband intelligent network. Broadband access network technology, Encryption and network security, Advanced topics for network management.

SEN 646 Software Technology and Innovation Management:

Current practices of technology and innovation management, concept of continuous and discontinuous innovation management, technology and innovation life cycle management, technology and innovation management issues in software business; successful origination, development, implementation, and diffusion of product and process innovations in industry; effective organization and management of the technological change process (marketing, R&D, engineering, and manufacturing) in new ventures, multidivisional and multinational enterprises; management of innovation and change in clients organization due to adoption of software solutions; generating, screening and developing new ideas for innovating software solutions; management of software products and their positioning in competitive market place.

SEN 649 Business Process Reengineering (BPR) and Software:

Introduction to process centric understanding of business processes; analysis and modeling techniques of business practices as processes, analysis of processes of different categories of organizations operating both in private and public sectors; e-Governance, e-Commerce, process of generating software solutions for improving performance of business processes, process performance parameter determination and determination of optimum software features for performance improvement; ROI estimation and measurement framework determination for verification; over view of different enterprise resource planning (ERP) applications and their uses in BPR; implementation and organizational change management; process performance parameter collection and validation of ROI estimation.

SEN 651 Embedded and Real-time Software Engineering

Basic concepts, terminology, and problems of real-time computing; software solutions to real-time problems; scheduling, specification of system requirements and design, real-time software architectures, languages and operating systems for real-time computing, real-time problems in a distributed processing system, and hardware-software interfaces; testing, verification and validation of real-time software; introduction to machine intelligence, signals, sensors and systems, data acquisition, data interpretation techniques, embedded operating systems, assembly and C programming, development and testing tools for embedded applications.