Course Curriculum Schema

Semester 1

S. No.	Course Code	Course Name	Credits	L-T-P
1	MA101	Calculus	3	3-0-0
2	PH101	Physics	3	3-0-0
3	PX101	Physics Lab	2	0-0-2
4	CS101	Introduction to Programming	4	4-0-0
5	CX101	Introduction to Programming Lab	2	0-0-2
6	HS101	English Communication	3	0-3-0
7	IX101	Innovation Lab I	6	0-0-6
8	UC1004	Yoga Education & Practices	2	0-0-2
		Total	**25**	

Semester 2

S. No.	Course Code	Course Name	Credits	L-T-P
1	MA102	Probability & Statistics	3	3-0-0
2	EE101	EE Fundamentals	3	3-0-0
3	EX101	EE Fundamentals Lab	2	0-0-2
4	CS102	Data Structures	4	4-0-0
5	CX102	Data Structures Lab	2	0-0-2
6	HS102	Professional Communication	3	0-3-0
7	IX101	Innovation Lab I	6	0-0-6
8	UC2004	Environment Studies	2	2-0-0
		Total	**25**	

Semester 3

S. No.	Course Code	Course Name	Credits	L-T-P
1	MA201	Discrete Mathematical Structures	3	3-0-0
2	CS201	Digital Logic Design	3	3-0-0
3	CX201	Digital Logic Design Lab	2	0-0-2
4	CS202	Design & Analysis of Algorithms	3	3-0-0
5	CX202	Design & Analysis of Algo. Lab	2	0-0-2
6	CS203	Theory of Computation	3	3-0-0
7	MJ301	Major Specialization I	3	3-0-0
8	MN301	Exploratory Minor	2	2-0-0
9	IX201	Innovation Lab II	4	0-0-4
		Total	**25**	

Semester 4

S. No.	Course Code	Course Name	Credits	L-T-P
1	MA202	Linear Algebra & Optimization	3	3-0-0
2	CS204	Computer Organization & Architecture	3	3-0-0
3	CS205	Operating Systems	3	3-0-0
4	CX205	Operating Systems Lab	2	0-0-2
5	CS206	Programming Methodology	3	3-0-0
6	CX206	Programming Methodology Lab	2	0-0-2
7	MJ401	Major Specialization II	3	3-0-0
8	MN401	Exploratory Minor	2	2-0-0
9	IX202	Innovation Lab III	4	0-0-4
		Total	**25**	

Semester 5

S. No.	Course Code	Course Name	Credits	L-T-P
1	CS301	Computer Networks	3	3-0-0
2	CX301	Computer Networks Lab	2	0-0-2
3	CS302	Compiler Design	3	3-0-0
4	MJ501	Major Specialization III	3	3-0-0
5	MJ502	Major Specialization IV	3	3-0-0
6	MN501	Exploratory Minor	2	2-0-0
7	IX301	Innovation Lab IV	4	0-0-4
		Total	**20**	

Semester 6

S. No.	Course Code	Course Name	Credits	L-T-P
1	CS303	Software Engineering	3	3-0-0
2	CS304	Database Management Systems	3	3-0-0
3	CX304	Database Management Systems Lab	2	0-0-2
4	MJ601	Major Specialization V	3	3-0-0
5	MJ602	Major Specialization VI	3	3-0-0
6	MN601	Exploratory Minor	2	2-0-0
7	IX301	Innovation Lab IV	4	0-0-4
		Total	**20**	

Semester 7

S. No.	Course Code	Course Name	Credits	L-T-P
1	MJ701	Advance Elective I	4	4-0-0
2	MJ702	Advance Elective II	4	4-0-0
3	MN701	Advanced Minor	2	2-0-0
4	OE701	Open Elective I	3	3-0-0
5	OE702	Open Elective II	2	2-0-0
6	SM401	Technical Seminar	2	0-2-0
		Total	**17**	

Semester 8

S. No.	Course Code	Course Name	Credits	L-T-P
1	MJ801	Advance Elective III	4	4-0-0
2	MJ802	Advance Elective IV	4	4-0-0
3	MN801	Advanced Minor	2	2-0-0
4	OE801	Open Elective III	3	3-0-0
5	OE802	Open Elective IV	2	2-0-0
6	PR402	Major Project	8	0-0-8
		Total	**23**	

**SCE = Software and Cloud Engineering*

Major Specialization I (Semester 3)

S. No.	Track	Course Name	Credits	L-T-P
1	SCE	Object-Oriented Analysis & Design	3	2-0-2
2	Artificial Intelligence	Fundamentals of AI & Game Theory	3	3-0-0
3	Data Science	Advanced Statistics for Data Science	3	2-1-0
4	Cyber Security	Information Security Fundamentals	3	3-0-0
5	Robotics	Mechanics of Robotics	3	2-1-0

Major Specialization II (Semester 4)

S. No.	Track	Course Name	Credits	L-T-P
1	SCE	Software Architecture	3	2-0-2
2	Artificial Intelligence	Introduction to Machine Learning	3	2-0-2
3	Data Science	Advanced SQL & NoSQL Databases	3	2-0-2
4	Cyber Security	Cryptography & Network Security	3	2-1-0
5	Robotics	Sensors, Actuators & Signal Processing	3	2-0-2

Major Specialization III (Semester 5)

S. No.	Track	Course Name	Credits	L-T-P
1	SCE	Cloud Computing Fundamentals	3	3-0-0
2	Artificial Intelligence	Deep Learning & Neural Networks	3	2-0-2
3	Data Science	Data Visualization & Storytelling	3	2-0-2
4	Cyber Security	Secure Network Protocols	3	3-0-0
5	Robotics	Embedded Systems & Microcontrollers	3	2-0-2

Major Specialization IV (Semester 5)

S. No.	Track	Course Name	Credits	L-T-P
1	SCE	Software Testing & Quality Assurance	3	2-0-2
2	Artificial Intelligence	Genetic Algorithms	3	2-0-2
3	Data Science	Big Data Technologies	3	2-0-2
4	Cyber Security	Ethical Hacking & Penetration Testing	3	2-0-2
5	Robotics	Robot Operating System (ROS)	3	2-0-2

Major Specialization V (Semester 6)

S. No.	Track	Course Name	Credits	L-T-P
1	SCE	Distributed Systems & Microservices	3	3-0-0
2	Artificial Intelligence	Natural Language Processing (NLP)	3	2-0-2
3	Data Science	Predictive Modeling & Analytics	3	2-0-2
4	Cyber Security	Advanced Cryptography	3	3-0-0
5	Robotics	Communication & IoT	3	2-0-2

Major Specialization VI (Semester 6)

S. No.	Track	Course Name	Credits	L-T-P
1	SCE	Building Scalable Cloud Services	3	2-0-2
2	Artificial Intelligence	Computer Vision	3	2-0-2
3	Data Science	Time Series Analysis & Forecasting	3	2-1-0
4	Cyber Security	Web & Application Security	3	2-0-2
5	Robotics	Autonomous Navigation	3	2-0-2

Advanced Electives Pool

*Students must select **4 courses total** from their respective track's pool to complete their Major requirements. Typically, students select 2 courses in Semester 7 and 2 courses in Semester 8.*

Track 1: Software and Cloud Engineering Elective Pool

Code	Course Name	Credits	L-T-P
SE-EL1	DevOps & Site Reliability Engineering	4	3-0-2
SE-EL2	Agile Project Governance & Management	4	3-1-0
SE-EL3	Serverless Architecture	4	3-0-2
SE-EL4	Secure Software Development (DevSecOps)	4	3-0-2
SE-EL5	Enterprise Application Architecture	4	3-1-0
SE-EL6	Software Metrics & Quality Engineering	4	4-0-0

Track 2: Artificial Intelligence Elective Pool

Code	Course Name	Credits	L-T-P
ML-EL1	Reinforcement Learning	4	3-0-2
ML-EL2	MLOps & Production AI	4	3-0-2
ML-EL3	Generative AI & Large Language Models	4	3-0-2
ML-EL4	Multi-Agent Systems	4	4-0-0
ML-EL5	Explainable AI (XAI) & AI Governance	4	4-0-0
ML-EL6	Probabilistic Graphical Models	4	3-1-0

Track 3: Data Science Elective Pool

Code	Course Name	Credits	L-T-P
DS-EL1	High-Dimensional Data Analysis	4	3-1-0
DS-EL2	Business Intelligence & Data Warehousing	4	3-0-2
DS-EL3	Mining Massive Datasets	4	3-0-2
DS-EL4	Data Governance, Privacy & Ethics	4	4-0-0
DS-EL5	Real-Time Analytics & Stream Processing	4	3-0-2
DS-EL6	Graph Mining & Social Network Analysis	4	3-0-2

Track 4: Cyber Security Elective Pool

Code	Course Name	Credits	L-T-P
CY-EL1	Digital Forensics & Incident Response	4	3-0-2
CY-EL2	Cloud Security Architecture	4	3-0-2
CY-EL3	Malware Analysis & Reverse Engineering	4	3-0-2
CY-EL4	Blockchain & Decentralized Identity	4	3-0-2
CY-EL5	Mobile & Wireless Security	4	3-0-2
CY-EL6	Cyber Law, Risk & Compliance	4	4-0-0

Track 5: Robotics Elective Pool

Code	Course Name	Credits	L-T-P
RB-EL1	Advanced Control Systems	4	3-1-0
RB-EL2	Deep Learning for Robot Perception	4	3-0-2
RB-EL3	Swarm Intelligence & Multi-Robot Systems	4	3-0-2
RB-EL4	Human-Robot Interaction (HRI)	4	3-1-0
RB-EL5	Advanced Motion Planning	4	3-0-2
RB-EL6	Manipulation & Grasping Mechanisms	4	3-0-2

Minor Specialization Tracks

Students select 4 distinct domains to explore during Semesters 3-6 (Level 1). In Semesters 7 and 8, they select 2 of those domains to study at an advanced level (Level 2).

Exploratory Minor (Level 1)

The focus of Level 1 is to introduce the domain terminology, data structures, and fundamental challenges from a Computer Science perspective.

Domain Code	Domain Name	Level 1 Course Name	Credits
MN-BIO	Bioinformatics	Introduction to Bioinformatics	2 (2-0-0)
MN-MED	Biomedical	Biomedical Signals & Systems	2 (2-0-0)
MN-DES	Product Design	Human-Computer Interaction (HCI)	2 (2-0-0)
MN-MFG	Smart Manufacturing	IoT for Industry	2 (2-0-0)
MN-MAT	Nanomaterials	Computational Materials Science	2 (2-0-0)
MN-FIN	FinTech	Algorithmic Financial Engineering	2 (2-0-0)
MN-SEM	Semiconductor & RISC-V	Digital System Design	2 (2-0-0)
MN-GEO	Geoinformatics	Fundamentals of GIS & Spatial Analysis	2 (2-0-0)
MN-CIV	Smart Cities	Urban Informatics & Smart Infrastructure	2 (2-0-0)
MN-EVM	Mechatronics in EV	Fundamentals of Automotive Embedded Systems	2 (2-0-0)

Advanced Minor (Level 2)

The focus of Level 2 is the rigorous application of advanced CS concepts (AI, Big Data, Blockchain, Simulation) within the chosen domain.

Domain Code	Domain Name	Level 2 Course Name	Credits
MN-BIO-A	Bioinformatics	Genomic Data Science	2 (2-0-0)
MN-MED-A	Biomedical	Medical Image Analysis	2 (2-0-0)
MN-DES-A	Product Design	Spatial Computing & AR/VR	2 (2-0-0)
MN-MFG-A	Smart Manufacturing	Digital Twins & Cyber-Physical Systems	2 (2-0-0)
MN-MAT-A	Nanomaterials	Material Informatics	2 (2-0-0)
MN-FIN-A	FinTech	Blockchain & DeFi Architectures	2 (2-0-0)
MN-SEM-A	Semiconductor & RISC-V	SoC Design & Verification	2 (2-0-0)
MN-GEO-A	Geoinformatics	Geospatial Data Science & AI	2 (2-0-0)
MN-CIV-A	Smart Cities	AI & Big Data for Urban Systems	2 (2-0-0)
MN-EVM-A	Mechatronics in EV	Autonomous Driving & Vehicle Intelligence	2 (2-0-0)

Semester I

Calculus (MA101)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Differential Calculus: Review of Single Variable Calculus: Limits, Continuity, Differentiability, Mean Value Theorems (Rolle's, Lagrange's, and Cauchy's). Successive Differentiation: Leibniz's Theorem for the nth Derivative of a Product of Functions. Series Expansions: Taylor's and Maclaurin's Series for Functions of a Single Variable. Applications: Curve Tracing in Cartesian and Polar Coordinates, Indeterminate Forms (L'Hôpital's Rule).

Unit 2
Integral Calculus: Review of Single Variable Integration: Definite Integrals as a Limit of a Sum, Fundamental Theorem of Calculus. Advanced Integration: Reduction Formulae for Trigonometric Integrals. Improper Integrals: Beta and Gamma Functions, Their Properties and Relationship. Applications of Integration: Calculating Arc Lengths, Surface Areas, and Volumes of Solids of Revolution.

Unit 3
Multivariable Differential Calculus: Functions of Several Variables: Introduction to Functions of Two and Three Variables, Limits and Continuity. Partial Differentiation: First and Higher-Order Partial Derivatives, Total Derivative. Advanced Topics: Euler's Theorem on Homogeneous Functions, Chain Rule for Multiple Variables. Applications: Taylor's Series for Functions of Two Variables, Finding Maxima, Minima, and Saddle Points; Method of Lagrange Multipliers for Constrained Optimization.

Unit 4
Multivariable Integration Calculus: Multiple Integrals: Double Integrals, Evaluation, and Change of Order of Integration. Change of Variables: Transforming Integrals from Cartesian to Polar Coordinates. Triple Integrals: Evaluation in Cartesian, Cylindrical, and Spherical Coordinates. Applications: Using Double Integrals to Compute Areas and Triple Integrals to Compute the Volume of 3D Solids.

Unit 5
3D Coordinate Geometry (Solid Geometry): The Line: Direction Cosines and Direction Ratios, Cartesian and Vector Equations of a Line, Conditions for Coplanar Lines, Shortest Distance Between Two Skew Lines. The Plane: Cartesian and Vector Equations of a Plane, Angle Between Two Planes, Distance of a Point from a Plane. The Sphere: Equation of a Sphere, Finding Its Center and Radius, Equation of the Tangent Plane at a Point.

Physics (PH101)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Quantum Mechanics (Computing Foundations): Limitations of Classical Mechanics. Wave-Particle Duality, de Broglie's Hypothesis, and Heisenberg Uncertainty Principle. **The Schrödinger Equation:** Time-Dependent and Time-Independent forms. Physical Interpretation of the Wave Function (Ψ). Applications: Particle in a 1D Infinite Potential Well (Energy Quantization) and **Quantum Tunneling** (Concept behind Flash Memory and tunneling limits in transistors). Introduction to Quantum Superposition and Qubits.

Unit 2
Semiconductor Physics (Hardware Foundations): Band Theory of Solids: Conductors, Insulators, and Semiconductors. Intrinsic and Extrinsic Semiconductors (n-Type and p-Type), Fermi Levels. **Semiconductor Devices:** Physics of the P-N Junction, I-V Characteristics (Diode Logic), Zener Diodes, and Bipolar Junction Transistors (BJTs). Optoelectronics: Photodiodes, Solar Cells, and LEDs (Light Emitting Diodes).

Unit 3
Electromagnetism and Communication: Electrostatics and Magnetostatics: Divergence and Curl of fields. **Maxwell's Equations:** Integral and Differential forms and their physical significance (The logic of electromagnetic fields). EM Wave Propagation in Free Space. **Optical Fibers:** Total Internal Reflection, Numerical Aperture, and Signal Attenuation (Physics of the Internet backbone).

Unit 4
Oscillations and Signal Theory: Simple Harmonic Motion (SHM). Damped Oscillations (Signal decay). Forced Oscillations and **Resonance** (Critical for clock speeds and filter design). Wave Motion: Transverse and Longitudinal waves, Principle of Superposition (Interference). Diffraction and Polarization (Basics of LCD screens).

Unit 5
Mechanics for Simulation (Game Physics): Newton's Laws and Galilean Invariance. **Kinematics and Dynamics:** Rectilinear and Projectile Motion (Trajectory algorithms). Work-Energy Theorem and Conservation of Energy. Impulse and Momentum: Conservation of Momentum during Collisions (Elastic/Inelastic) and Impact (Physics Engine logic). Introduction to Special Relativity: Time Dilation and Length Contraction (GPS Synchronization logic).

Physics Lab (PX101)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Semiconductor Diodes Focuses on analyzing the V-I characteristics of a P-N Junction Diode in forward and reverse bias to understand the fundamental building block of logic gates and rectification.

Lab 2: Voltage Regulation Focuses on the breakdown mechanism of a Zener Diode and its application as a Voltage Regulator, simulating power stability in computer power supply units (PSUs).

Lab 3: Optoelectronics and Sensors Focuses on the characteristics of a Solar Cell or Photodiode, understanding how light energy is converted to electrical signals, applicable to optical sensors and energy harvesting.

Lab 4: Energy Band Gap Focuses on determining the Energy Band Gap of a semiconductor material using the Four-Probe Method, linking temperature changes to conductivity (hardware thermal management).

Lab 5: Quantum Constants Focuses on the determination of Planck's Constant using LEDs of different wavelengths, verifying the quantum mechanical relationship between energy and frequency.

Lab 6: Optical Fiber Communication Focuses on determining the Numerical Aperture (NA) and bending losses of an Optical Fiber, simulating data transmission efficiency and signal attenuation in network cables.

Lab 7: Wave Optics and Diffraction Focuses on determining the wavelength of a Laser source using a Diffraction Grating, illustrating wave interference principles used in holography and optical storage.

Lab 8: Electromagnetism Focuses on the variation of the Magnetic Field along the axis of a circular coil (Stewart-Gee's Experiment), visualizing field lines and electromagnetic induction principles.

Lab 9: Resonance and Signals Focuses on the study of standing waves and resonance frequency using Melde's Experiment, providing a physical analogy for signal processing and harmonic analysis.

Lab 10: Rigid Body Dynamics Focuses on determining the acceleration due to gravity (g) and Radius of Gyration using a Compound Pendulum, verifying physical laws used in rigid body physics engines.

Introduction to Programming (CS101)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
Problem Solving and Core Concepts: Introduction to Algorithms and Flowcharts, Structure of a C++ Program, Preprocessor directives, Console Input/Output (iostream), Fundamental Data Types and Literals, Variable Scope and Initialization, Type Conversions (Implicit/Explicit), Operators (Arithmetic, Relational, Logical), Control Flow (Conditionals, Loops, break/continue).

Unit 2
Compound Data Types and Memory: Pointers (Declaration, Dereferencing), Dynamic Memory Allocation (new/delete), Array structures (1D/2D C-style Arrays vs std::vector), String manipulation (C-style vs std::string), User-Defined Types (Structures for grouping data, Unions for memory sharing).

Unit 3
Modular Programming and OOP Fundamentals: Function definitions and Overloading, Parameter Passing mechanisms (Value, Pointer, Reference), Introduction to Classes and Objects, Encapsulation principles, Access Specifiers (public, private, protected), Object lifecycle management (Constructors, Destructors).

Unit 4
Advanced Object-Oriented Concepts: Inheritance types (Base and Derived classes), Code reusability, Polymorphism (Compile-time Overloading vs Run-time Overriding), Virtual Functions, Abstract Classes, Static Data Members, Exception Handling (try, catch, throw), Standard Exception hierarchy.

Unit 5
STL and Modern C++: Standard Template Library (Sequence Containers, Associative Containers, Container Adaptors), Iterators and Generic Algorithms (Sort, Find), Modern C++ features (Smart Pointers for memory safety, Lambda Expressions, Range-based loops), Utilities (std::pair, std::tuple).

Introduction to Programming (CS101)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Basic I/O and Arithmetic Focuses on the implementation of basic Input/Output operations and arithmetic logic to construct a Mini Calculator, establishing familiarity with the programming environment.

Lab 2: Control Flow Patterns Focuses on the application of conditional statements and looping constructs to generate complex visual patterns (Diamond, Pascal's Triangle) and process iterative data reports.

Lab 3: Pointers and Memory Management Focuses on direct memory manipulation using pointers, implementing dynamic allocation for structures, and performing efficient matrix operations.

Lab 4: Recursion and Combinatorics Focuses on the implementation of recursive algorithms to solve complex combinatorial problems (Subsets, Combination Sum) and array partitioning strategies.

Lab 5: Classes and Encapsulation Focuses on Object-Oriented fundamentals, implementing Classes and Objects to demonstrate data hiding, Constructor logic, and the Deep Copy mechanism.

Lab 6: Inheritance and Polymorphism Focuses on hierarchical class structures, implementing Inheritance for code reusability and Virtual Functions to achieve Runtime Polymorphism.

Lab 7: Advanced Class Features and Exceptions Focuses on robustness and design patterns, implementing Static members, the Singleton Pattern, and Exception Handling for error management.

Lab 8: The Standard Template Library (STL) Focuses on leveraging pre-built generic containers (Maps, Stacks, Queues) and standard algorithms to efficiently process and organize data.

Lab 9: Modern C++ Features Focuses on contemporary C++ practices, specifically implementing automatic memory management with Smart Pointers and using Lambda expressions for concise logic.

Lab 10: Capstone: File System Simulation Focuses on the integration of all learned concepts, designing a Command-Line File System simulation using OOP principles and Advanced Data Structures.

English Communication (HS101)

← Back to Scheme

Credits: 3 (0-3-0)

Course Content

Unit 1
Functional Grammar and Corrections: Remedial Grammar: Revisiting Tenses (Present, Past, Future) to ensure correct sentence timing. Subject-Verb Agreement drills. Articles (a, an, the) and Prepositions (in, on, at, by). Common Errors in English: Identifying and correcting "Indianisms" and translation errors. Sentence Structure: Simple, Compound, and Complex sentences.

Unit 2
Phonetics and Pronunciation: The Sounds of English: Introduction to Vowels and Consonants (Basic IPA). Syllables and Word Stress: Understanding where to place emphasis in words (e.g., Pho-TO-gra-phy vs. Pho-to-GRAPH-ic). Intonation and Rhythm: Rising and Falling tones to convey emotion and questions. Neutral Accent training: Reducing Mother Tongue Influence (MTI).

Unit 3
Listening and Comprehension: Active Listening Drills: Listening to audio clips (news, simple dialogues) and answering questions. Note-taking: Techniques to capture key points during a conversation. Vocabulary Building: Synonyms, Antonyms, Homophones, and basic idioms. Contextual guessing: Understanding words based on the surrounding sentence.

Unit 4
Basic Speaking and Conversation: Breaking the Ice: Self-Introductions (Name, Background, Hobbies). Small Talk: Greetings, asking about health/weather, and ending conversations politely. Situational Dialogues: Role-playing everyday scenarios (e.g., Asking for directions, Ordering food, Talking to a teacher). JAM (Just A Minute) Sessions: Speaking for 60 seconds on simple topics to build confidence and fluency.

Unit 5
Reading and Basic Writing: Reading Skills: Skimming and Scanning simple texts/newspaper headlines for information. Reading aloud for clarity and pause management. Basic Writing: Constructing coherent paragraphs with a Topic Sentence. Informal Letter Writing (to friends/family) to practice flow. Transformation of sentences (Active to Passive voice basics).

Innovation Lab I (IX101)

← Back to Scheme

Credits: 6 (0-0-6)
*Note: Students must select **one** of the following technical tracks for the duration of the semester.*

Track A: Web Development (Full Stack)

Task 1: The Foundations of the Web (DOM and Semantics)
Focuses on the raw "View" layer of the web. Over three sessions, students will move beyond basic syntax to understand the Document Object Model (DOM) tree. They will hand-code a responsive "Personal Portfolio" landing page using semantic HTML5 and CSS Grid/Flexbox, culminating in writing raw JavaScript to manipulate DOM elements. This establishes the baseline for how the browser renders the View before frameworks abstract it away.

Task 2: Asynchronous JavaScript and Data Fetching
Focuses on how the View requests data. Students will master the Event Loop, Promises, and async/await. The deliverable is a "Dynamic Weather Dashboard" that fetches real-time data from a public API. This introduces the concept that the View must interact with external data sources (External Controllers), parsing JSON responses to update the UI without page reloads.

Task 3: The View Layer: Component Architecture with React
Focuses strictly on the 'V' (View) in MVC. Students will decompose a complex UI into reusable, isolated pieces using React. They will build an interactive "Product Gallery," focusing strictly on JSX and Props. They will learn that the View's job is solely to present data provided to it, reinforcing the separation of concerns between presentation and logic.

Task 4: State Management (Local View Logic)
Focuses on interactivity within the View. Students will deep-dive into React Hooks (useState, useEffect). They will engineer a "Task/Kanban Board" allowing users to create and filter items. This teaches the difference between "Application State" (Data) and "UI State" (is this menu open?), a critical distinction in MVC architectures.

Task 5: Next.js and Structural Routing
Focuses on the framework that holds the MVC structure together. Students will migrate to the Next.js ecosystem. They will build a "Multi-page Blog" utilizing the App Router. They will learn how file-system-based routing acts as the traffic director for the application, mapping URLs to specific Views and Layouts.

Task 6: Rendering Strategies (SSR as Server-Generated Views)
Focuses on where the View is constructed. Students will experimentally compare Client-Side Rendering (CSR) against Server-Side Rendering (SSR). They will build a "News Aggregator" that pre-renders HTML on the server. They will learn how the server can act as the primary View generator to improve SEO and performance before the client JavaScript takes over.

Task 7: The Controller Layer: API Routes and Logic
Focuses on the 'C' (Controller). Students will learn to write backend logic within Next.js using API Routes (Serverless Functions). They will implement the backend for a "Feedback System," handling HTTP methods (GET, POST). They will learn that the Controller's job is to receive

input, validate it, process business logic, and determine the response, keeping this logic out of the React components.

Task 8: The Model Layer: Databases and ORM
Focuses on the 'M' (Model). Students will connect their application to a persistent cloud database (PostgreSQL or MongoDB) using an ORM like Prisma. They will define strict Data Schemas for an "Inventory Management System." This lab emphasizes that the Model represents the data structure and business rules (constraints) independent of how that data is displayed or accessed.

Task 9: MVC Integration (Connecting M, V, and C)
Focuses on the complete data flow. Students will refactor their Inventory System into a fully functional CRUD application. They will wire the system so that a user action (View) triggers an API call (Controller), which queries the Database (Model), and returns data to update the UI. This solidifies the "One-Way Data Flow" and separation of duties inherent in MVC.

Task 10: Authentication and Production Deployment
Focuses on securing the MVC pipeline. Students will implement secure login flows (NextAuth/Auth.js) to protect their Controllers (API routes) and Views (Private Pages). The final activity involves deploying a "Real-time Chat Interface" to a production environment (Vercel), configuring environment variables, and setting up a CI/CD pipeline to manage their full-stack application.

Track B: iOS Development (Swift)

Task 1: Swift Fundamentals: Type Safety and Data Modeling
Focuses on the core logic and unique safety features of the Swift language. Over three sessions, students will move beyond standard OOP to master the distinction between Value Types (Structs) and Reference Types (Classes). They will explore Swift's rigorous Type System, including Optionals (handling null safety), Enums with Associated Values for state modeling, and Error Handling (do-try-catch). The deliverable is a complex "CLI Logic Engine" (such as a Banking System or RPG Logic) that prevents runtime crashes through compile-time checks, ensuring the data model is robust before any UI is introduced.

Task 2: The View Layer: UIKit, AutoLayout, and Gestures
Focuses on the "Imperative" way of building interfaces and handling user interaction, essential for understanding the OS's history. Students will manipulate the View hierarchy using Interface Builder (Storyboards) and AutoLayout Constraints. Crucially, they will implement Event-Driven Programming by attaching UIGestureRecognizer (Tap, Pinch, Swipe, Pan) to views. They will build an "Interactive Profile Card" that responds to physical inputs with animations, understanding how the "View" captures user intent and notifies the code via the Target-Action pattern.

Task 3: The Controller Layer: Delegation and MVC
Focuses on the 'C' (Controller) and the specific communication patterns of iOS. Students will master the Delegation Design Pattern (blind communication) by manually implementing a UITableView, the backbone of most iOS applications. They will build a "Contact Management System" where a UIViewController acts as the mediator between the Data Model and the View. This task enforces the strict Model-View-Controller (MVC) separation found in legacy codebases, teaching students how to manage the lifecycle (viewDidLoad) and memory references.

Task 4: State Management (The Source of Truth)
Focuses on how data drives the modern UI. Students will deep-dive into the reactivity model of SwiftUI (@State, @Binding, @Environment). They will engineer an "Interactive Order Form" containing toggles, steppers, and inputs where changes in the data model instantly redraw the interface without manual intervention. This teaches the core concept that the UI is a function of its State, training students to abandon the "UpdateUI()" methods used in Task 2.

Task 5: The Paradigm Shift: Declarative UI with SwiftUI
Focuses on the modern "View" layer and the shift from Imperative to Declarative programming. Students will transition to SwiftUI, learning how to describe what the UI should look like rather than how to build it. They will recreate a complex "Responsive Dashboard" using Stacks (HStack, VStack, ZStack) and Modifiers. They will analyze how the SwiftUI View Tree handles layout, accessibility, and Dark Mode adaptation automatically, contrasting the reduction in code volume against their previous UIKit work.

Task 6: The MVVM Architecture (Separating Logic)
Focuses on the Model-View-ViewModel pattern, the industry standard for modern iOS. Students will learn to decouple Business Logic completely from the UI. They will refactor a "Trivia Quiz Engine" so that all game rules, score calculations, and state transitions reside in an @Observable (or ObservableObject) ViewModel class. The View simply observes this object. This reinforces the separation of concerns: The View handles presentation, while the ViewModel handles the intelligence and testability.

Task 7: Navigation Structure and Data Injection
Focuses on scaling the application architecture. Students will learn to structure larger, multi-

screen applications using NavigationStack and List (Dynamic Arrays). They will build a "Dynamic Asset Tracker" that passes data (custom Structs) securely from a parent list to a detail view using NavigationLink. This task emphasizes Data Injection—passing dependencies down the View hierarchy—and managing the Heap vs. Stack memory when moving between screens.

Task 8: Asynchronous Networking and Concurrency
Focuses on connecting the app to external environments. Students will master Swift's modern Concurrency model (async/await, Task, Actors) and URLSession. They will build a "Live News Feed" that fetches JSON data from a public API, parses it using the Codable protocol, and handles Loading States (Activity Indicators) and Error States. This lab emphasizes thread safety—performing heavy networking on background threads while safely updating the UI on the Main Actor.

Task 9: The Model Layer: Persistence with SwiftData
Focuses on the 'M' (Model) and data durability. Students will implement offline capability using Apple's native SwiftData framework (a modern abstraction over CoreData). They will build a "Persistent Journal App" that performs full CRUD (Create, Read, Update, Delete) operations. They will define strict Data Models (@Model) that survive app restarts, learning to query using Predicates and sort data locally, effectively turning the device into a database server.

Task 10: Hardware Integration and Deployment
Focuses on the "Mobile" advantage and production readiness. Students will interface with device sensors using CoreLocation (GPS) and MapKit. The final activity involves building a "Memory Map" (Geotagging App) where users pin notes to physical locations. The task concludes with the Deployment Pipeline: managing Info.plist permissions, creating App Icons, Archiving the build, and understanding the TestFlight/App Store submission process.

Yoga Education & Practices (UC1004)

← Back to Scheme

Credits: 2 (0-0-2)

Course Guidelines

This course is practical and seminar-based. Students are expected to practice fundamental Asanas and deliver a comprehensive presentation (PPT) on one of the theoretical themes listed below, supported by scientific evidence or case studies.

Course Content & Presentation Themes

Theme 1: Foundations and Philosophy of Yoga
Topics for Presentation: Historical evolution of Yoga (Vedic to Modern). Introduction to Patanjali's Yoga Sutras. The Eight Limbs of Yoga (Ashtanga): Yama, Niyama, Asana, Pranayama, Pratyahara, Dharana, Dhyana, and Samadhi. Misconceptions about Yoga vs. the reality of holistic wellness.

Theme 2: Physical Culture (Asanas) and Anatomy
Topics for Presentation: The science of Surya Namaskar (Sun Salutation) and its physiological effects. Classification of Asanas: Standing, Sitting, Supine, and Prone. Impact of Yoga on the Musculoskeletal system (Flexibility, Core strength, Posture correction). Yoga for preventing ergonomic hazards (e.g., "Tech Neck" and Back pain in computer professionals).

Theme 3: Pranayama and Respiratory Physiology
Topics for Presentation: The connection between Breath and the Nervous System. Techniques: Nadi Shodhana (Alternate Nostril Breathing), Kapalbhati (Cleaning Breath), and Bhramari (Humming Bee Breath). Scientific analysis of how Pranayama regulates the Autonomic Nervous System (Sympathetic vs. Parasympathetic) and reduces cortisol levels.

Theme 4: Mental Health and Stress Management
Topics for Presentation: Concept of Pratyahara (Withdrawal of senses) in a digital age. Meditation techniques (Dhyana) for focus and concentration. Yoga Nidra (Psychic Sleep) for deep relaxation. Role of Yoga in managing anxiety, depression, and insomnia among students.

Theme 5: Applied Yoga and Lifestyle
Topics for Presentation: Yoga for Lifestyle Disorders (Diabetes, Hypertension, Obesity). Corporate Yoga: Techniques for stress management in the workplace. Diet and Nutrition: The concept of Mitahara and Sattvic diet. Integration of Yoga values (Truthfulness, Non-violence) into professional ethics.

Semester II

Probability & Statistics (MA102)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Foundations of Probability: Sample spaces, Events, Axiomatic definition of Probability. Permutations and Combinations in complex counting problems. Conditional Probability, Independence of events, and Total Probability Theorem. **Bayes' Theorem** and its applications in classification and diagnostic systems (Naïve Bayes foundation). Introduction to Random Variables: Definition, Cumulative Distribution Functions (CDF), and Probability Mass/Density Functions (PMF/PDF).

Unit 2
Random Variables and Distributions: Theoretical distributions essential for data modeling. *Discrete:* Bernoulli, Binomial, Geometric, and Poisson (modeling rare events/cyber attacks). *Continuous:* Uniform, Exponential (reliability/waiting times), Gamma, Beta, and **Normal (Gaussian) Distribution** (properties and the Empirical Rule). Expectation, Variance, Skewness, Kurtosis, and Moment Generating Functions (MGF). Transformation of Random Variables.

Unit 3
Multivariate Analysis and Information Theory: Jointly distributed random variables (Discrete and Continuous). Marginal and Conditional distributions. Independent Random Variables. Covariance, Correlation Coefficient, and Rank Correlation. **Information Theory for ML:** Self-Information, Entropy (Shannon), Joint and Conditional Entropy, Mutual Information, and **Kullback-Leibler (KL) Divergence** (foundations for Cross-Entropy Loss in Neural Networks).

Unit 4
Statistical Inference and Estimation: Sampling distributions: Chi-square, t-distribution, and F-distribution. Central Limit Theorem (CLT) and Law of Large Numbers. **Parameter Estimation:** Point estimation using **Maximum Likelihood Estimation (MLE)** and **Maximum A Posteriori (MAP)** (Bayesian approach). Interval Estimation: Confidence Intervals for mean, variance, and proportions. Hypothesis Testing: Null/Alternative hypotheses, Type I/II errors, p-values, Z-test, t-test, and ANOVA.

Unit 5
Stochastic Processes for RL and Cyber: Introduction to Stochastic processes. **Markov Chains:** State space, Transition Probability Matrix (TPM), Chapman-Kolmogorov equations, Stationary distributions, and Ergodicity (Mathematical foundation of **Reinforcement Learning**). Random Walks and Poisson Processes (Network traffic modeling). Introduction to Monte Carlo Methods and Sampling techniques.

EE Fundamentals (EE101)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
DC Circuit Analysis and Power Foundations: Fundamental Concepts: Electrical Current, Voltage, Resistance, Power, and Energy. The Concept of "Ground" and Reference Voltages in Digital Systems. Ohm's Law. Circuit Elements: Resistors, Inductors, and Capacitors. Series and Parallel Combinations. Practical concepts of Pull-up and Pull-down Resistors. Analysis Methods: Kirchhoff's Laws (KCL/KVL), Mesh Analysis, and Nodal Analysis. Network Theorems: Thevenin's, Norton's, and Maximum Power Transfer Theorem.

Unit 2
AC Circuits, Transients, and Signal Basics: AC Fundamentals: Generation of Sinusoidal Voltage, Frequency, Period, Phase, Average, and RMS Values. Introduction to Non-Sinusoidal Signals (Square Waves and Pulses). Transient Analysis: Behavior of RC and RL circuits with DC excitation, Time Constants, and their relation to Digital Signal Propagation Delay. AC Circuit Analysis: Phasor Representation, Series R-L, R-C, and R-L-C Circuits. Impedance, Resonance, and Bandwidth. Signal Conditioning: Passive Low-Pass and High-Pass RC Filters.

Unit 3
Semiconductor Devices and Digital Physics: Diodes: P-N Junction Physics, V-I Characteristics. Applications: Rectifiers, Zener Regulation, and Clipping/Clamping Circuits. Bipolar Junction Transistor (BJT): Operation as a Switch (Cutoff vs. Saturation). Field-Effect Transistors (MOSFET): JFET and Enhancement-mode MOSFETs. Comparison of BJT vs. MOSFET logic. CMOS Technology: The CMOS Inverter and physical construction of Logic Gates (AND, OR, NOT).

Unit 4
Operational Amplifiers and Data Conversion: Operational Amplifiers (Op-Amps): Characteristics of Ideal vs. Practical Op-Amps. Virtual Ground Concept. Configurations: Inverting, Non-Inverting, and Unity Gain Buffer. Applications: Summing Amplifier, Differentiator, Integrator, and Comparator (Zero Crossing Detector). Data Conversion: Analog-to-Digital Converters (ADC) and Digital-to-Analog Converters (DAC). Sampling Theorem (Nyquist Rate) and Quantization.

Unit 5
Electronic Systems, Interfacing, and Communication: Sensors and Transducers: Active vs. Passive Sensors (LDR, Photodiode, Thermistor, Ultrasonic). Signal Conditioning requirements. Actuators and Drivers: Interfacing Microcontrollers to High-Power Loads (Relays, DC Motors). Opto-isolators for Galvanic Isolation. H-Bridge Circuits for Motor Direction Control. PWM (Pulse Width Modulation) basics.

EE Fundamentals Lab (EX101)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Verification of Circuit Laws (KCL and KVL) Focuses on the fundamental conservation laws of electrical engineering. Students will construct a resistive network to verify Kirchhoff's Current Law (node analysis) and Voltage Law (loop analysis) by measuring actual branch currents and voltage drops.

Lab 2: Network Theorems (Thevenin, Norton, and Max Power) Focuses on circuit simplification techniques. Students will experimentally determine the Thevenin equivalent voltage and resistance of a complex circuit and verify the Maximum Power Transfer theorem, which is crucial for impedance matching in signal transmission.

Lab 3: Transient Response of RC Circuits (Time Constants) Focuses on the timing behavior of circuits. Students will observe the charging and discharging curves of a Capacitor through a Resistor on an oscilloscope, calculating the Time Constant to understand propagation delays in digital logic.

Lab 4: Passive Filters (Low-Pass and High-Pass) Focuses on signal conditioning. Students will design and test RC filter circuits to understand how to attenuate high-frequency noise (Low-Pass) or block DC offsets (High-Pass), relating this to signal integrity in communication lines.

Lab 5: Diode Characteristics and Logic Protection Focuses on the P-N junction. Students will plot the V-I characteristics of a diode and implement a "Clipping Circuit," demonstrating how diodes are used to protect sensitive CPU pins from over-voltage spikes.

Lab 6: The Transistor as a Digital Switch Focuses on the core component of the CPU. Students will configure a BJT or MOSFET to switch a load (like an LED or Relay) ON and OFF, identifying the Cutoff (Logic 0) and Saturation (Logic 1) regions of operation.

Lab 7: Linear Op-Amp Circuits (Math Operations) Focuses on analog signal processing. Students will implement Inverting and Non-Inverting amplifiers and a Summing Amplifier using the 741 IC.

Lab 8: Comparators and Digital-to-Analog Conversion Focuses on the boundary between analog and digital. Students will build a Comparator circuit (1-bit ADC) and a simple 4-bit R-2R Ladder Network to generate analog voltages from digital inputs (DAC).

Lab 9: Sensor Interfacing (LDR and Thermistors) Focuses on reading the physical world. Students will interface resistive sensors (Light Dependent Resistor or Thermistor) using a Voltage Divider configuration to convert physical changes into measurable voltage changes.

Lab 10: Pulse Width Modulation (PWM) and Motor Control Focuses on actuation. Students will use a transistor driver to control the speed of a small DC motor or the brightness of an LED using a PWM signal (simulated via function generator), introducing the concept of duty cycle control.

Data Structures (CS102)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
Dynamic Data Structures (Linked Lists): Limitations of arrays and vectors, Node structure and dynamic memory allocation, Singly Linked List (Creation, Insertion, Deletion, Reversal), Doubly Linked List (Memory overhead, Bidirectional traversal), Circular Linked Lists, Applications of Linked Lists (Polynomial arithmetic, Sparse Matrix representation).

Unit 2
Restricted Linear Structures (Stacks and Queues): The Stack ADT, Array vs. Linked List implementation of Stacks, System Stack and Recursion, Queue ADT, Linear Queue limitations, Circular Queue implementation, Deque (Double Ended Queue), Priority Queue fundamentals, Applications (Expression conversion/evaluation, Job scheduling).

Unit 3
Hierarchical Structures (Binary Trees): Tree terminology, Binary Tree representations (Sequential Array vs. Linked Nodes), Tree Traversals (In-order, Pre-order, Post-order, Level-order), Construction of trees from traversals, Threaded Binary Trees, Binary Search Trees (BST), Operations on BST (Search, Insert, Delete), The Skewed Tree problem.

Unit 4
Advanced Trees and Heaps: Self-Balancing properties, AVL Trees (Balance Factors, Left/Right Rotations), Red-Black Trees (Properties, Color invariants, Insertion logic), Heaps (Max-Heap and Min-Heap structures), Array representation of Heaps, Heapify operations, Heap Sort logic, Introduction to B-Trees for external storage.

Unit 5
Graphs: Graph terminology (Vertex, Edge, Weight, Degree), Graph Representations (Adjacency Matrix, Adjacency List, Adjacency Multi-list), Storage complexity of representations, Graph Traversals (Breadth-First Search, Depth-First Search), Cycle detection in graphs, Connectivity (Connected components).

Data Structures Lab (CX102)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Singly Linked Lists (SLL) Focuses on the fundamentals of dynamic memory allocation, node creation, and pointer manipulation to implement the standard Singly Linked List operations (Insert, Delete, Traversal).

Lab 2: Doubly and Circular Linked Lists Focuses on managing complex pointer structures, specifically handling bidirectional traversal in Doubly Linked Lists and the wrap-around logic of Circular Linked Lists.

Lab 3: Stack Implementation and Applications Focuses on the Last-In-First-Out (LIFO) principle, exploring Stack implementation using both Arrays and Linked Lists, and applying it to linear data processing.

Lab 4: Queues and Double-Ended Queues Focuses on the First-In-First-Out (FIFO) principle, specifically implementing Linear Queues, Circular Queues (to handle array reuse), and Deques.

Lab 5: Binary Tree Fundamentals Focuses on non-linear data organization, implementing the basic Binary Tree structure and mastering recursive traversal algorithms (Pre-order, In-order, Post-order).

Lab 6: Binary Search Trees (BST) Focuses on the specific properties of Search Trees, implementing logic to maintain sorted order ($Left < Root < Right$) during insertion and performing efficient lookups.

Lab 7: Heaps and Priority Queues Focuses on array-based tree representations, specifically implementing Min-Heaps and Max-Heaps and the "Heapify" logic required for Priority Queues.

Lab 8: Balanced Tree Logic (AVL) Focuses on the concept of self-balancing data structures, specifically calculating balance factors and implementing tree rotations to maintain optimal height.

Lab 9: Graph Representations Focuses on the storage of network data, implementing Graphs using Adjacency Matrices for dense graphs and Adjacency Lists for sparse graphs.

Lab 10: Graph Traversals Focuses on algorithms to navigate graph structures, specifically implementing Breadth-First Search (BFS) using queues and Depth-First Search (DFS) using stacks/recursion.

Professional Communication (HS102)

← Back to Scheme

Credits: 3 (0-3-0)

Course Content

Unit 1
Professional Correspondence and Workplace Communication: Foundations of Tone: Transitioning from academic to professional language. Formal vs. Semi-formal registers. The "Bottom Line Up Front" (BLUF) principle. Email Etiquette: Constructing professional subject lines, salutations, and sign-offs. Managing threads and attachments. Netiquette and response time expectations. Professional Requests and Responses: Drafting apologies, requests for leave/extensions, and follow-up emails. Documentation: Writing effective Minutes of Meeting (MoM) and internal office memos.

Unit 2
Technical Writing and Document Preparation (LaTeX): Introduction to LaTeX: The necessity of standardized formatting in industry. Setting up an environment (Overleaf). Basic syntax, file structure, and compilation. Document Structure: Creating Title Pages, Abstracts, Sections, and Subsections. Automated Table of Contents. Technical Elements: Mathematical typesetting, inserting Figures/Images, and creating Tables. Referencing and Reports: Managing Bibliographies/Citations (BibTeX). Writing a comprehensive Technical Project Proposal or Laboratory Report.

Unit 3
Interpersonal Dynamics and Group Discussions: Listening Skills: Active listening vs. passive hearing. Non-verbal cues in group settings. Group Discussion (GD) Mechanisms: Initiation techniques, body of discussion, and summarization. Difference between Debate (argument) and GD (consensus). Collaboration Skills: Turn-taking, agreeing/disagreeing politely, and intervening in conflict. Role-Playing: Simulation of corporate boardroom scenarios and problem-solving meetings.

Unit 4
Effective Presentation Skills and Public Speaking: Content Organization: Structuring a presentation (Introduction, Hook, Body, Conclusion). The 10-20-30 Rule. Storytelling in technical contexts. Visual Aids: Designing effective slides (PowerPoint/Beamer). Principles of minimalism and data visualization. Delivery Techniques: Voice modulation, pitch, pace, and pause. Eye contact and platform presence. Audience Management: Handling Q&A sessions and managing technical glitches gracefully.

Unit 5
Career Readiness and Recruitment Skills: Personal Branding: Understanding the purpose of a Resume/CV. Chronological vs. Functional formats. Resume Building: Creating ATS (Applicant Tracking System) friendly resumes. Writing tailored Cover Letters. Interview Fundamentals: Types of interviews (HR vs. Technical). Professional grooming and body language. Answering Techniques: The STAR Method (Situation, Task, Action, Result) for behavioral questions. Mock interview practice.

Environment Studies (UC2004)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Introduction to Environmental Studies & Ecosystems: Foundations: Multidisciplinary nature of environmental studies; Scope and importance; Concept of sustainability and sustainable development. Ecosystem Dynamics: Structure and function of ecosystem; Energy flow in an ecosystem: food chains, food webs and ecological succession. Case Studies: Characteristics of Forest ecosystem, Grassland ecosystem, Desert ecosystem, and Aquatic ecosystems (ponds, streams, lakes, rivers, oceans, estuaries).

Unit 2
Natural Resources: Renewable and Non-renewable Resources: Land Resources: Land use change; Land degradation, soil erosion and desertification. Deforestation: Causes and impacts due to mining and dam building on environment, forests, biodiversity and tribal populations. Water Resources: Use and over-exploitation of surface and ground water, floods, droughts, conflicts over water (international & inter-state). Energy Resources: Renewable and non-renewable energy sources, use of alternate energy sources, growing energy needs, case studies.

Unit 3
Biodiversity and Conservation: Levels of Diversity: Genetic, species and ecosystem diversity; Biogeographic zones of India; Biodiversity patterns and global biodiversity hot spots. Biodiversity in India: India as a mega-biodiversity nation; Endangered and endemic species of India. Threats and Conservation: Habitat loss, poaching of wildlife, man-wildlife conflicts, biological invasions; In-situ and Ex-situ conservation of biodiversity. Ecosystem Services: Ecological, economic, social, ethical, aesthetic and Informational value.

Unit 4
Environmental Pollution, Policies & Practices: Pollution and Waste: Types, causes, effects and controls of Air, water, soil and noise pollution; Nuclear hazards and human health risks; Solid waste management (urban and industrial). Global Issues: Climate change, global warming, ozone layer depletion, acid rain and impacts on human communities and agriculture. Environmental Laws: Environment Protection Act; Air (Prevention & Control of Pollution) Act; Water Act; Wildlife Protection Act; Forest Conservation Act. Agreements and Rights: International agreements (Montreal and Kyoto protocols, CBD); Nature reserves, tribal populations and rights, and human wildlife conflicts in Indian context.

Unit 5
Human Communities and the Environment: Population and Society: Human population growth: Impacts on environment, human health and welfare; Resettlement and rehabilitation of project affected persons; case studies. Disaster Management: Floods, earthquake, cyclones and landslides. Environmental Movements: Chipko, Silent valley, Bishnois of Rajasthan. Ethics and Communication: Role of Indian and other religions/cultures in environmental conservation; Environmental communication and public awareness (e.g., CNG vehicles in Delhi).

Semester III

Discrete Mathematical Structures (MA201)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Mathematical Logic and Proofs: Propositional Logic: Statements, Connectives, Truth Tables, Tautologies, and Contradictions. Normal Forms (CNF, DNF). Predicate Logic: Quantifiers ($\forall, \exists$), Nested Quantifiers, and Inference Theory. Proof Techniques: Direct, Indirect (Contrapositive), Contradiction, and **Mathematical Induction** (Strong and Weak) with applications to algorithm correctness.

Unit 2
Sets, Relations, and Functions: Set Theory: Operations, Power Sets, Principle of Inclusion-Exclusion. **Relations:** Properties (Reflexive, Symmetric, Transitive), Equivalence Relations and Partitions (Database foundations). Partial Order Relations: Posets, Hasse Diagrams, Lattices, and Chains. Functions: Injective, Surjective, Bijective, Inverse Functions, and the Pigeonhole Principle with applications to hashing.

Unit 3
Combinatorics and Recurrence Relations: Counting Principles: Permutations, Combinations, and Derangements. Binomial and Multinomial Theorems. **Recurrence Relations:** Generating Functions, Linear Recurrence Relations with Constant Coefficients (Homogeneous and Non-homogeneous). Solving recurrences for **Algorithm Analysis** (e.g., Fibonacci, Divide and Conquer complexity).

Unit 4
Graph Theory: Fundamentals: Directed and Undirected graphs, Weighted graphs, Paths, Cycles, and Connectivity. Eulerian and Hamiltonian Paths (Network routing problems). Graph Representation (Adjacency Matrix/List). Planar Graphs and Graph Coloring (Map coloring, Register allocation). Isomorphism and Homomorphism. Shortest Path algorithms concepts (Dijkstra/Floyd-Warshall context).

Unit 5
Trees and Algebraic Structures: **Trees:** Rooted Trees, Binary Search Trees, Spanning Trees (Prim's and Kruskal's logic), and Tree Traversals. **Algebraic Structures:** Groups, Subgroups, Cyclic Groups, Lagrange's Theorem. Rings and Fields (Finite Fields). Introduction to Number Theory: Modular Arithmetic, Euclidean Algorithm, Fermat's Little Theorem, and Euler's Totient Function (Mathematical foundation for **RSA Cryptography**).

Digital Logic Design (CS201)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Digital Systems and Boolean Algebra: Introduction to Digital vs. Analog Systems. Number Systems (Binary, Octal, Hexadecimal) and Base Conversions. Signed Number Representations (Sign-Magnitude, 1's and 2's Complement). Binary Codes (BCD, Gray Code). Axiomatic Definition of Boolean Algebra, Basic Theorems and Properties. Logic Gates (AND, OR, NOT, NAND, NOR, XOR, XNOR). Canonical and Standard Forms (SOP, POS).

Unit 2
Logic Function Minimization and Combinational Circuits: Minimization of Boolean functions using Karnaugh Maps (K-Maps) up to 4 variables. "Don't Care" conditions. Introduction to Combinational Logic Circuits. Design of Arithmetic Circuits: Half Adder, Full Adder, Ripple Carry Adder, Half Subtractor, Full Subtractor. Magnitude Comparators. Code Converters. Introduction to Hardware Description Language (HDL).

Unit 3
MSI Combinational Components: Design and analysis of Medium Scale Integration (MSI) components. Decoders and Encoders (including Priority Encoders). Multiplexers (MUX) as universal function generators and for data routing. Demultiplexers (DEMUX). Introduction to Programmable Logic Devices (PLDs): Read-Only Memory (ROM), Programmable Logic Array (PLA), and Programmable Array Logic (PAL).

Unit 4
Sequential Logic Fundamentals: Introduction to Sequential Circuits: Latches vs. Flip-Flops. The SR Latch (NAND/NOR implementations). Clocked SR, D, JK, and T Flip-Flops. Master-Slave JK Flip-Flop. Characteristic Tables and Excitation Tables. Timing Parameters: Propagation Delay, Setup Time, and Hold Time. Introduction to State Diagrams and State Tables.

Unit 5
Registers, Counters, and State Machines: Registers: Buffer, Serial-In/Serial-Out (SISO), Parallel-In/Parallel-Out (PIPO), and Shift Registers. Counters: Asynchronous (Ripple) vs. Synchronous Counters. Design of Synchronous Counters using Flip-Flop Excitation Tables. Introduction to Finite State Machines (FSMs): Moore and Mealy models. FSM Design Procedure.

Digital Logic Design (CS201)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Logic Gate Verification Focuses on familiarization with the digital logic simulation environment (e.g., Logisim, Multisim). Students will construct and simulate all basic logic gates (AND, OR, NOT, NAND, NOR, XOR, XNOR) to verify their respective truth tables.

Lab 2: Implementation of Boolean Functions Focuses on translating Boolean algebra into digital circuits. Students will implement logic functions directly from their standard Sum-of-Products (SOP) and Product-of-Sums (POS) forms to understand their structural equivalence.

Lab 3: Circuit Minimization with K-Maps Focuses on logic optimization. Students will use Karnaugh Maps (K-Maps) to simplify a 4-variable Boolean function and then implement both the original and the simplified circuits to prove that they are functionally identical but with reduced gate count.

Lab 4: Design of Arithmetic Circuits Focuses on core computational blocks. Students will design, implement, and test a Half Adder and a Full Adder. These will then be cascaded to construct a 4-bit Ripple Carry Adder to perform binary addition.

Lab 5: Design of Data Selectors and Routers Focuses on MSI combinational logic. Students will design and test a 4-to-1 Multiplexer (MUX) to demonstrate data selection and a 1-to-4 Demultiplexer (DEMUX) to demonstrate data distribution.

Lab 6: Design of Encoders and Decoders Focuses on data representation and addressing. Students will implement a Decimal-to-BCD Priority Encoder and a 3-to-8 Line Decoder, which are fundamental components for memory addressing and I/O control.

Lab 7: Latches and Flip-Flops Focuses on the fundamentals of sequential memory. Students will build a basic SR Latch using cross-coupled NAND gates and then advance to constructing a clocked D-type Flip-Flop, observing the difference between level-triggered and edge-triggered behavior.

Lab 8: Flip-Flop Conversions Focuses on the versatility of sequential elements. Students will implement a JK Flip-Flop and a T Flip-Flop by augmenting a D Flip-Flop with external combinational logic, based on their characteristic equations.

Lab 9: Registers and Counters Focuses on sequential MSI components. Students will design and simulate a 4-bit Parallel-In, Serial-Out (PISO) Shift Register and a 3-bit synchronous binary up-counter using JK or T flip-flops.

Lab 10: Capstone: Finite State Machine (FSM) Design Focuses on the integration

of all concepts to create a control unit. Students will design, implement, and test a complete Moore or Mealy type Finite State Machine to detect a specific binary sequence (e.g., "1011") from a serial input.

Design & Analysis of Algorithms (CS202)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Foundations of Algorithm Analysis: The Algorithm as a concept, Measuring Performance. Time and Space Complexity. Asymptotic Notations: Big-O (Upper Bound), Big-Omega (Lower Bound), and Big-Theta (Tight Bound). Best, Average, and Worst-Case Analysis. Analysis of Iterative Algorithms. Analysis of Recursive Algorithms using Recurrence Relations: Substitution Method, Recursion Tree Method, and the Master Theorem for solving recurrences.

Unit 2
Divide and Conquer & Greedy Algorithms: The Divide and Conquer strategy: Recurrence relation and analysis. Classic Algorithms: Binary Search, Merge Sort, and Quicksort (including randomized versions). The Greedy Method: General approach, proof of correctness (Greedy Choice Property, Optimal Substructure). Applications: Fractional Knapsack Problem, Job Sequencing with Deadlines, Huffman Coding, and Minimum Spanning Tree algorithms (Prim's and Kruskal's).

Unit 3
Dynamic Programming: The Dynamic Programming paradigm, comparison with Divide and Conquer. Overlapping Subproblems and Optimal Substructure properties. Design approaches: Memoization (Top-Down) vs. Tabulation (Bottom-Up). Applications: 0/1 Knapsack Problem, Longest Common Subsequence (LCS), Matrix Chain Multiplication, and the All-Pairs Shortest Path problem (Floyd-Warshall Algorithm).

Unit 4
Backtracking and Advanced Graph Algorithms: Backtracking: The general method, constructing the state-space tree. Applications: N-Queens Problem, Sum of Subsets, and Graph Coloring. Branch and Bound as an optimization of backtracking. Single-Source Shortest Paths: Dijkstra's Algorithm (using Priority Queues) and the Bellman-Ford Algorithm for handling negative edge weights.

Unit 5
Complexity Theory and String Matching: Introduction to Computational Complexity. The complexity classes P (Polynomial time) and NP (Nondeterministic Polynomial time). Understanding NP-Completeness and the concept of reduction. Introduction to NP-Complete problems (e.g., TSP, SAT). String Matching Algorithms: Naive approach, Rabin-Karp Algorithm, and the Knuth-Morris-Pratt (KMP) Algorithm.

Design & Analysis of Algorithms (CS202)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Empirical Analysis of Sorting Algorithms Focuses on experimentally verifying theoretical time complexities. Students will implement and compare the runtime performance of an $O(n^2)$ sort (e.g., Insertion Sort) with an $O(n \cdot logn)$ sort (e.g., Merge Sort) across various input sizes.

Lab 2: Divide and Conquer (Quicksort) Focuses on the practical implementation of the Divide and Conquer paradigm. Students will implement the Quicksort algorithm, paying close attention to the partitioning logic and the choice of the pivot element.

Lab 3: Greedy Algorithms (Knapsack and Job Sequencing) Focuses on making locally optimal choices to achieve a global optimum. Students will implement the greedy solution for the Fractional Knapsack problem and the Job Sequencing with Deadlines problem.

Lab 4: Minimum Spanning Trees Focuses on applying greedy algorithms to graph structures. Students will implement either Prim's algorithm (using a priority queue) or Kruskal's algorithm (using a Disjoint Set Union data structure) to find the MST of a given graph.

Lab 5: Dynamic Programming (Memoization) Focuses on the top-down approach to solving problems with overlapping subproblems. Students will implement a recursive solution to the 0/1 Knapsack problem, using a memoization table to store and reuse results.

Lab 6: Dynamic Programming (Tabulation) Focuses on the bottom-up approach to dynamic programming. Students will implement an iterative solution for the Longest Common Subsequence (LCS) problem using a 2D table to build up the solution from smaller subproblems.

Lab 7: Backtracking (N-Queens) Focuses on solving constraint satisfaction problems by exploring a state-space tree. Students will implement a backtracking algorithm to find all possible solutions for placing N queens on an N×N chessboard without attacking each other.

Lab 8: Shortest Path Algorithms (Dijkstra's) Focuses on finding the most efficient path in a network. Students will implement Dijkstra's algorithm to find the single-source shortest paths in a weighted graph with non-negative edge weights, using a priority queue for efficiency.

Lab 9: String Matching (KMP Algorithm) Focuses on efficient pattern searching within text. Students will implement the Knuth-Morris-Pratt (KMP) algorithm, which involves preprocessing the pattern to create a Longest Prefix Suffix (LPS) array to avoid redundant comparisons.

Lab 10: Shortest Path with Negative Weights (Bellman-Ford) Focuses on handling

more complex graph scenarios. Students will implement the Bellman-Ford algorithm, which can compute single-source shortest paths in weighted graphs that may contain negative edge weights, and also detect negative cycles.

Theory of Computation (CS203)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Finite Automata and Regular Languages: Introduction to the theory of computation, Alphabets, Strings, and Languages. Deterministic Finite Automata (DFA) and Nondeterministic Finite Automata (NFA). Equivalence of NFA and DFA. Regular Expressions and their relationship with Finite Automata. Moore and Mealy Machines. Closure properties of regular languages.

Unit 2
Grammars and Properties of Regular Languages: Introduction to Formal Grammars and the Chomsky Hierarchy. Regular Grammars (Right-Linear and Left-Linear). The Pumping Lemma for Regular Languages, a tool to prove that a language is not regular. Minimization of Finite State Machines.

Unit 3
Context-Free Languages and Pushdown Automata: Context-Free Grammars (CFG): Derivations, Parse Trees, and Ambiguity in grammars. Pushdown Automata (PDA) as the machine model for CFGs. Deterministic vs. Nondeterministic PDAs. Simplification of CFGs (removing useless symbols, null productions, and unit productions). Normal Forms: Chomsky Normal Form (CNF) and Greibach Normal Form (GNF).

Unit 4
Turing Machines: The Turing Machine (TM) as the ultimate model of computation. Formal definition, instantaneous description, and transition diagrams. Design of Turing Machines. Variants of Turing Machines (Multi-tape, Nondeterministic) and their equivalence to the standard model. The Church-Turing Thesis.

Unit 5
Undecidability and Complexity: Recursive and Recursively Enumerable Languages. The Universal Turing Machine. The concept of undecidability. The Halting Problem as a key undecidable problem. Introduction to computational complexity, the complexity classes P and NP, and the notion of NP-Completeness.

Innovation Lab II (IX201)

← Back to Scheme

Credits: 6 (0-0-6)
Note: Students continue in their selected track from Innovation Lab I. The focus shifts from building basic functional applications to engineering robust, performant, and industry-standard architectures. Technology is treated purely as a vehicle to understand underlying computer science and system design principles.

Track A: Web Development (Full Stack Architecture)

Task 1: Global State & Distributed UI Management
Focuses on managing complex data structures across a decoupled interface. Students move beyond local component state to handle global "Application State." They will implement a robust state manager (like Redux Toolkit or Zustand) to build a "Complex E-commerce Cart." They will learn the concepts of immutability, pure functions (reducers), and the pub/sub pattern, understanding how state changes propagate through a large component tree efficiently.

Task 2: Middleware & The Request Pipeline
Focuses on intercepting and processing HTTP requests before they reach the Controller. Students will explore the middleware pattern. They will build a "Role-Based Access Control (RBAC) System" (Admin vs. User vs. Guest) for an organization portal. They will learn how request parsing, header inspection, and token verification operate in the middle-tier to enforce security and logic globally without polluting individual endpoints.

Task 3: Caching Strategies & System Performance
Focuses on reducing computational overhead and database load. Students will understand the trade-offs of storing data in memory versus persistent storage. Using Redis or Next.js advanced caching, they will engineer a "High-Traffic Leaderboard." They will learn concepts like Cache Invalidation, Time-to-Live (TTL), and Write-Through caching, observing the dramatic latency reduction in their API response times.

Task 4: Relational Data Modeling & Query Optimization
Focuses on structuring complex data entities. Students will dive deep into PostgreSQL, moving beyond basic ORM usage. They will build the backend for a "Social Media Feed" requiring multi-table Joins (Users, Posts, Comments, Likes). They will learn about database normalization, foreign key constraints, indexing for performance, and the N+1 query problem, optimizing raw SQL execution plans.

Task 5: API Design: Pagination & Cursor Strategies
Focuses on efficient data transfer and RESTful best practices. Students will learn how to handle massive datasets being requested by the View. They will build an "Infinite Scroll Data Grid" for a large inventory system. They will contrast Offset-based Pagination with Cursor-based Pagination, understanding the performance implications of deep database offsets and how to design APIs that scale linearly.

Task 6: Background Processing & Task Queues
Focuses on decoupling heavy computations from the HTTP request-response cycle. Students will learn that not all tasks can be processed synchronously. Using task queues (like BullMQ or

Vercel Inngest), they will build an "Automated Report Generator" that compiles large PDFs and emails them. They will understand the concept of Worker processes, job retries, and asynchronous eventual consistency.

Task 7: Web Security Fundamentals (OWASP)
Focuses on offensive and defensive security practices. Students will intentionally exploit vulnerabilities in a provided sandbox and then patch them. They will secure a "Financial Vault App," implementing mitigations against Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), and SQL Injection. They will learn about Content Security Policies (CSP), HTTP-only cookies, and data sanitization boundaries.

Task 8: Automated Testing (TDD & E2E)
Focuses on software reliability and the CI/CD mindset. Students will adopt Test-Driven Development (TDD) using Jest and Cypress/Playwright. They will write tests *before* code to implement a strict "Payment Processing Logic Layer." They will learn the distinct roles of Unit, Integration, and End-to-End (E2E) testing, understanding how code coverage metrics dictate production readiness.

Task 9: Containerization & Ephemeral Environments
Focuses on environment parity and the "works on my machine" problem. Students will learn operating system virtualization concepts using Docker. They will containerize a multi-tier application (Node.js API + PostgreSQL + Redis) using Docker Compose. They will understand image layering, container networking, and volumes, ensuring consistent execution across development and production servers.

Task 10: Infrastructure as Code (IaC) & CI/CD Pipelines
Focuses on automation and modern DevOps practices. Students will use GitHub Actions to automate their workflow. They will configure a pipeline that lints, tests, builds, and deploys a "Production Release Candidate." They will understand the concept of continuous integration/continuous delivery, immutable deployments, and semantic versioning, wrapping up the semester with an automated, zero-downtime deployment.

Track B: iOS Development (Advanced Swift & System Design)

Task 1: Advanced Composition & Custom Layouts
Focuses on the mathematics and geometry of the View layer. Students will move beyond standard stacks to implement custom rendering logic. They will utilize the SwiftUI Layout Protocol and GeometryReader to construct a "Complex Data Visualization Dashboard" (e.g., dynamic bar charts and radial graphs). They will learn how view trees negotiate size constraints and coordinate spaces during the layout pass.

Task 2: Reactive Programming & Data Streams
Focuses on handling continuous flows of data over time. Students will learn the reactive programming paradigm using Combine and Swift AsyncSequence. They will build a "Live Stock Market Ticker." They will master operators (map, filter, debounce, combineLatest) to process asynchronous events, transforming chaotic data streams into clean, thread-safe updates for the UI layer.

Task 3: Local File System & Resource Caching
Focuses on data locality and network optimization. Students will interact directly with the iOS Sandbox architecture. They will build an "Offline-First Media Player" that downloads, caches, and manages large audio/image files using FileManager and NSCache. They will learn memory eviction policies, cache hits/misses, and the lifecycle constraints of temporary versus document directories.

Task 4: Relational Persistence & Migrations
Focuses on complex data evolution inside a mobile client. Students will deepen their SwiftData/CoreData knowledge to handle relationships (One-to-Many, Many-to-Many). They will build a "Relational Budget Tracker." Crucially, they will execute Schema Migrations, learning how to alter database structures without corrupting or losing existing user data across app updates.

Task 5: Background Execution & Operating System Limits
Focuses on app lifecycles when not in the foreground. Students will interface with the OS scheduler to perform tasks efficiently without draining the battery. They will build a "Silent News Sync" using BackgroundTasks and APNs (Silent Push Notifications). They will learn the strict resource limits imposed by mobile operating systems and how to yield execution context gracefully.

Task 6: Deep Linking & App Navigation State
Focuses on routing and external application entry points. Students will learn how an app interacts with the broader OS ecosystem via Universal Links and URL Schemes. They will build an "E-commerce App" that can be opened from a web browser directly to a specific product screen. They will learn how to parse incoming URLs and pragmatically rebuild the application's navigation state from a cold start.

Task 7: Cryptography & Secure Storage
Focuses on protecting sensitive user data. Students will bypass standard UserDefaults to master the iOS Security Framework. They will build a "Password Vault" utilizing the Keychain and Biometric Authentication (FaceID/TouchID) via LocalAuthentication. They will learn concepts of encryption at rest, secure enclaves, and symmetric vs. asymmetric keys.

Task 8: Automated Testing (Unit & UI)
Focuses on ensuring correctness and preventing regressions. Students will integrate XCTest into

their workflow. They will write unit tests for a complex "Math/Tax Calculation Engine" and UI tests that use accessibility identifiers to tap through an application. They will learn how to mock network dependencies, inject test data, and automate the validation of critical user journeys.

Task 9: Memory Management & Instruments
Focuses on performance profiling and deep system diagnostics. Students will dive into Automatic Reference Counting (ARC) mechanics (strong, weak, unowned references). They will be given a purposely flawed app and will use Xcode Instruments (Leaks, Time Profiler) to identify and fix Retain Cycles and UI frame drops. They will learn how memory leaks occur and how to engineer deterministic deallocation.

Task 10: Fastlane & Automated App Distribution
Focuses on mobile DevOps and the release pipeline. Students will automate the tedious process of app signing and deployment using Fastlane and GitHub Actions. They will configure a pipeline that increments build numbers, runs test suites, manages provisioning profiles, and pushes a build to TestFlight. They will learn the concept of CI/CD specifically tailored for the Apple ecosystem's strict code-signing requirements.

Major Specialisation I

Object-Oriented Analysis & Design (SE201)

← Back to Scheme

Credits: 3 (2-0-2)

Course Content

Unit 1
Modeling with UML: Introduction to the Software Development Life Cycle (SDLC) and the Unified Process. The role of Analysis vs. Design. Unified Modeling Language (UML) 2.0. Functional Modeling: Use Case Diagrams and Scenarios. Structural Modeling: Class Diagrams, Object Diagrams, and Component Diagrams. Behavioral Modeling: Sequence Diagrams, Activity Diagrams, and State Machine Diagrams.

Unit 2
Design Principles and GRASP: Fundamental design concepts: Abstraction, Encapsulation, Modularity, Hierarchy. Metrics for Design Quality: Coupling and Cohesion. General Responsibility Assignment Software Patterns (GRASP): Creator, Information Expert, Low Coupling, High Cohesion, and Controller patterns.

Unit 3
Architectural Principles and SOLID: The SOLID Principles of Object-Oriented Design: Single Responsibility Principle (SRP), Open/Closed Principle (OCP), Liskov Substitution Principle (LSP), Interface Segregation Principle (ISP), and Dependency Inversion Principle (DIP). Law of Demeter. Dependency Injection and Inversion of Control (IoC).

Unit 4
GoF Design Patterns (Creational and Structural): Introduction to Design Patterns. Creational Patterns: Singleton (Thread-safe implementation), Factory Method, Abstract Factory, and Builder. Structural Patterns: Adapter (Class vs. Object adapters), Composite (Tree structures), Decorator (Dynamic responsibilities), Facade, and Proxy.

Unit 5
GoF Design Patterns (Behavioral) and Refactoring: Behavioral Patterns: Observer (Publish-Subscribe), Strategy (Algorithm encapsulation), Command, Iterator, and State Pattern. Introduction to Code Smells and Refactoring techniques. Anti-patterns. Mapping Design to Code.

Laboratory Content

Lab 1: Requirement Analysis and Use Cases Focuses on identifying actors and use cases from a problem statement (e.g., Library Management System) and creating a comprehensive Use Case Diagram.

Lab 2: Static Structure Modeling Focuses on domain modeling. Students will create detailed Class Diagrams identifying attributes, operations, and relationships (Association, Aggregation, Composition, Generalization).

Lab 3: Interaction and Behavior Modeling Focuses on dynamic system behavior. Students will design Sequence Diagrams to model specific scenarios (e.g., "Withdraw Cash") and Activity Diagrams for business workflows.

Lab 4: Implementing SOLID Principles Focuses on code quality. Students will be given a "bad" codebase violating SOLID principles and must refactor it to adhere to SRP, OCP, and LSP.

Lab 5: Creational Patterns (Factory and Singleton) Focuses on object creation mechanisms. Students will implement a Database Connection pool using Singleton and a UI element generator using the Factory method.

Lab 6: Structural Patterns (Adapter and Decorator) Focuses on class composition. Students will implement the Adapter pattern to integrate a legacy third-party library and the Decorator pattern to add features to a text editor (e.g., scrolling, borders).

Lab 7: Behavioral Patterns (Strategy and Observer) Focuses on object communication. Students will implement a Payment Processing system using the Strategy pattern and a Stock Market ticker using the Observer pattern.

Lab 8: The Composite Pattern Focuses on hierarchical structures. Students will design a File System structure (Folders and Files) where individual objects and compositions are treated uniformly.

Lab 9: State Pattern Implementation Focuses on state-dependent behavior. Students will implement the logic for a Vending Machine or an Order Processing System where behavior changes based on internal state.

Lab 10: Capstone: Full System Design Focuses on the end-to-end process. Students will select a project (e.g., E-Commerce Backend), produce full UML documentation, and implement the core modules using at least 3 distinct design patterns.

Fundamentals of AI & Game Theory (ML201)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Introduction to Artificial Intelligence: History and foundations of AI, The Turing Test and Chinese Room Argument. Intelligent Agents: Agents and Environments, The Concept of Rationality, The Nature of Environments (Fully/Partially observable, Deterministic/Stochastic), Structure of Agents (Simple Reflex, Model-based, Goal-based, Utility-based).

Unit 2
Problem Solving by Search: Problem-Solving Agents, Formulating Problems, Example Problems (8-Puzzle, TSP). Uninformed Search Strategies: Breadth-First Search (BFS), Depth-First Search (DFS), Depth-Limited Search, Iterative Deepening DFS. Informed (Heuristic) Search Strategies: Greedy Best-First Search, A* Search (Optimality and Completeness), Heuristic Functions (Admissibility and Consistency). Local Search Algorithms: Hill-climbing, Simulated Annealing.

Unit 3
Game Theory and Adversarial Search: Games as Search Problems. Zero-Sum Games. Perfect Information Games: The Minimax Algorithm, Optimal decisions in games. Alpha-Beta Pruning (optimization). Imperfect Real-time decisions. Introduction to General Game Playing. Non-Zero-Sum Games: Prisoner's Dilemma, Nash Equilibrium, Cooperative vs. Non-Cooperative Games.

Unit 4
Knowledge Representation and Reasoning: Logical Agents. Propositional Logic: Syntax and Semantics, Entailment, Inference rules. First-Order Logic (FOL): Syntax and Semantics, Quantifiers (Universal, Existential). Inference in FOL: Forward Chaining, Backward Chaining, and Resolution. Knowledge Engineering and Ontologies.

Unit 5
Uncertainty and Utility Theory: Quantifying Uncertainty: Probability axioms, Random Variables, Independence, Bayes' Rule and its application. Making Simple Decisions: Utility Theory, Utility Functions, Decision Networks, The Value of Information. Introduction to Markov Decision Processes (MDP) basics (States, Actions, Rewards).

Advanced Statistics for Data Science (DS201)

← Back to Scheme

Credits: 3 (2-1-0)

Course Content

Unit 1
Advanced Hypothesis Testing and Experimental Design: Review of Hypothesis Testing framework (p-value, Type I/II errors). Statistical Power and Sample Size calculation. A/B Testing: Design, Execution, and Interpretation. Non-parametric Tests for non-normal data: Mann-Whitney U test, Wilcoxon signed-rank test, Chi-Square tests for independence and goodness-of-fit.

Unit 2
Linear Regression Models: Simple and Multiple Linear Regression. Coefficient estimation using Ordinary Least Squares (OLS). Model Evaluation: R-squared, Adjusted R-squared, Root Mean Squared Error (RMSE). Assumptions of Linear Regression and Diagnostic Procedures: Checking for Linearity, Independence (Durbin-Watson test), Normality of residuals (Q-Q plots), and Homoscedasticity (Breusch-Pagan test). Handling Multicollinearity with Variance Inflation Factor (VIF).

Unit 3
Bayesian Inference in Practice: The Bayesian approach vs. the Frequentist approach. Components: Prior, Likelihood, Posterior, and Evidence. Conjugate Priors for simplifying posterior calculation. Bayesian Credible Intervals vs. Confidence Intervals. Introduction to computational methods for intractable posteriors: Markov Chain Monte Carlo (MCMC) and Gibbs Sampling.

Unit 4
Dimensionality Reduction and Manifold Learning: The "Curse of Dimensionality." Principal Component Analysis (PCA): Mathematical foundation via Eigen-decomposition of the covariance matrix, Scree Plots for component selection, and interpretation of principal components. Linear Discriminant Analysis (LDA) as a supervised alternative for classification. Introduction to manifold learning with t-SNE for visualization.

Unit 5
Resampling Methods and Generalized Linear Models: Computationally intensive methods for inference and validation. Cross-Validation techniques (k-fold, Leave-One-Out) for robust model evaluation. Bootstrapping for estimating the uncertainty of statistics. Introduction to Generalized Linear Models (GLMs): Link functions, and the statistical foundation of Logistic Regression for binary classification.

Information Security Fundamentals (CY201)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Security Concepts and Models: The CIA Triad (Confidentiality, Integrity, Availability). Security Attacks: Passive (Interception, Traffic Analysis) vs. Active (Modification, Masquerade, Replay, DoS). Security Services (X.800) and Mechanisms. The Model for Network Security. User Authentication: Identification vs. Authentication, Password Hashing, Salting, Rainbow Tables, and Dictionary Attacks.

Unit 2
Classical Cryptography and Number Theory: Substitution Ciphers (Caesar, Monoalphabetic, Playfair, Vigenère). Transposition Ciphers (Rail Fence, Columnar). Steganography. Mathematical Foundations: Modular Arithmetic, Euclidean Algorithm for GCD, Extended Euclidean Algorithm, Fermat's Little Theorem, Euler's Totient Function, and Primality Testing (Miller-Rabin).

Unit 3
Symmetric Cryptography: Stream Ciphers vs. Block Ciphers. The Feistel Cipher Structure. Data Encryption Standard (DES): Structure, Function, and Strength. Triple DES (3DES). Advanced Encryption Standard (AES): Transformations (SubBytes, ShiftRows, MixColumns, AddRoundKey). Block Cipher Modes of Operation: ECB, CBC, CFB, OFB, and CTR (Counter Mode).

Unit 4
Asymmetric Cryptography and Key Exchange: Principles of Public-Key Cryptosystems. The RSA Algorithm: Key Generation, Encryption, and Decryption mathematics. Security of RSA. Diffie-Hellman Key Exchange: The discrete logarithm problem, protocol, and Man-in-the-Middle (MITM) vulnerability. Introduction to Elliptic Curve Cryptography (ECC).

Unit 5
Data Integrity and Access Control: Cryptographic Hash Functions: Properties (Pre-image resistance, Second pre-image resistance, Collision resistance). The Secure Hash Algorithm (SHA) family. Message Authentication Codes (HMAC). Digital Signatures (DSA, RSA-PSS). Access Control Models: Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role-Based Access Control (RBAC).

Mechanics of Robotics (RB201)

← to Scheme

Credits: 3 (2-1-0)

Course Content

Unit 1
Spatial Descriptions and Transformations: Introduction to robotics, Robot components (links, joints), and Degrees of Freedom (DOF). Position and Orientation representation in 3D space. Coordinate Frames. Rotation Matrices and their properties. Homogeneous Transformation Matrices for representing combined rotation and translation. Composition of transformations.

Unit 2
Forward and Inverse Kinematics: Kinematic chains. Forward Kinematics: Calculating the end-effector's position and orientation from given joint angles. Denavit-Hartenberg (D-H) convention for systematically assigning coordinate frames to manipulators. Inverse Kinematics: Calculating the required joint angles to achieve a desired end-effector pose. Analytical vs. Numerical solutions. Workspace analysis.

Unit 3
Velocity Kinematics - The Jacobian: Linear and angular velocity of rigid bodies. The Jacobian matrix relating joint velocities to the end-effector's linear and angular velocities. Derivation of the Jacobian. Singularities: Configurations where the manipulator loses degrees of freedom and the Jacobian is not invertible. Relationship between Jacobian and static forces.

Unit 4
Manipulator Dynamics: Introduction to dynamics. Newton-Euler formulation for a single rigid body. Lagrangian Dynamics: A systematic approach based on kinetic and potential energy. Derivation of the equations of motion for simple manipulators. The structure of the dynamic model: Inertia matrix, Coriolis and Centrifugal terms, and Gravity vector.

Unit 5
Trajectory Planning and Introduction to Control: Path vs. Trajectory. Trajectory generation in joint-space and Cartesian-space. Polynomial trajectories (cubic, quintic) for smooth point-to-point motion. Introduction to robot control concepts. The need for feedback control. Proportional-Derivative (PD) control for joint-space motion. Introduction to different actuation systems (electric, hydraulic, pneumatic).

Semester IV

Linear Algebra & Optimization (MA202)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Matrices and Linear Systems: Matrix Algebra, Rank of a matrix, Reduction to Echelon and Normal forms. Consistency and solution of Systems of Linear Equations (Homogeneous and Non-homogeneous) using Gauss Elimination and Gauss-Jordan methods. LU Decomposition (Algorithmic approach to solving systems). Inverse of a matrix via elementary row operations.

Unit 2
Vector Spaces and Inner Products: Vector Spaces: Subspaces, Linear Independence/Dependence, Span, Basis, and Dimension. Linear Transformations: Kernel, Range, Rank-Nullity Theorem, and Matrix representation of transformations. Inner Product Spaces: Norms, Orthogonality, Orthogonal Projections, and the Gram-Schmidt Orthogonalization Process (QR Decomposition foundation).

Unit 3
Eigen Theory and Matrix Factorization: Eigenvalues and Eigenvectors: Characteristic equation, Algebraic and Geometric Multiplicity. Cayley-Hamilton Theorem. Diagonalization of Matrices. Real Symmetric Matrices and Orthogonal Diagonalization. Advanced Factorization: Introduction to Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) (Mathematical foundation for Data Compression and Dimensionality Reduction).

Unit 4
Linear Programming (Optimization): Introduction to Optimization and Convex Sets. Formulation of Linear Programming Problems (LPP). Solution methods: Graphical Method and the Simplex Algorithm (Big-M method, Two-phase method). Duality in Linear Programming: Primal-Dual relationships and Sensitivity Analysis (Economic interpretation of resources).

Unit 5
Non-Linear and Numerical Optimization: Unconstrained Optimization: Necessary and sufficient conditions for local extrema. Iterative Search Methods: Gradient Descent (Steepest Descent), Newton-Raphson Method, and Conjugate Gradient Method (Training logic for Neural Networks). Constrained Optimization: Lagrange Multipliers (Revisited) and Karush-Kuhn-Tucker (KKT) conditions for non-linear constraints.

Computer Organization & Architecture (CS204)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Fundamentals of Computer Organization: Basic structure of a computer system (CPU, memory, I/O), Data representation and formats (fixed-point, floating-point, character codes), ALU operations and status flags, Register organization and register transfer language, Bus structures and types of data paths, Instruction cycle (fetch, decode, execute) and micro-operations, Introduction to instruction set design and basic types of instructions (data movement, arithmetic/logic, control).

Unit 2
8085 and 8086 Architectures: Overview of 8-bit vs. 16-bit microprocessors, Internal architecture of 8085 (registers, ALU, control unit, buses) as a baseline model, 8086 architecture: Bus Interface Unit (BIU) and Execution Unit (EU), General-purpose, segment, index, and pointer registers, Memory segmentation and logical vs. physical addresses, Clock, reset, and basic timing concepts, Concept of a simple instruction pipeline via 8086 prefetch queue.

Unit 3
Instruction Sets, Addressing Modes, and Control: 8086 instruction format and classification (data transfer, arithmetic, logic, control flow, string operations), Addressing modes (immediate, register, direct, register indirect, indexed, based indexed, relative), Stack organization and procedure calls/returns, Condition codes and branching, Hardwired vs. microprogrammed control units, Micro-instructions and micro-operations, Introduction to interrupt structure (maskable vs. non-maskable) and basic exception handling model.

Unit 4
Memory Hierarchy and I/O Organization: Main memory organization (RAM, ROM, memory maps), Memory interleaving and bandwidth considerations, Cache memory hierarchy (L1/L2/L3), cache mapping techniques (direct-mapped, associative, set-associative), cache write policies and performance trade-offs, Concept of virtual memory and address translation, I/O-mapped vs. memory-mapped I/O, Programmed I/O, interrupt-driven I/O, and Direct Memory Access (DMA) for high-speed peripherals.

Unit 5
Modern Microarchitectures and Multi-Core Systems: RISC vs. CISC philosophies in contemporary processors (x86 vs. ARM), Pipelining hazards (structural, data, control) and high-level techniques to mitigate them, Overview of superscalar execution and out-of-order pipelines, Multi-core and many-core processors, shared memory and coherence concepts, System-on-Chip (SoC) integration in modern designs, Case studies of current CPU families (Intel/AMD x86 and Apple M-series) focusing on cores, caches, integrated accelerators (GPU/Neural engines), and unified memory architectures.

Operating Systems (CS205)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Overview of Operating Systems and System Calls: Role of an Operating System as resource manager and abstraction layer, OS services and components (process management, memory, storage, I/O), System structures (monolithic, layered, microkernel), User mode vs. kernel mode and protection boundary, System call interface and OS APIs, Introduction to context switching and mode switch, Basic case studies of UNIX/Linux and modern desktop/server OSes from a programmer's perspective.

Unit 2
Processes, Threads, and CPU Scheduling: Process concept and process states, Process control block (PCB) and context switch, Operations on processes (creation, termination, fork/exec model), Threads vs. processes, user-level vs. kernel-level threads, Multithreading models and typical threading issues, CPU scheduling goals and criteria, Classic scheduling algorithms (FCFS, SJF/SRTF, Priority, Round Robin, Multilevel queues), Starvation and aging, Interview-oriented reasoning about scheduler behavior and time complexity.

Unit 3
Synchronization, Concurrency, and Deadlocks: Race conditions and the critical section problem, Requirements for a correct synchronization solution (mutual exclusion, progress, bounded waiting), Software approaches (Peterson's algorithm, bakery algorithm), Hardware support for synchronization (test-and-set, compare-and-swap), Synchronization primitives (mutexes, semaphores, condition variables, monitors), Classical IPC problems (producer–consumer, readers–writers, dining philosophers), Deadlock characterization and necessary conditions, Deadlock prevention, avoidance (Banker's algorithm), detection, and recovery strategies, Typical interview problems around concurrency bugs.

Unit 4
Memory Management and Virtual Memory: Address spaces and memory allocation goals (relocation, protection, sharing), Contiguous allocation, fragmentation (internal vs. external), Paging and address translation, page tables and TLBs, Segmentation and segment-based protection, Demand paging and page faults, Page replacement algorithms (FIFO, Optimal, LRU and approximations), Thrashing and working set ideas, High-level view of how modern OSes manage process address spaces, stacks, heaps, and memory-mapped files.

Unit 5
File Systems, I/O, and Practical OS Perspectives: File concept, access methods, and directory structures, File system mounting, journaling, and basic consistency mechanisms, Space allocation methods (contiguous, linked, indexed), Free-space management (bitmaps, free lists), Overview of common file systems (e.g., ext-family, NTFS) from a developer's viewpoint, I/O subsystem organization, buffering, caching, and spooling, Disk scheduling algorithms (FCFS, SSTF, SCAN, C-SCAN, LOOK family), Introduction to virtualization and containers as OS-level abstractions, Mapping OS concepts to typical software engineering and interview problems (e.g., process vs. thread design, debugging deadlocks, performance tuning).

Operating Systems Lab (CX205)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Exploring System Calls and Processes
Focuses on writing simple programs to invoke basic OS services (file operations, process creation), observing user-mode vs. kernel-mode transitions using tools like `strace` or equivalent, and inspecting process attributes via proc-like interfaces.

Lab 2: Process Creation, Hierarchies, and Scheduling
Focuses on creating process trees using `fork`/`exec`-style patterns, measuring execution time under different loads, and empirically observing the impact of scheduling policies and priorities exposed by the underlying OS.

Lab 3: Thread Programming Basics
Focuses on implementing concurrent tasks using threads (e.g., `pthread` or language-level threads), sharing data structures between threads, and comparing process-based vs. thread-based designs for a simple workload such as parallel computation.

Lab 4: CPU-Bound vs. I/O-Bound Workloads
Focuses on designing CPU-bound and I/O-bound test programs, running them under different concurrency patterns, and empirically characterizing how the scheduler and blocking I/O behavior influence throughput and responsiveness.

Lab 5: Mutual Exclusion and Race Conditions
Focuses on constructing intentional race conditions on shared variables, then resolving them using mutexes and atomic operations, and analyzing incorrect vs. correct behavior with timing diagrams and logs.

Lab 6: Classical Synchronization Problems
Focuses on implementing canonical synchronization patterns (producer–consumer, readers–writers) using semaphores or condition variables, exploring buffer sizes and scheduling to understand starvation and fairness trade-offs.

Lab 7: Deadlocks in Practice
Focuses on building small programs that deliberately cause deadlocks (lock ordering, resource allocation), detecting and reproducing them reliably, and then modifying the design to eliminate deadlock using ordering or timeout strategies.

Lab 8: Memory Management and Paging Simulation
Focuses on simulating page replacement algorithms (FIFO, Optimal, LRU) over reference strings, computing page-fault statistics, and interpreting how locality affects the behavior of each algorithm in realistic workloads.

Lab 9: File System Operations and Caching Effects
Focuses on implementing file-intensive programs (sequential vs. random access patterns), measuring performance under varying access patterns, and reasoning about the role of buffering and OS-level caching in the observed behavior.

Lab 10: Mini Shell / Process Manager Capstone
Focuses on integrating process, thread, and I/O concepts by implementing a mini shell or process manager that can spawn and monitor processes, support basic job control features, and demonstrate handling of signals or termination conditions.

Programming Methodology (CS206)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Programming Language Taxonomy and Imperative Paradigm: Classification of programming languages by execution model (compiled vs interpreted, JIT), generation (1GL assembly to 5GL AI-based), and level of abstraction, Imperative programming fundamentals: von Neumann model, mutable state, sequential execution, and side effects, C/C++ as quintessential systems languages with manual memory management and zero-cost abstractions, Fortran legacy and its influence on scientific computing paradigms, Detailed study of language implementation trade-offs (performance vs. productivity, static vs. dynamic typing).

Unit 2
Object-Oriented Programming Across Languages: Core OOP principles (encapsulation, inheritance, polymorphism, abstraction) independent of syntax, Comparative implementations in Java (fully OOP with GC), C++ (multi-paradigm with RAII), Python (dynamic OOP), and C# (.NET ecosystem), Design patterns as language-agnostic solutions (Singleton, Factory, Observer, Strategy), Interface-based programming and dependency inversion, Language-specific idioms for polymorphism (virtual functions, interfaces, duck typing) and their runtime costs.

Unit 3
Functional Programming Concepts and Languages: Declarative vs. imperative styles, pure functions, immutability, and referential transparency, First-class and higher-order functions, Lambda calculus as theoretical foundation connecting to Theory of Computation (CS203), Haskell (lazy pure functional), Scala (hybrid OOP/FP on JVM), JavaScript (functional capabilities in browser/server), Closures, currying, and monads as abstraction mechanisms, Pattern matching and algebraic data types vs. traditional conditionals.

Unit 4
Scripting, Concurrent, and Domain-Specific Languages: Dynamic languages for rapid prototyping (Python, JavaScript, Ruby), scripting paradigms (shell, configuration DSLs), Garbage collection strategies and their impact on real-time systems, Concurrent programming models (shared memory threads vs. message passing actors), Go (goroutines/channels), Erlang (actor model), Domain-specific languages (SQL, HTML/CSS, shader languages, regex), Embedded DSLs vs. external DSLs, Metaprogramming (templates/macros in C++, decorators in Python, LINQ in C#).

Unit 5
Language Design for Compilers and Modern Ecosystems: Lexical structure, syntax vs. semantics, parsing strategies (LL/LR) connecting to compiler design preparation, Type systems (static/dynamic, strong/weak, nominal/structural), Memory models and safety guarantees across languages, Modern language ecosystems and interoperability (Python/C interop, Java Native Interface, WebAssembly), Language evolution case studies (C++ standardization, Python 2→3 migration, Rust for safe systems programming), Criteria for language selection in production systems based on performance, ecosystem, team expertise, and deployment constraints.

Programming Methodology Lab (CX206)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Imperative vs. Object-Oriented Implementations
Focuses on implementing the same algorithm (e.g., sorting network or matrix operations) using C-style procedural code vs. object-oriented class hierarchies in C++, observing compile-time vs. runtime differences and encapsulation benefits.

Lab 2: Cross-Language Function Equivalence
Focuses on writing identical functionality (e.g., data processing pipeline) in C++, Java, and Python, comparing compilation/execution models, type safety, and performance characteristics across the three language ecosystems.

Lab 3: Functional Programming Patterns
Focuses on implementing recursive algorithms (e.g., tree traversal, list processing) using functional style with higher-order functions and immutability in Python/JavaScript, contrasting with imperative loops and mutation.

Lab 4: Design Patterns Implementation
Focuses on implementing classic design patterns (Strategy, Observer, Factory) in two different languages (C++/Python), comparing syntax differences and runtime behavior for polymorphism and loose coupling.

Lab 5: Concurrent Programming Models
Focuses on implementing parallel workloads using threads (C++/Java) vs. goroutines (Go) or async/await (JavaScript/Python), measuring scalability and reasoning about race conditions across paradigms.

Lab 6: Domain-Specific Language Usage
Focuses on processing the same dataset using SQL queries, regex patterns, and shell scripting, then integrating results via Python, understanding how DSLs compose with general-purpose languages.

Lab 7: Metaprogramming Techniques
Focuses on code generation using C++ templates, Python decorators, and Java annotations, implementing a mini DSL or configuration system that demonstrates compile-time vs. runtime metaprogramming trade-offs.

Lab 8: Language Interoperability
Focuses on calling C/C++ code from Python (ctypes/cffi), Java Native Interface, or embedding Python in C++, building a hybrid application that leverages language strengths for different subsystems.

Lab 9: Parsing and Lexical Analysis
Focuses on implementing simple lexers/parsers for toy languages using regex, hand-coded state machines, and parser combinator libraries across languages, connecting theory from CS203 to practical language processing.

Lab 10: Language Ecosystem Capstone
Focuses on building a polyglot microservice (e.g., data pipeline with Python ETL + Go API + SQL backend), containerizing components, and measuring end-to-end performance across the language/runtime boundaries.

Innovation Lab III (IX202)

← Back to Scheme

Credits: 6 (0-0-6)
Note: Students continue in their selected track. This course focuses on enterprise-grade architecture, highly available distributed systems, and deep integration with operating systems/cloud infrastructure. The goal is to prepare students to design systems that handle massive scale and complex domain requirements, acting as a precursor to their final year Capstone Project.

Track A: Web Development (Distributed Systems & Cloud Architecture)

Task 1: Real-Time Protocols & Bidirectional Communication
Focuses on moving beyond the stateless HTTP request-response cycle. Students will use WebSockets and Server-Sent Events (SSE) to build a "Live Collaborative Whiteboard" (similar to Miro/Figma). They will learn the concepts of persistent TCP connections, broadcast mechanisms, socket rooms, and handling connection drops in real-time environments.

Task 2: Graph-Based API Design (GraphQL)
Focuses on declarative data fetching and solving REST bottlenecks. Students will transition from REST to GraphQL to build a "Customizable Analytics Dashboard." They will learn to mitigate Over-fetching and Under-fetching, write Resolvers, and understand the Abstract Syntax Tree (AST) of a query, allowing the View to dictate exactly what data the Controller should return.

Task 3: Event-Driven Architecture & Message Brokers
Focuses on decoupling backend services for scalability. Students will implement a message broker (RabbitMQ or Apache Kafka) to engineer an "Asynchronous Order Processing Pipeline." They will learn the Publisher/Subscriber pattern, event streaming, message durability, and how to prevent system failure by buffering sudden spikes in traffic through queues.

Task 4: Microservices & API Gateways
Focuses on breaking down monolithic architectures. Students will split a backend into separate, independent services (e.g., Auth Service, Inventory Service). They will configure an API Gateway (like NGINX or Kong) to act as a reverse proxy for a "Modular E-Commerce Backend." They will learn about service discovery, request routing, and cross-origin resource sharing (CORS) across multiple domains.

Task 5: High-Performance Binary Protocols (gRPC)
Focuses on efficient inter-service communication. Students will replace JSON payloads with Protocol Buffers (Protobuf). They will engineer a "High-Throughput Logging Service" where microservices communicate via gRPC. They will learn about payload serialization, strong typing in network contracts, and why binary transmission outperforms text-based JSON in microservice clusters.

Task 6: Object Storage & Content Delivery Networks (CDNs)
Focuses on handling massive media assets globally. Students will integrate AWS S3 (or equivalent) to build a "Video Hosting Platform." They will implement Multipart Uploads for large

files, generate Pre-signed URLs for secure direct-to-cloud uploads, and configure a CDN (CloudFront) to cache assets at edge locations, drastically reducing latency for global users.

Task 7: Serverless Compute & Edge Functions
Focuses on dynamic execution without server management. Students will deploy code to AWS Lambda or Cloudflare Workers. They will build a "Dynamic Image Resizer" that intercepts image requests and resizes them on the fly at the Edge. They will learn the concepts of "Cold Starts," stateless execution environments, and cost-optimization in serverless architectures.

Task 8: Distributed Search Engines
Focuses on rapid retrieval of unstructured data. Students will integrate a full-text search engine (Elasticsearch, Algolia, or Meilisearch). They will build a "Faceted Search Interface" capable of typo-tolerance and complex filtering. They will learn about inverted indexes, tokenization, and how search databases differ fundamentally from relational databases.

Task 9: System Observability & Telemetry
Focuses on understanding system health in a distributed environment. Students will instrument their microservices using OpenTelemetry. They will build a "System Health Dashboard" integrating Prometheus (for metrics) and Grafana (for visualization). They will learn the three pillars of observability: Logs, Metrics, and Distributed Tracing (tracking a single request across multiple microservices).

Task 10: Infrastructure as Code (IaC)
Focuses on automating cloud environments. Students will use Terraform or AWS CDK to programmatically define their entire infrastructure. They will deploy an "Automated Cloud Provisioning" script that spins up databases, VPCs, load balancers, and compute instances. They will learn the philosophy of Immutable Infrastructure, treating servers as "cattle, not pets," readying them for enterprise DevOps roles.

Track B: iOS Development (Deep Ecosystem & Enterprise Mobile)

Task 1: Real-Time Sockets & Background Keep-Alives
Focuses on maintaining persistent network connections on a mobile device. Students will use URLSessionWebSocketTask to build a "Live Multiplayer Trivia Client." They will learn how to handle WebSocket frames, manage background thread execution, and deal with the OS dropping connections when the app enters the background state.

Task 2: Advanced Media & HTTP Live Streaming (HLS)
Focuses on handling heavy continuous data streams. Students will utilize AVFoundation to build a "Custom HLS Video Player." They will learn how mobile devices handle m3u8 playlists, adaptive bitrate streaming (shifting video quality based on network strength), buffer management, and integrating Picture-in-Picture (PiP).

Task 3: On-Device Machine Learning (CoreML)
Focuses on leveraging the iPhone's Neural Engine. Students will integrate a pre-trained machine learning model using CoreML and the Vision framework. They will build a "Real-Time Object Detector" utilizing the device camera. They will learn about model quantization, running inference on the edge without internet connectivity, and processing CVPixelBuffers efficiently.

Task 4: Spatial Computing & Augmented Reality (ARKit)
Focuses on interacting with the physical world. Students will use ARKit and RealityKit to build an "Interactive Furniture Placer." They will learn the mathematics of Visual Inertial Odometry (VIO), plane detection, ray casting, and the rendering loop required to lock 3D assets to physical real-world coordinates.

Task 5: System Extensions & App Groups (WidgetKit)
Focuses on breaking out of the main application sandbox. Students will engineer a "Live Activity & Home Screen Widget" for an ongoing sports match. They will learn about Timeline Providers, strict memory-constrained environments, and establishing App Groups to share UserDefaults and database files securely between the main app and its extensions.

Task 6: Hardware Connectivity & IoT (CoreBluetooth)
Focuses on interfacing with external physical devices. Students will act as a Central Manager to build an "IoT Heart Rate Monitor" interface. They will learn the Bluetooth Low Energy (BLE) protocol, discovering peripherals, connecting, and subscribing to specific Characteristics and Services to read live sensor data.

Task 7: Graph Data on Mobile (Apollo Client)
Focuses on handling modern declarative APIs. Students will integrate Apollo iOS to connect to a GraphQL backend. They will build a "Deeply Nested Social Client." They will learn how to write '.graphql' files, generate strongly-typed Swift models automatically, and utilize Apollo's normalized local caching to drastically reduce network requests.

Task 8: Low-Level Interoperability (C/C++ in Swift)
Focuses on utilizing legacy code and maximizing performance. Students will learn how Swift interacts with lower-level languages. They will build a "High-Performance Audio Synthesizer" by bridging a C++ digital signal processing library into Swift. They will learn about bridging headers, unsafe memory pointers, and memory layout compatibility.

Task 9: App Thinning & On-Demand Resources (ODR)
Focuses on optimizing the application footprint. Students will learn how to reduce initial

download sizes to improve user acquisition. They will build an "Asset-Heavy Mobile Game Interface" that dynamically downloads levels and high-res textures only when the user reaches them. They will learn about App Slicing, asset catalogs, and managing ODR storage limits.

Task 10: Enterprise Architecture & Modularization (SPM)
Focuses on scaling codebases for large engineering teams. Students will break down a monolithic iOS app into smaller, isolated frameworks using Swift Package Manager (SPM). They will build a "Multi-Module Super App." They will learn the difference between static and dynamic libraries, managing compilation boundaries, and designing internal APIs, preparing them for Enterprise iOS roles.

Major Specialisation II

Software Architecture (SE202)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Architectural Foundations and Quality Attributes: Definition and role of software architecture in system development lifecycle, Architectural structures (modules, components, connectors, deployment), Quality attributes (performance, availability, security, maintainability, scalability) and their trade-offs, Architecture Influence Cycles and decision making frameworks, Non-functional requirements elicitation and specification using scenarios and utility trees.

Unit 2
Architectural Styles and Patterns: Classification of architectural styles (layered, client-server, pipe-filter, event-driven, service-oriented), Microservices architecture principles and decomposition strategies, Model-View-Controller (MVC) and variations (MVP, MVVM), Architectural patterns (Broker, Blackboard, Peer-to-Peer), RESTful principles and API design best practices for distributed systems.

Unit 3
Component-Based and Service-Oriented Architectures: Component principles (cohesion, coupling, interfaces, contracts), CORBA/CCS component model vs. modern container-based approaches, Service-Oriented Architecture (SOA) fundamentals, ESB patterns and anti-patterns, Microservices orchestration vs. choreography, Domain-Driven Design (DDD) bounded contexts and context mapping.

Unit 4
Architectural Views and Documentation: 4+1 View Model (Logical, Process, Development, Physical, Scenarios), UML 2.x structure and behavior diagrams for architecture representation, Architecture Description Languages (ADLs), Tool-supported architectural modeling (Enterprise Architect, ArchiMate), Documenting architectural decisions using ADRs (Architecture Decision Records).

Unit 5
Architecture Evaluation and Evolution: Architecture Tradeoff Analysis Method (ATAM), Software Architecture Analysis Method (SAAM), Scenario-based evaluation techniques, Risk-driven architecture evaluation, Refactoring legacy architectures, Evolutionary architecture principles (fitness functions, strangler pattern), Cloud-native architecture migration strategies.

Practical

Lab 1: Quality Attribute Scenarios
Focuses on eliciting and documenting non-functional requirements using quality attribute scenarios for a case study system, creating utility trees and prioritizing trade-offs.

Lab 2: Architectural Style Analysis
Focuses on analyzing existing systems to identify applied architectural styles, documenting style characteristics and limitations through reverse engineering exercises.

Lab 3: Microservices Decomposition
Focuses on decomposing monolithic applications into microservices using domain-driven design techniques, identifying bounded contexts and service boundaries.

Lab 4: MVC Implementation Patterns
Focuses on implementing the same business logic using different MVC variations across web frameworks, comparing controller responsibilities and view separation.

Lab 5: REST API Design
Focuses on designing and documenting RESTful APIs following HATEOAS principles, including resource modeling, HTTP methods, status codes, and hypermedia controls.

Lab 6: Component Interface Contracts
Focuses on defining and implementing component interfaces with versioning strategies, demonstrating contract-first design and interface segregation principles.

Lab 7: SOA vs Microservices Simulation
Focuses on implementing service communication patterns (synchronous RPC vs. asynchronous messaging) and measuring performance/scalability differences.

Lab 8: 4+1 View Modeling
Focuses on creating complete 4+1 architectural views for a medium-complexity system using UML tools, ensuring view consistency and scenario coverage.

Lab 9: Architecture Decision Records
Focuses on writing ADRs for architectural decisions in an evolving project, documenting context, decision, consequences, and status changes over time.

Lab 10: ATAM Evaluation Workshop
Focuses on conducting Architecture Tradeoff Analysis Method (ATAM) workshop on peer projects, identifying risks and documenting mitigation strategies.

Introduction to Machine Learning (ML202)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Foundations of Machine Learning: Core concepts and taxonomy of machine learning (supervised, unsupervised, reinforcement learning), Bias-variance tradeoff and model capacity, Overfitting, underfitting, and the No Free Lunch theorem, Evaluation metrics (accuracy, precision, recall, F1-score, ROC-AUC, confusion matrix), Cross-validation strategies (k-fold, stratified, time-series), Feature engineering fundamentals (scaling, encoding, feature selection).

Unit 2
Linear Models and Regression: Linear regression assumptions and ordinary least squares (OLS) solution, Gradient descent variants (batch, stochastic, mini-batch), Regularization techniques (L1/Lasso, L2/Ridge, Elastic Net), Logistic regression for binary classification, Probability interpretation and maximum likelihood estimation, Multinomial logistic regression and softmax, Model diagnostics (residual analysis, learning curves).

Unit 3
Nonlinear Models and Decision Boundaries: Decision trees (CART algorithm, entropy, Gini impurity), Tree ensemble methods (bagging, random forests), Gradient boosting machines (XGBoost fundamentals), Support Vector Machines (maximal margin separator, kernel trick), Kernel functions (linear, polynomial, RBF), Nonlinear separability and the curse of dimensionality.

Unit 4
Unsupervised Learning and Clustering: K-means clustering algorithm and limitations, Hierarchical clustering (agglomerative, divisive), Density-based clustering (DBSCAN), Dimensionality reduction techniques (PCA, t-SNE, UMAP), Anomaly detection methods (isolation forest, one-class SVM), Association rule mining (Apriori, FP-growth).

Unit 5
Model Selection, Optimization, and Deployment: Hyperparameter optimization (grid search, random search, Bayesian optimization), Ensemble learning principles (stacking, voting), Model interpretability techniques (SHAP, LIME, partial dependence plots), Introduction to ML pipelines and cross-validation pitfalls, MLOps concepts (model versioning, monitoring, retraining), Practical considerations for production ML systems.

Practical

Lab 1: ML Workflow and Evaluation Metrics
Focuses on implementing complete ML pipeline with data splitting, model training, and comprehensive evaluation using multiple metrics and cross-validation techniques.

Lab 2: Linear Regression Implementation
Focuses on implementing linear regression from scratch using gradient descent, comparing with scikit-learn implementation, and analyzing convergence behavior.

Lab 3: Logistic Regression and Classification
Focuses on binary classification problems, implementing logistic regression with regularization options, and interpreting probability outputs with ROC curves.

Lab 4: Decision Trees and Feature Importance
Focuses on building decision trees, visualizing tree structures, extracting feature importance, and pruning strategies to control overfitting.

Lab 5: Random Forests Ensemble
Focuses on implementing bagging with decision trees and comparing performance with single decision trees across multiple datasets.

Lab 6: SVM with Different Kernels
Focuses on experimenting with linear vs. nonlinear SVMs using various kernel functions and understanding the kernel trick through visualization.

Lab 7: K-means Clustering Analysis
Focuses on clustering datasets with known structures, evaluating cluster quality using silhouette scores, and handling initialization sensitivity.

Lab 8: Dimensionality Reduction Visualization
Focuses on applying PCA, t-SNE, and UMAP to high-dimensional datasets and creating insightful 2D/3D visualizations of data structure.

Lab 9: Hyperparameter Tuning Pipeline
Focuses on implementing grid search and random search for multiple algorithms, creating learning curves, and selecting optimal model configurations.

Lab 10: End-to-End ML Project
Focuses on complete ML project from data ingestion through model deployment, including feature engineering, model selection, evaluation, and basic MLOps practices.

Advanced SQL & NoSQL Databases (DS202)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Advanced SQL: Window Functions and Analytics: Window functions fundamentals (ROW_NUMBER, RANK, DENSE_RANK, NTILE), Aggregate window functions (SUM, AVG, COUNT over partitions), Framing clauses and sliding windows, LAG/LEAD functions for time-series analysis, FIRST_VALUE/LAST_VALUE and running totals, Common Table Expressions (CTEs) with recursive queries, Advanced JOIN patterns (self-joins, lateral joins).

Unit 2
Advanced SQL: Optimization and Performance: Query execution plans and cost estimation, Index types (B-tree, bitmap, hash, full-text), Composite indexes and covering indexes, Index maintenance and fragmentation, Query optimization strategies (rewrite rules, materialized views), EXPLAIN/ANALYZE usage, Partitioning strategies (range, list, hash partitioning), Statistics gathering and histogram usage.

Unit 3
Document Databases and NoSQL Fundamentals: NoSQL categories (document, key-value, column-family, graph), Document database model (MongoDB, CouchDB), Schema design for document stores (embedding vs. referencing), Query languages (MongoDB query language, JSONPath), Indexing strategies (compound, geospatial, text indexes), Aggregation pipelines and map-reduce in document stores.

Unit 4
Key-Value, Column-Family, and Wide-Column Stores: Key-value stores (Redis, DynamoDB) use cases and patterns (caching, sessions, leaderboards), Column-family databases (Cassandra, HBase), Wide-column model and denormalization strategies, CQL (Cassandra Query Language) vs. traditional SQL, Data modeling for high write throughput (partition keys, clustering columns), CAP theorem and consistency models.

Unit 5
Polyglot Persistence and Database Selection: Polyglot persistence strategies matching database types to workload patterns, SQL vs NoSQL selection criteria (ACID vs BASE, schema evolution), Hybrid architectures (SQL master + NoSQL cache/offload), NewSQL databases (CockroachDB, Yugabyte), Graph databases fundamentals (property graphs, Cypher query language), Time-series databases (InfluxDB, TimescaleDB).

Practical

Lab 1: Window Functions Analytics
Focuses on implementing running totals, ranking, and sliding window calculations using window functions across multiple datasets.

Lab 2: Recursive CTEs and Hierarchical Data
Focuses on solving hierarchical data problems (employee-manager trees, bill-of-materials) using recursive CTEs and path enumeration.

Lab 3: Query Optimization Analysis
Focuses on analyzing execution plans, creating strategic indexes, and measuring performance improvements using EXPLAIN ANALYZE.

Lab 4: Index Strategy Design
Focuses on designing composite indexes for complex queries, measuring covering index benefits, and analyzing fragmentation patterns.

Lab 5: MongoDB Document Modeling
Focuses on designing document schemas for application use cases, implementing embedding vs. referencing strategies with performance comparison.

Lab 6: MongoDB Aggregation Pipelines
Focuses on building complex aggregation pipelines with *lookup*,unwind, *group*, *and*facet stages for advanced analytics.

Lab 7: Cassandra Data Modeling
Focuses on designing partition/clustering keys for high-throughput workloads, implementing CQL queries matching the data model.

Lab 8: Redis Caching Patterns
Focuses on implementing cache-aside, write-through, and pub-sub patterns using Redis data structures and expiration policies.

Lab 9: Polyglot Persistence Integration
Focuses on building hybrid applications using SQL primary store + NoSQL cache/offload, measuring consistency/performance trade-offs.

Lab 10: Database Selection Case Study
Focuses on evaluating database technologies against workload requirements, creating polyglot persistence recommendations with cost analysis.

Cryptography & Network Security (CY202)

← Back to Scheme

Credits: 3 (2-1-0)

Theory

Unit 1
Cryptography Fundamentals and Classical Techniques: Security services (confidentiality, integrity, authentication, non-repudiation), Classical ciphers (Caesar, Vigenère, Playfair, Hill), Cryptanalysis techniques (frequency analysis, known-plaintext attacks), Block vs. stream ciphers, Shannon's principles of confusion and diffusion, Modes of block cipher operation (ECB, CBC, CFB, OFB, CTR), Initialization vectors and nonce usage.

Unit 2
Symmetric Key Cryptography: Data Encryption Standard (DES) algorithm and cryptanalysis (differential, linear), Advanced Encryption Standard (AES) - Rijndael structure, S-boxes, key expansion, rounds structure, Block cipher modes security analysis, Key management challenges in symmetric cryptography, Key derivation functions and key stretching, Practical symmetric cipher selection criteria.

Unit 3
Number Theory and Public-Key Cryptography: Modular arithmetic, prime generation, Fermat/Euler theorems, Greatest Common Divisor algorithms, RSA cryptosystem (key generation, encryption/decryption, security proofs), Diffie-Hellman key exchange and man-in-the-middle attacks, Digital Signature Algorithm (DSA), Elliptic Curve Cryptography (ECC) fundamentals and advantages over RSA.

Unit 4
Hash Functions, MACs, and Authentication: Cryptographic hash function properties (collision resistance, preimage resistance), MD5 weaknesses and SHA-family algorithms, HMAC construction and security proofs, Message authentication codes (MACs), Digital signatures and non-repudiation, PKI concepts (certificates, CAs, CRLs), X.509 certificate format and validation.

Unit 5
Network Security Protocols and Applications: Transport Layer Security (TLS/SSL) handshake and cipher suites, IPsec architecture (AH, ESP, IKE), VPN protocols and tunnel modes, Email security (PGP, S/MIME), Kerberos authentication and ticket granting, Common network attacks (MITM, replay, session hijacking), Secure protocol design principles and cryptographic agility.

Practical

Lab 1: Classical Cipher Implementation
Focuses on implementing and cryptanalyzing classical ciphers (Caesar, Vigenère), demonstrating frequency analysis attacks and basic cryptanalysis techniques.

Lab 2: Block Cipher Modes Analysis
Focuses on implementing ECB/CBC/CTR modes, demonstrating padding oracle attacks on ECB and malleability issues in CBC mode.

Lab 3: AES Implementation Study
Focuses on studying AES encryption/decryption process, key expansion, and analyzing performance characteristics across different key sizes.

Lab 4: RSA Key Generation and Attacks
Focuses on RSA implementation, textbook vs. padded RSA, demonstrating small exponent attacks and common modulus attacks.

Lab 5: Diffie-Hellman Key Exchange
Focuses on implementing Diffie-Hellman key exchange, demonstrating man-in-the-middle attacks and ephemeral key usage.

Lab 6: Hash Function Collision Attacks
Focuses on demonstrating MD5 collision attacks and analyzing SHA-256 preimage resistance through birthday paradox experiments.

Lab 7: HMAC and Message Authentication
Focuses on implementing HMAC construction and verifying security properties against length extension attacks.

Lab 8: Digital Signature Verification
Focuses on creating/verifying digital signatures using RSA/ECDSA, implementing X.509 certificate chain validation.

Lab 9: TLS Handshake Analysis
Focuses on capturing and analyzing TLS handshakes using Wireshark, identifying cipher suite negotiation and certificate validation.

Lab 10: Network Security Protocol Implementation
Focuses on implementing simplified TLS/IPsec protocol components, demonstrating secure key exchange and data protection.

Sensors, Actuators & Signal Processing (RB202)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Sensor Fundamentals and Classification: Sensor principles and transduction mechanisms (resistive, capacitive, inductive, piezoelectric, optical), Sensor specifications (sensitivity, resolution, accuracy, hysteresis, linearity), Active vs. passive sensors, Analog vs. digital sensors, Environmental factors affecting sensor performance (temperature drift, noise, EMI), Sensor fusion concepts and Kalman filtering introduction.

Unit 2
Physical Sensors for Robotics: Position and displacement sensors (potentiometers, LVDT, optical encoders, resolvers), Proximity and range sensors (ultrasonic, infrared, laser time-of-flight, LIDAR), Force/torque sensors (strain gauges, piezoelectric), Temperature sensors (thermocouples, RTDs, thermistors, infrared), Accelerometers and gyroscopes (MEMS principles, frequency response).

Unit 3
Signal Conditioning and Interface Circuits: Analog signal conditioning (amplification, filtering, offset compensation), Instrumentation amplifiers and bridge circuits, Anti-aliasing filters and sampling theory, ADC/DAC fundamentals and selection criteria, Digital signal preprocessing (decimation, FIR/IIR filtering), Excitation circuits for active sensors (constant current/voltage sources).

Unit 4
Actuators and Drive Electronics: DC motors (characteristics, torque-speed curves, commutation), Stepper motors (full/half step, microstepping), Servo motors (position feedback loops), Solenoids and voice coil actuators, Pneumatic/hydraulic actuators, Piezoelectric actuators, H-bridge drivers, PWM techniques and motor control algorithms.

Unit 5
Digital Signal Processing for Sensor Data: Time-domain analysis (signal averaging, peak detection), Frequency-domain analysis (FFT fundamentals, spectral leakage), Digital filters (FIR/IIR design, stability), Noise reduction techniques (matched filtering, wavelet denoising), Sensor calibration and characterization, Real-time signal processing constraints and embedded implementation considerations.

Practical

Lab 1: Sensor Characterization
Focuses on measuring and documenting sensor specifications (sensitivity, linearity, hysteresis) using controlled test setups and data acquisition systems.

Lab 2: Wheatstone Bridge and Strain Gauges
Focuses on implementing instrumentation amplifier circuits for strain gauge measurements and characterizing force/load relationships.

Lab 3: IMU Sensor Fusion
Focuses on interfacing accelerometers and gyroscopes, implementing complementary and Kalman filters for attitude estimation.

Lab 4: Ultrasonic and IR Ranging
Focuses on comparing time-of-flight vs. triangulation ranging methods, characterizing accuracy vs. distance profiles.

Lab 5: Anti-Aliasing Filter Design
Focuses on designing and testing analog anti-aliasing filters, verifying Nyquist compliance with oscilloscope measurements.

Lab 6: ADC Performance Analysis
Focuses on characterizing ADC resolution, sampling rate, and ENOB through signal reconstruction and SNR measurements.

Lab 7: DC Motor Characterization
Focuses on measuring motor torque-speed curves, implementing open-loop speed control using PWM duty cycle variation.

Lab 8: Servo Motor Position Control
Focuses on implementing PID position control loops for servo motors with setpoint tracking and disturbance rejection analysis.

Lab 9: Digital Filter Implementation
Focuses on designing and implementing FIR/IIR filters for sensor noise reduction, comparing frequency responses.

Lab 10: Sensor-Actuator Integration
Focuses on building complete closed-loop systems combining multiple sensors with actuator control, implementing sensor fusion and real-time processing.

Semester V

Computer Networks (CS301)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Network Fundamentals and Reference Models: Network types (LAN, MAN, WAN, PAN) and topologies, Client-server vs. peer-to-peer architectures, OSI vs. TCP/IP models with layer responsibilities, Encapsulation/decapsulation process, Protocol data units (PDU, frame, packet, segment), Network devices (hubs, switches, routers, gateways), Performance metrics (bandwidth, throughput, latency, jitter, goodput), Interview questions on layered architectures and protocol interactions.

Unit 2
Physical and Data Link Layers: Transmission media (twisted pair, coaxial, fiber optic, wireless), Signal encoding (NRZ, Manchester, 4B/5B), Error detection (parity, checksum, CRC polynomials), Error correction (Hamming codes), Framing methods (byte/packet/character stuffing, bit-oriented), Flow control (stop-and-wait, sliding window), ARQ protocols (Go-Back-N, Selective Repeat), MAC protocols (ALOHA, CSMA/CD, CSMA/CA).

Unit 3
Network Layer - Addressing and Routing: IPv4 addressing (classes, CIDR, subnetting, VLSM, supernetting), IPv6 addressing and transition mechanisms, ARP/RARP protocol operation, ICMP error reporting and diagnostics, Routing algorithms (distance vector, link state, hierarchical), Routing protocols (RIP, OSPF, BGP), Static vs. dynamic routing, Route summarization and longest prefix matching.

Unit 4
Network Layer - Congestion and Internetworking: Congestion control mechanisms (leaky bucket, token bucket), Quality of Service (QoS) concepts (DiffServ, IntServ), NAT/PAT operation and port address translation, Fragmentation and reassembly, MPLS basics and label switching, Internetworking challenges (heterogeneous networks, fragmentation, MTU issues), Interview problems on subnet calculation, CIDR aggregation, and traceroute analysis.

Unit 5
Transport and Application Layers: UDP protocol characteristics and use cases, TCP protocol (3-way handshake, state diagram, connection release), TCP flow control (sliding window, silly window syndrome), TCP congestion control (slow start, congestion avoidance, fast retransmit/recovery), Reliable data transfer principles, Application layer protocols (DNS resolution, HTTP/HTTPS, SMTP, FTP, DHCP), Socket programming fundamentals and port number assignments.

Computer Networks Lab (CX301)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: Network Device Study and Cable Crimping
Focuses on identifying network devices (hub, switch, router), crimping straight-through and crossover cables, verifying connectivity using cable testers.

Lab 2: Basic Network Commands and Troubleshooting
Focuses on using ping, traceroute, nslookup, netstat, ifconfig/ipconfig, analyzing connectivity issues and DNS resolution problems.

Lab 3: IPv4 Subnetting and IP Addressing
Focuses on calculating subnets, configuring IP addresses/subnets/VLSM, verifying connectivity across different subnets using Packet Tracer.

Lab 4: ARP Protocol Analysis
Focuses on capturing ARP requests/replies using Wireshark, configuring static ARP entries, analyzing ARP cache poisoning scenarios.

Lab 5: Ethernet Frame Analysis
Focuses on capturing Ethernet frames with Wireshark, analyzing frame structure (preamble, header, CRC), observing CSMA/CD behavior.

Lab 6: VLAN Configuration and Trunking
Focuses on configuring VLANs on managed switches, setting up trunk links (802.1Q), inter-VLAN routing using router-on-a-stick.

Lab 7: Static Routing Configuration
Focuses on configuring static routes between multiple routers, verifying routing tables, analyzing path determination using show ip route.

Lab 8: Dynamic Routing Protocols (RIP/OSPF)
Focuses on configuring RIP and OSPF, observing route convergence, analyzing routing protocol metrics and neighbor relationships.

Lab 9: TCP 3-Way Handshake and Connection States
Focuses on capturing TCP handshake packets with Wireshark, configuring socket programs to observe SYN/ACK/FIN states.

Lab 10: Network Simulation and Congestion Analysis
Focuses on building multi-router topologies in Packet Tracer, generating traffic to observe congestion, analyzing throughput vs. packet loss.

Compiler Design (CS302)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Compiler Structure and Lexical Analysis: Role of compilers vs interpreters, Phases of compiler (lexical, syntax, semantic, optimization, code generation), Symbol tables and their organization (hash tables, trees), Lexical analysis fundamentals connecting to finite automata (DFA/NFA from CS203), Regular expressions to NFA conversion, Lexical analyzer generators (Lex/Flex tool overview), Token classification and symbol table management, Handling identifiers, keywords, literals, and comments.

Unit 2
Top-Down and Bottom-Up Parsing: Parsing as recognition problem (connecting CFG from CS203 to practical parsing), Top-down parsing (recursive descent, LL(1) parsers, FIRST/FOLLOW sets), Handling left recursion and left factoring, Predictive parsing tables, Bottom-up parsing (shift-reduce, LR(0), SLR(1), LALR(1), CLR(1)), Handle finding and parsing conflicts (shift-reduce, reduce-reduce), Parser generators (Yacc/Bison tool overview).

Unit 3
Syntax-Directed Translation and Semantics: Syntax-directed definitions (S-attributed, L-attributed), Syntax-directed translation schemes, Abstract syntax trees (AST) construction, Semantic analysis connecting to Programming Methodology paradigms (CS206), Type systems (static/dynamic typing, type equivalence, type inference), Type checking algorithms (structural/declarative), Symbol table operations during semantic analysis, Attribute grammars for semantic rules.

Unit 4
Intermediate Code Generation and Runtime: Three-address code (TAC) forms (triple, indirect triple), Syntax trees to TAC translation, Boolean expressions and control flow (DAG construction), Runtime environments (activation records, stack allocation), Parameter passing mechanisms (call-by-value/reference), Storage allocation (static/dynamic, heap management), Garbage collection basics (mark-and-sweep, reference counting), Intermediate representations (IR) for optimization preparation.

Unit 5
Code Optimization and Generation: Local optimization (constant folding, strength reduction, common subexpression elimination), Global optimization (data flow analysis - reaching definitions, live variables, available expressions), Control flow graphs and dominators, Loop optimizations (invariant code motion, induction variable elimination), Register allocation (graph coloring, spilling), Instruction selection and peephole optimization, Target machine characteristics and code generation strategies.

Innovation Lab IV (IX301): Capstone Product Development

← Back to Scheme

Credits: 12 (0-0-12) **Assuming a heavier weight for the final year**
Note: This is an individual, year-long zero-to-one engineering capstone. There is no weekly instructional curriculum. Students must formulate a real-world problem statement and independently engineer a market-ready product. To prevent trivial submissions, projects will strictly be evaluated against enterprise-grade architectural and business requirements.

Track A: Web Development (SaaS Product Engineering)

Aim: The student must architect, build, and deploy a fully functional Software-as-a-Service (SaaS) platform capable of handling real users, transactions, and concurrent traffic.

Requirement 1: Multi-Tenant Architecture & Advanced Auth
The platform cannot be a simple single-user app. It must support multiple isolated organizations or tenants (e.g., a B2B SaaS). It must implement robust Role-Based Access Control (RBAC) with hierarchical permissions (Owner, Admin, Viewer), secure session management, and OAuth2.0 integration (Google/GitHub SSO).

Requirement 2: Complex Business Logic & Asynchronous Processing
The system must transcend basic CRUD operations. It must feature a complex core engine (e.g., scheduling algorithms, automated document generation, data aggregation). Heavy computations or third-party API integrations must be decoupled from the main thread using background job queues (e.g., Redis/BullMQ) and Webhooks.

Requirement 3: High-Performance Data & Caching Layer
The database architecture must be strictly normalized (PostgreSQL/MySQL) with proper indexing. The platform must implement a caching strategy (Redis or Memcached) to minimize database hits on high-traffic endpoints. Search functionality must be optimized (e.g., using Elasticsearch or vector embeddings).

Requirement 4: Monetization & External Integrations
A true SaaS must be able to accept money. The platform must integrate a payment gateway (Stripe, Razorpay, or Paddle) handling complex billing scenarios such as recurring subscriptions, usage-based metering, or tier-based access control, complete with automated invoice generation and webhook event handling.

Requirement 5: Infrastructure, DevOps, & Observability
The application must not be manually deployed. Students must configure a fully automated CI/CD pipeline (GitHub Actions) that runs integration tests before containerizing the app via Docker. The production environment must include an SSL certificate, custom domain routing, and telemetry (Logs and Metrics via Prometheus/Grafana or Datadog) to monitor uptime and API latency.

Track B: iOS Development (App Store Deep-Tech Product)

Aim: The student must design, engineer, and publish a native, high-performance iOS application that strictly adheres to Apple's Human Interface Guidelines (HIG) and leverages the unique hardware capabilities of the device.

Requirement 1: Deep Hardware or OS Ecosystem Integration
The application cannot simply be a wrapper for a web API. It must rely heavily on native iOS frameworks to solve a problem. This includes significant utilization of at least two of the following: CoreML (On-device AI), ARKit (Spatial Computing), HealthKit, CoreBluetooth (IoT), or background location tracking, making the app uniquely "Mobile-First."

Requirement 2: Offline-First Architecture & Data Synchronization
The app must remain fully functional without an internet connection. Students must implement a robust local database (SwiftData or Realm) combined with an intelligent background synchronization engine. Conflict resolution strategies must be implemented for when the device reconnects to the network to ensure data integrity with the cloud backend.

Requirement 3: Modular Codebase & Reactive State Management
The codebase must be structurally sound and maintainable. It must be split into independent modules using Swift Package Manager (SPM). The UI layer must strictly decouple from business logic using a modern architectural pattern (MVVM, VIPER, or The Composable Architecture), with completely reactive state management (Combine or Swift Async/Await).

Requirement 4: Memory Optimization & UI Fluidity
The application must pass rigorous profiling using Xcode Instruments. It must demonstrate zero memory leaks (resolving all retain cycles) and maintain a consistent 60/120 FPS scrolling performance, even when rendering massive data sets or complex CoreAnimation/Metal graphics.

Requirement 5: App Store Readiness & Monetization
The product must be packaged for public consumption. This includes implementing In-App Purchases (IAP) or subscriptions (via StoreKit or RevenueCat), integrating crash reporting tools (Firebase Crashlytics), passing automated UI/Unit tests on a CI/CD pipeline (Fastlane), and culminating in a successful submission to TestFlight and the iOS App Store Review process.

Major Specialisation III

Cloud Computing Fundamentals (SE301)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Cloud Computing Foundations and Service Models: Essential characteristics of cloud computing (on-demand, elasticity, pay-as-you-go), Service models (IaaS, PaaS, SaaS, FaaS), Deployment models (public, private, hybrid, multi-cloud), Cloud value proposition (capital vs. operational expenditure, scalability economics), Shared responsibility model, Cloud-native application principles and twelve-factor app methodology.

Unit 2
Compute Services and Container Orchestration: Virtual machines and instance types across major clouds (EC2, Azure VMs, GCP Compute Engine), Auto Scaling Groups and launch templates, Managed Kubernetes services (EKS, AKS, GKE), Containerization fundamentals (Docker images, registries), Orchestration concepts (pods, deployments, services, ingress), Serverless compute (AWS Lambda, Azure Functions, Cloud Run) and event-driven architectures.

Unit 3
Storage Services and Data Management: Object storage (S3, Blob Storage, Cloud Storage) characteristics and use cases, Block storage (EBS, managed disks) for databases, File storage (EFS, NFS) for shared access, Content Delivery Networks (CloudFront, CDN), Database services (RDS, DynamoDB, DocumentDB, BigQuery), Data transfer services (DataSync, Storage Gateway), Backup and disaster recovery strategies.

Unit 4
Networking, Security, and Identity: Virtual networking (VPC, subnets, route tables, NAT gateways, security groups), Load balancing (ALB, NLB, Global Accelerator), DNS management (Route 53, Azure DNS), Identity and Access Management (IAM roles, policies, MFA), Secrets management (Secrets Manager, Key Vault), Encryption at rest/transit, Web Application Firewall (WAF), Security best practices and compliance frameworks.

Unit 5
Monitoring, CI/CD, and Cost Management: CloudWatch, Azure Monitor, Stackdriver for metrics, logs, alarms, Application Performance Monitoring (APM), Infrastructure as Code (CloudFormation, Terraform, ARM templates), CI/CD pipelines (CodePipeline, GitHub Actions, Azure DevOps), Cost optimization strategies (reserved instances, spot instances, cost allocation tags), Well-Architected Framework reviews, Multi-cloud and hybrid cloud strategies.

Deep Learning & Neural Networks (ML301)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Neural Networks Fundamentals: Perceptron and multi-layer perceptron (MLP), Activation functions (sigmoid, tanh, ReLU, Leaky ReLU, Swish), Forward propagation and backpropagation algorithm derivation, Gradient descent optimization (SGD, momentum, AdaGrad, RMSprop, Adam), Vanishing/exploding gradients problem, Weight initialization strategies (Xavier, He), Universal approximation theorem.

Unit 2
Convolutional Neural Networks: Motivation for CNNs in image processing, Convolution operation and feature maps, Pooling layers (max, average, global), CNN architectures (LeNet, AlexNet, VGG, ResNet residual connections), Transfer learning and fine-tuning strategies, Data augmentation techniques, Batch normalization and layer normalization, Object detection fundamentals (sliding window, region proposals).

Unit 3
Recurrent Networks and Sequence Models: RNN architecture and vanishing gradient problem, Long Short-Term Memory (LSTM) cells and gates, Gated Recurrent Units (GRU), Bidirectional RNNs, Sequence-to-sequence models with attention mechanisms, Beam search decoding, Transformer architecture (self-attention, multi-head attention, positional encoding), BERT and GPT model families.

Unit 4
Advanced Architectures and Generative Models: Generative Adversarial Networks (GANs) - minimax game, training stability, DCGAN improvements, Variational Autoencoders (VAEs) - encoder-decoder, KL divergence, Diffusion models fundamentals (forward/reverse diffusion process, DDPM), Autoencoders and dimensionality reduction, Graph Neural Networks (GNN) introduction, Model compression techniques (pruning, quantization).

Unit 5
Training, Optimization, and Deployment: Loss functions (cross-entropy, MSE, Dice, contrastive losses), Learning rate schedules and schedulers, Gradient clipping, Mixed precision training, Distributed training strategies (data parallel, model parallel), Overfitting prevention (dropout, early stopping, data augmentation), Model evaluation for deep learning (confusion matrix, precision-recall curves), TensorRT/ONNX deployment, Edge deployment considerations.

Practical

Lab 1: MLP Implementation and Training
Focuses on implementing MLP from scratch, experimenting with activation functions and optimizers, visualizing decision boundaries.

Lab 2: CNN for Image Classification
Focuses on building CNNs for CIFAR10/MNIST, implementing convolution/pooling layers, transfer learning from pretrained models.

Lab 3: RNN/LSTM for Sequence Prediction
Focuses on character-level text generation and time-series forecasting using LSTM/GRU networks.

Lab 4: Attention and Transformer Basics
Focuses on implementing self-attention mechanisms and simple transformer encoder for text classification.

Lab 5: GAN Implementation
Focuses on training DCGAN for MNIST digit generation, analyzing training dynamics and mode collapse.

Lab 6: VAE and Diffusion Model Sampling
Focuses on implementing VAE for image reconstruction and basic diffusion model denoising process.

Lab 7: Object Detection Pipeline
Focuses on using YOLO/SSD for object detection, evaluating mAP metrics on custom datasets.

Lab 8: Advanced Training Techniques
Focuses on implementing learning rate scheduling, gradient clipping, mixed precision training.

Lab 9: Model Optimization and Compression
Focuses on pruning, quantization, and knowledge distillation for deployment-ready models.

Lab 10: End-to-End DL Project
Focuses on complete pipeline from data collection through model training, evaluation, and deployment using cloud services.

Data Visualization & Storytelling (DS301)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Foundations of Data Visualization: Data visualization principles and human perception (pre-attentive processing, Gestalt principles), Data types and appropriate encodings (position, length, angle, area, color, shape), Chart taxonomy (categorical, temporal, spatial, hierarchical, network data), Color theory for visualization (perceptual uniformity, color blindness accessibility), Visual encoding principles (Tufte's data-ink ratio, Cleveland/McGill hierarchy).

Unit 2
Statistical Graphics and Exploratory Analysis: Exploratory data analysis (EDA) workflow, Univariate distributions (histograms, box plots, violin plots, density plots), Bivariate relationships (scatterplots, heatmaps, correlation matrices), Multivariate visualization techniques (parallel coordinates, scatterplot matrices, dimensionality reduction), Outlier detection and anomaly visualization strategies.

Unit 3
Geospatial and Network Visualization: Choropleth maps and spatial aggregation pitfalls, Proportional symbol maps, Flow maps and origin-destination visualization, Spatial autocorrelation and clustering (Moran's I), Network visualization techniques (node-link diagrams, adjacency matrices, arc diagrams), Force-directed layouts, hierarchical edge bundling.

Unit 4
Interactive and Dynamic Visualizations: Event handling and brushing/linking, Zooming/panning/filtering interactions, Animated transitions and small multiples, Dashboard design principles (layout, cognitive load, task prioritization), Storytelling techniques (scrollytelling, annotated charts, guided narratives), Level-of-detail management and progressive disclosure.

Unit 5
Data Storytelling and Presentation: Narrative structures for data stories (explanatory vs. exploratory analysis), Visual hierarchy and emphasis techniques, Audience analysis and stakeholder communication, Presentation design (slide layouts, chart choice, annotation), Dashboard evaluation frameworks (USE framework, stakeholder interviews), Ethical considerations (visual manipulation, statistical fallacies, misleading scales).

Practical

Lab 1: Univariate and Bivariate Charts
Focuses on creating histograms, box plots, scatterplots, and correlation matrices using proper visual encodings and accessibility considerations.

Lab 2: Multivariate EDA Visualizations
Focuses on parallel coordinates, scatterplot matrices, and dimensionality reduction visualizations for pattern discovery.

Lab 3: Geospatial Data Visualization
Focuses on choropleth maps, proportional symbols, and flow maps using real-world geographic datasets.

Lab 4: Network Graph Visualization
Focuses on node-link diagrams, force-directed layouts, and network metrics visualization.

Lab 5: Interactive Chart Development
Focuses on implementing brushing, filtering, and zooming interactions with D3.js or similar libraries.

Lab 6: Animated Transitions and Storytelling
Focuses on smooth transitions between visualization states and scrollytelling implementations.

Lab 7: Dashboard Design and Layout
Focuses on creating multi-chart dashboards with proper layout, responsive design, and interaction coordination.

Lab 8: Data Story Development
Focuses on complete narrative development from raw data through publication-ready visualizations.

Lab 9: Presentation and Stakeholder Communication
Focuses on presentation design, audience adaptation, and receiving stakeholder feedback.

Lab 10: Visualization Critique and Improvement
Focuses on analyzing existing visualizations, identifying flaws, and redesigning for better clarity and impact.

Secure Network Protocols (CY301)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Secure Protocol Design Principles: CIA triad in networked environments, Threat modeling for protocols (STRIDE methodology), Secure protocol design patterns (encrypt-then-authenticate, stateful/stateless protocols), Cryptographic agility and algorithm lifecycle management, Protocol downgrade attacks and version negotiation vulnerabilities, Perfect Forward Secrecy (PFS) requirements, Side-channel attack considerations in protocol design.

Unit 2
TLS/SSL Protocol Deep Dive: TLS handshake protocol (client hello, server hello, key exchange, certificate verification), Cipher suite negotiation and security implications, Record protocol and fragmentation, Alert protocol handling, Session resumption and tickets, TLS 1.2 vs 1.3 differences (0-RTT, encrypted SNI), Certificate chain validation and revocation checking (OCSP, CRL).

Unit 3
IPsec Architecture and Protocols: IPsec protocol suite overview (AH, ESP, IKE), Security associations and databases (SAD, SPD), ESP packet format and encryption/authentication modes, AH integrity protection mechanism, IKEv1 vs IKEv2 (main/aggressive mode, quick mode), Diffie-Hellman key exchange in IKE, NAT traversal and MOBIKE extensions.

Unit 4
Wireless and Application Layer Security: WPA2/WPA3 protocol analysis (4-way handshake, PMK/PTK derivation), Enterprise vs Personal modes, Dragonfly handshake in WPA3, HTTPS ecosystem (HSTS, HPKP, Certificate Transparency), DNS security (DNSSEC, DANE, DoH/DoT), SSH protocol (key exchange, authentication, channel multiplexing), Secure email protocols (S/MIME, PGP/OpenPGP).

Unit 5
Protocol Vulnerabilities and Modern Protocols: Historical protocol failures (SSH1 CRC32, TLS renegotiation, BEAST, POODLE, Heartbleed), Protocol implementation flaws (timing attacks, padding oracles), QUIC protocol (0-RTT, connection migration, multiplexing), WireGuard protocol design, Signal protocol for messaging (double ratchet, X3DH), Post-quantum cryptography considerations for protocols.

Embedded Systems & Microcontrollers (RB301)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Embedded Systems Architecture and Programming: Embedded system characteristics (real-time constraints, resource limitations), Microcontroller vs microprocessor architectures, ARM Cortex-M series architecture overview, Harvard vs Von Neumann memory architectures, Interrupt handling (NVIC, priority levels), Bare-metal programming vs RTOS, Power management (sleep modes, clock gating, dynamic voltage scaling).

Unit 2
Arduino Ecosystem and Peripheral Programming: Arduino architecture (ATmega328P, pin multiplexing), GPIO programming (digital input/output, pull-up/pull-down), Analog-to-digital conversion (ADC channels, reference selection), Pulse Width Modulation (PWM) for motor control, Serial communication (UART, SPI, I2C protocols), Timer/Counter peripherals (overflow, compare match modes), Arduino IDE workflow and bootloader.

Unit 3
Wireless Microcontrollers (ESP32/ESP8266): ESP32 SoC architecture (dual-core Xtensa LX6, ultra-low power coprocessor), WiFi and Bluetooth stack integration, FreeRTOS port for ESP32, MQTT protocol implementation, Over-The-Air (OTA) firmware updates, Deep sleep and ULP coprocessor for battery-powered applications, GPIO matrix and peripheral routing flexibility.

Unit 4
Edge AI Platforms (NVIDIA Jetson): Jetson Nano/TX2 architecture (Carver SoC, GPU compute capability), CUDA programming model for embedded GPUs, TensorRT optimization for deep learning inference, NVIDIA container runtime (JetPack SDK), GPIO/I2C/SPI/UART interfacing with robotics peripherals, Camera interface (CSI, MIPI), Power modes and thermal management.

Unit 5
Real-Time Operating Systems and Robotics Integration: FreeRTOS fundamentals (tasks, queues, semaphores, mutexes), Task scheduling (preemptive, priority inheritance), ROS2 Micro (micro-ROS) for embedded systems, Device drivers and HAL implementation, Sensor fusion on microcontrollers (Madgwick/Kalman filters), Communication protocols for robotics (CAN bus, ROS topics/services), Hardware abstraction layers for multi-platform development.

Practical

Lab 1: Arduino GPIO and Digital Interfacing
Focuses on digital input/output programming, debouncing techniques, LED matrix control, and interrupt-driven button handling.

Lab 2: Arduino Analog and PWM Applications
Focuses on ADC reading with noise filtering, PWM motor speed control, servo positioning, and analog sensor interfacing.

Lab 3: Arduino Serial Communication Protocols
Focuses on UART, I2C, SPI master/slave implementations with sensors (MPU6050, BMP280, OLED displays).

Lab 4: ESP32 WiFi and MQTT Client
Focuses on WiFi station/AP modes, HTTP/HTTPS clients, MQTT publish/subscribe for IoT data pipelines.

Lab 5: ESP32 FreeRTOS Multitasking
Focuses on task creation/scheduling, inter-task communication, sensor data processing with WiFi transmission.

Lab 6: Jetson GPIO and Camera Interfacing
Focuses on GPIO control for robotics peripherals, CSI camera capture, OpenCV processing pipelines.

Lab 7: Jetson TensorRT Inference
Focuses on converting trained models to TensorRT, real-time inference benchmarking, GPU memory optimization.

Lab 8: FreeRTOS Device Driver Development
Focuses on implementing sensor drivers with proper error handling and timeout mechanisms.

Lab 9: micro-ROS Embedded Integration
Focuses on integrating micro-ROS with Arduino/ESP32 for ROS2 communication over serial/WiFi.

Lab 10: Complete Robotics Node Development
Focuses on building complete sensor-actuator-controller node integrating all platforms and communication protocols.

Major Specialisation IV

Software Testing & Quality Assurance (SE401)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Testing Fundamentals and Test Planning: Testing principles (early testing, defect clustering, pesticide paradox, context-dependent testing), Test process integration with SDLC, Test planning components (scope, strategy, resources, schedule, risks), Test design techniques classification (black-box, white-box, experience-based), IEEE 829 test documentation standards, Test maturity models and TMMi levels.

Unit 2
Black-Box Testing Techniques: Equivalence partitioning and boundary value analysis, Decision table testing and state transition testing, Use case testing and exploratory testing, Classification tree method, Pairwise testing and combinatorial test design, Domain testing and risk-based testing strategies, Test oracle problem and pairwise compatibility testing.

Unit 3
White-Box Testing and Code Coverage: Statement, branch/decision, and condition coverage metrics, Multiple condition coverage (MC/DC), Path testing and basis path testing, Loop testing strategies, Data flow testing (define-use paths), Mutation testing fundamentals and mutant operators, Code coverage tools and interpretation of coverage reports.

Unit 4
Test Automation and Frameworks: Test automation pyramid (unit, integration, UI), Automation frameworks (data-driven, keyword-driven, hybrid), Page Object Model and Screen Object Model, Selenium WebDriver architecture and locators, API testing with REST Assured/Postman, Performance testing basics (load, stress, endurance), Continuous testing in CI/CD pipelines.

Unit 5
Quality Assurance and Test Management: Software quality models (ISO 9126, ISO 25010), Quality gates and defect leakage metrics, Test management tools (JIRA, TestRail, Zephyr), Defect lifecycle and severity/priority classification, Test estimation techniques (3-point estimation, Work Breakdown Structure), Shift-left testing and DevOps test automation, Testability design principles and observability.

Practical

Lab 1: Test Case Design Workshop
Focuses on applying equivalence partitioning, boundary value analysis, and decision tables to create comprehensive test cases from requirements.

Lab 2: State Transition Testing
Focuses on modeling system behavior as state machines and deriving test cases from state transition diagrams.

Lab 3: Code Coverage Analysis
Focuses on instrumenting code with coverage tools, achieving target coverage levels, and interpreting coverage reports.

Lab 4: Mutation Testing Experiment
Focuses on generating mutants, measuring mutation score, and analyzing test suite effectiveness against mutants.

Lab 5: Selenium Web Automation
Focuses on Page Object Model implementation, cross-browser testing, and handling dynamic web elements.

Lab 6: API Testing Automation
Focuses on REST API testing with data-driven frameworks, authentication testing, and contract testing.

Lab 7: Performance Test Planning
Focuses on creating load test scenarios, ramp-up strategies, and analyzing performance bottlenecks.

Lab 8: Test Management Workflow
Focuses on defect lifecycle management, test execution tracking, and generating test metrics reports.

Lab 9: CI/CD Test Integration
Focuses on integrating automated tests into Jenkins/GitHub Actions pipelines with parallel execution.

Lab 10: Complete Test Automation Project
Focuses on end-to-end test automation suite development following test pyramid principles with quality gates.

Genetic Algorithms (ML401)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Foundations of Evolutionary Computation: Biological evolution principles and Darwinian natural selection, Genetic algorithms as population-based stochastic optimization, Search space representation and fitness landscapes, Schema theorem and building block hypothesis, Convergence properties and implicit parallelism, Comparison with gradient-based optimization methods, No Free Lunch theorem implications for EAs.

Unit 2
Genetic Algorithm Components and Operators: Encoding schemes (binary, real-valued, permutation, tree-based), Fitness function design and scaling (rank, linear, exponential), Selection methods (roulette wheel, tournament, rank, elitism), Crossover operators (single-point, uniform, arithmetic, blend), Mutation strategies (bit-flip, Gaussian, swap, scramble), Reproduction and generational replacement policies.

Unit 3
Advanced Genetic Algorithm Techniques: Adaptive parameter control (self-adaptive mutation rates, dynamic crossover probabilities), Niching methods (fitness sharing, crowding, island models), Multi-objective optimization (Pareto dominance, NSGA-II, SPEA2), Constraint handling (penalty functions, repair algorithms, decoder-based), Hybridization with local search (memetic algorithms), Parallel GA architectures.

Unit 4
Genetic Programming and Symbolic Regression: Tree-based representation for GP, Primitive sets and strongly-typed GP, Crossover and mutation for tree structures (subtree exchange, hoist), Bloat control mechanisms (Pareto front approximation, operator equal fitness), Automatic programming and symbolic regression, Koza's GP parameters and ramped half-and-half initialization, Multi-expression programming (MEP).

Unit 5
Real-World Applications and Engineering: Combinatorial optimization (traveling salesman problem, scheduling, knapsack), Function optimization (Rastrigin, Ackley, multimodal deceptive functions), Machine learning applications (feature selection, neural architecture search, hyperparameter optimization), Engineering design optimization (truss structures, aerodynamic shapes), Parameter tuning for deep learning and robotics controllers.

Practical

Lab 1: Basic GA Implementation
Focuses on implementing single-objective GA for function optimization, experimenting with encoding schemes and selection methods.

Lab 2: Selection and Crossover Analysis
Focuses on comparative study of selection operators and crossover types, measuring convergence speed and solution quality.

Lab 3: Fitness Landscape Visualization
Focuses on visualizing fitness landscapes, analyzing local/global optima, and deceptive problem characteristics.

Lab 4: Multi-objective NSGA-II Implementation
Focuses on Pareto front approximation, diversity preservation, and non-dominated sorting for engineering trade-offs.

Lab 5: Genetic Programming Basics
Focuses on tree-based GP for symbolic regression, analyzing expression trees and bloat control effectiveness.

Lab 6: TSP Optimization
Focuses on permutation encoding for traveling salesman problem, comparing GA with nearest neighbor heuristics.

Lab 7: Job Shop Scheduling
Focuses on constraint handling and schedule optimization using priority-based genetic representations.

Lab 8: Neural Network Hyperparameter Tuning
Focuses on using GA for architecture search and hyperparameter optimization of MLPs.

Lab 9: Memetic Algorithm Hybridization
Focuses on combining GA with gradient descent/local search for improved convergence.

Lab 10: Real-World Engineering Optimization
Focuses on complete GA/GP pipeline for structural/mechanical design optimization problems.

Big Data Technologies (DS401)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Big Data Fundamentals and Hadoop Ecosystem: 3V's characteristics (Volume, Velocity, Variety), CAP theorem and BASE consistency, Hadoop Distributed File System (HDFS) architecture (NameNode, DataNode, federation, high availability), HDFS data replication and fault tolerance, MapReduce programming model (mapper, reducer, combiner, partitioner), YARN resource management and application lifecycle.

Unit 2
Advanced Hadoop Components: Hive data warehousing (metastore, SerDe, partitioning, bucketing), HiveQL optimization and execution engine, Pig Latin scripting for ETL pipelines, HBase NoSQL database (column-family model, coprocessors), Sqoop data transfer between RDBMS and HDFS, Flume streaming data ingestion, Oozie workflow orchestration.

Unit 3
Apache Spark Core and RDDs: Spark architecture (driver, executors, cluster manager), Resilient Distributed Datasets (RDDs) lineage and fault tolerance, Spark transformations (map, flatMap, filter, join) and actions (collect, count, reduce), Spark SQL and DataFrames, Catalyst optimizer and Tungsten execution engine, Delta Lake for ACID transactions on data lakes.

Unit 4
Spark Streaming, MLlib, and GraphX: Structured Streaming (micro-batch vs. continuous processing), Kafka integration and exactly-once semantics, Spark MLlib pipelines (transformers, estimators, cross-validation), MLflow integration for experiment tracking, GraphX graph-parallel computation (Pregel API, PageRank, connected components), GraphFrames for DataFrame-based graph analytics.

Unit 5
Big Data Platforms and Cloud Integration: Cloud data platforms (AWS EMR, Azure HDInsight, Databricks), Managed Spark services and autoscaling, Data lake architectures (lakehouse pattern, medallion architecture), Apache Kafka fundamentals (topics, partitions, consumer groups, exactly-once), Apache Airflow DAG orchestration, Cost optimization strategies and spot instance utilization.

Practical

Lab 1: HDFS Operations and MapReduce
Focuses on HDFS shell commands, managing files/directories, implementing MapReduce jobs for word count and log analysis.

Lab 2: Hive Data Warehousing
Focuses on creating partitioned tables, HiveQL queries, UDF development, and analyzing query execution plans.

Lab 3: Spark RDD Programming
Focuses on RDD transformations/actions, custom partitioners, and broadcast variables for performance optimization.

Lab 4: Spark SQL and DataFrames
Focuses on DataFrame operations, Catalyst optimizer analysis, window functions, and Delta Lake transactions.

Lab 5: Spark Streaming with Kafka
Focuses on real-time data ingestion, stateful aggregations, and checkpointing for fault tolerance.

Lab 6: MLlib Machine Learning Pipelines
Focuses on end-to-end ML pipelines with feature engineering, hyperparameter tuning, and model persistence.

Lab 7: HBase NoSQL Operations
Focuses on column-family design, bulk loading, coprocessor implementation, and Phoenix SQL layer.

Lab 8: Data Pipeline Orchestration
Focuses on Apache Airflow DAGs integrating Spark, Hive, and HDFS for complete ETL workflows.

Lab 9: Cloud Big Data Services
Focuses on AWS EMR/S3 or Azure HDInsight deployments with autoscaling and cost monitoring.

Lab 10: Lakehouse Architecture Project
Focuses on complete data lake implementation with ingestion, transformation, ML training, and dashboarding.

Ethical Hacking & Penetration Testing (CY401)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Ethical Hacking Fundamentals and Reconnaissance: Cybersecurity kill chain and MITRE ATT&CK framework, Legal and ethical considerations (ROE, NDA, scope agreements), Passive reconnaissance (OSINT, WHOIS, DNS enumeration, Shodan), Active reconnaissance (port scanning, service versioning, banner grabbing), Nmap scripting engine (NSE), Vulnerability scanning (Nessus, OpenVAS), Footprinting techniques and information leakage prevention.

Unit 2
System Hacking and Privilege Escalation: Password cracking methodologies (dictionary, brute-force, rainbow tables), Hashcat and John the Ripper usage, Buffer overflow exploitation (stack/heap overflows), Privilege escalation techniques (Linux kernel exploits, Windows UAC bypass, SUID abuse), Password spraying and credential stuffing, Mimikatz for credential dumping, Lateral movement (Pass-the-Hash, Pass-the-Ticket).

Unit 3
Web Application Penetration Testing: OWASP Top 10 vulnerabilities, SQL injection (blind, time-based, error-based), Cross-Site Scripting (XSS - reflected, stored, DOM-based), Cross-Site Request Forgery (CSRF/XSRF), Server-Side Request Forgery (SSRF), Authentication bypass techniques, Session management flaws, Business logic vulnerabilities, Burp Suite Professional workflow (proxy, repeater, intruder, scanner).

Unit 4
Network Penetration Testing and Wireless: Network service exploitation (SMB, RDP, SSH weak credentials), Metasploit Framework (MSF) modules and payloads, Exploit development lifecycle, Post-exploitation modules (meterpreter, persistence, pivoting), Wireless attacks (WEP/WPA2 cracking, evil twin AP, KRACK), Man-in-the-Middle attacks (ARP poisoning, SSL stripping), Network device exploitation (router/switch configuration errors).

Unit 5
Advanced Pentesting and Reporting: Active Directory attacks (Kerberoasting, AS-REP roasting, Golden/Silver tickets), Container and Kubernetes security testing, Cloud platform pentesting (IAM misconfigurations, S3 bucket enumeration), Red teaming methodologies and OPSEC, Penetration test reporting (executive summary, technical findings, risk ratings, remediation), Continuous security testing (DAST, SAST, IAST), Defensive strategies and blue teaming.

Practical

Lab 1: Reconnaissance and Scanning
Focuses on passive/active reconnaissance, Nmap scripting, vulnerability scanning with OpenVAS/Nessus.

Lab 2: Password Cracking and Hash Analysis
Focuses on cracking various hash types with Hashcat/John, creating custom wordlists and rules.

Lab 3: Buffer Overflow Exploitation
Focuses on stack/heap overflow exploitation, shellcode development, and ASLR bypass techniques.

Lab 4: Metasploit Framework Mastery
Focuses on MSF modules, payload customization, post-exploitation, and pivoting scenarios.

Lab 5: SQL Injection Exploitation
Focuses on discovering and exploiting SQLi vulnerabilities, database enumeration, privilege escalation.

Lab 6: XSS and CSRF Attacks
Focuses on reflected/stored XSS payloads, CSRF token bypass, DOM-based XSS exploitation.

Lab 7: Burp Suite Web Pentesting
Focuses on proxy interception, repeater manipulation, intruder fuzzing, and Burp extensions.

Lab 8: Wireless and Network Attacks
Focuses on WPA2 cracking, ARP poisoning, SSL stripping, and network service exploitation.

Lab 9: Active Directory Domain Pentesting
Focuses on Kerberoasting, DCSync attacks, and lateral movement in Windows environments.

Lab 10: Complete Pentest Report
Focuses on red team engagement simulation with executive/technical reporting and remediation roadmap.

Robot Operating System (ROS) (RB401)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
ROS Fundamentals and Communication: ROS design principles (modularity, distributed systems, hardware abstraction), ROS master (roscore) and graph concepts, Node architecture and lifecycle, ROS topics (publish/subscribe model, QoS policies), ROS services (request/response), ROS parameters (global configuration), ROS messages and serialization (msg/srv/action definitions), ROS command line tools (rostopic, rosservice, rosparam).

Unit 2
ROS Packages, Workspaces, and Tools: Catkin build system and workspace structure (src, devel, build), Package manifest (package.xml), CMakeLists.txt configuration, ROS launch files (parameter substitution, remapping, namespaces), RViz visualization (displays, panels, plugins), rqt tools suite (rqt_graph, rqt_console, rqt_plot), Gazebo simulator integration and URDF modeling.

Unit 3
TF and Robot Kinematics: TF (Transform) library and frame transforms, Static/dynamic broadcasters and listeners, TF trees and visualization, Forward kinematics using KDL, Denavit-Hartenberg parameters, Inverse kinematics solvers, ROS Control framework (hardware_interface, controller_manager), Joint trajectory controllers.

Unit 4
ROS Navigation Stack: ROS navigation architecture (move_base, costmaps, global/local planners), AMCL (Adaptive Monte Carlo Localization), SLAM (gmapping, cartographer, hdl_graph_slam), Path planning algorithms (A*, DWA, TEB), Recovery behaviors and costmap configuration, Nav2 stack improvements (Behavior Trees, SMAC planner), Map server and lifecycle management.

Unit 5
ROS 2, Perception, and Advanced Topics: ROS 2 DDS middleware (CycloneDDS, Fast-RTPS), ROS 2 nodes and executables (colcon build system), Composition and launch improvements, ROS-Industrial (URDF for manufacturing), Perception pipelines (PCL point clouds, OpenCV integration), MoveIt motion planning (OMPL planners, collision checking), ROS realtime extensions and micro-ROS for embedded systems.

Practical

Lab 1: ROS Workspace and Basic Nodes
Focuses on creating catkin workspaces, publisher/subscriber nodes, launch files, and parameter servers.

Lab 2: Custom Message Types and Services
Focuses on defining msg/srv/action files, implementing service clients/servers, and message serialization.

Lab 3: TF Transform Broadcasting
Focuses on static/dynamic TF broadcasters, frame hierarchies, and RViz visualization of transform trees.

Lab 4: URDF Robot Modeling
Focuses on creating URDF models with links/joints, meshes, sensors, and Gazebo simulation plugins.

Lab 5: ROS Control Implementation
Focuses on hardware interfaces, joint controllers, and gazebo_ros plugins for simulated robot control.

Lab 6: Navigation Stack Configuration
Focuses on map creation, AMCL localization, path planning, and obstacle avoidance tuning.

Lab 7: SLAM Implementation
Focuses on gmapping/cartographer SLAM with real/simulated robot data, loop closure analysis.

Lab 8: MoveIt Motion Planning
Focuses on MoveIt setup assistant, planning scenes, collision objects, and pick/place pipelines.

Lab 9: Perception Pipeline Development
Focuses on point cloud processing, object detection, and camera-LiDAR fusion.

Lab 10: Complete ROS Robotics Application
Focuses on autonomous navigation project integrating localization, mapping, planning, and manipulation.

Semester VI

Software Engineering (CS303)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Software Engineering Foundations and Process Models: Software crisis and characteristics of software, Software development life cycle (SDLC) models (waterfall, prototyping, iterative, spiral), Agile manifesto and principles, Scrum framework (roles, ceremonies, artifacts), Kanban and lean software development, Capability Maturity Model Integration (CMMI) levels, Process selection criteria based on project characteristics and risk profiles.

Unit 2
Requirements Engineering and Analysis: Requirements elicitation techniques (interviews, surveys, observation, JAD sessions), Functional vs. non-functional requirements, Use case modeling (actors, scenarios, relationships), User stories and acceptance criteria, Requirements traceability matrix, Requirements validation (reviews, prototyping), Change management and version control of requirements specifications.

Unit 3
Software Design Principles and UML: Design concepts (modularity, cohesion, coupling, abstraction), Architectural patterns (MVC, layered, microservices), UML 2.x diagrams (class, sequence, activity, state, component, deployment), Design principles (SOLID, DRY, KISS), Design patterns (creational, structural, behavioral - focusing on Strategy, Observer, Factory), Refactoring techniques and code smells.

Unit 4
Software Testing Fundamentals and Strategies: Testing principles (early testing, defect clustering, pesticide paradox), Test case design techniques (equivalence partitioning, boundary value analysis, decision table testing), Levels of testing (unit, integration, system, acceptance), Test-driven development (TDD) and behavior-driven development (BDD), Test automation frameworks and tools, Mutation testing concepts.

Unit 5
Software Project Management and Quality: Effort estimation techniques (COCOMO, function point analysis, planning poker), Project scheduling (Gantt charts, PERT/CPM), Risk management (identification, analysis, mitigation), Software metrics (size, complexity, quality metrics), Software configuration management (version control, branching strategies), Software quality assurance (SQA) and process audits, DevOps practices and CI/CD pipelines.

Database Management Systems (CS304)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Database Systems and ER Modeling: Database management systems vs. file systems, ANSI/SPARC architecture (external, conceptual, internal levels), Data independence (logical/physical), Entity-Relationship model (entities, relationships, attributes, keys), ER diagram notation and cardinality constraints, Enhanced ER (EER) features (specialization, generalization, aggregation), Converting ER diagrams to relational schemas.

Unit 2
Relational Model and SQL: Relational data model (domains, relations, tuples), Codd's relational algebra operations (selection, projection, union, set difference, Cartesian product, joins), Relational calculus (tuple, domain), SQL fundamentals (DDL, DML, DCL, TCL), Nested queries and correlated subqueries, Aggregate functions, GROUP BY, HAVING, EXISTS/IN operators, Views and virtual tables.

Unit 3
Transaction Management and Concurrency: ACID properties, Transaction states and implementation, Serializability (conflict serializability, view serializability), Concurrency control protocols (two-phase locking, timestamp ordering), Deadlock prevention, detection, recovery, Granularity of locking (database, table, row level), Isolation levels (Read Uncommitted, Read Committed, Repeatable Read, Serializable).

Unit 4
File Organization and Indexing: Physical file organization (heap, sequential, hashed), RAID levels and redundancy, Index structures (primary, secondary, clustered, non-clustered), B+ tree index organization and search algorithms, Dynamic hashing (extendible, linear), Query processing (cost estimation, equivalence rules), Join algorithms (nested loop, block nested loop, sort-merge, hash join).

Unit 5
Query Optimization and Advanced Topics: Query execution plans and optimization strategies, Heuristic optimization rules, Cost-based optimization, Distributed databases (fragmentation, replication, transparency), Query decomposition and distributed query processing, Object-relational databases, XML/NoSQL integration concepts, Database recovery techniques (log-based, checkpointing, shadow paging).

Database Management Systems Lab (CX304)

← Back to Scheme

Credits: 2 (0-0-2)

Course Content

Lab 1: DDL and Basic DML Operations
Focuses on creating databases, tables with constraints (PRIMARY KEY, FOREIGN KEY, CHECK, UNIQUE), inserting, updating, and deleting records with basic SELECT queries.

Lab 2: Advanced SQL Queries
Focuses on nested queries, multiple JOIN types (INNER, LEFT, RIGHT, FULL OUTER), aggregate functions with GROUP BY/HAVING clauses.

Lab 3: Subqueries and Set Operations
Focuses on correlated subqueries, EXISTS/IN/NOT IN operators, UNION/INTERSECT/MINUS operations, and view creation/manipulation.

Lab 4: Stored Procedures and Functions
Focuses on creating PL/SQL procedures, functions, triggers, and packages for business logic encapsulation.

Lab 5: Transaction Control and Concurrency
Focuses on implementing ACID transactions, SAVEPOINT usage, demonstrating lost update and dirty read anomalies.

Lab 6: Isolation Levels Testing
Focuses on configuring different transaction isolation levels and observing phantom reads, non-repeatable reads behavior.

Lab 7: Index Analysis and Performance
Focuses on creating various index types (B-tree, bitmap), analyzing execution plans, measuring query performance improvements.

Lab 8: Query Optimization Study
Focuses on rewriting queries for better performance, using EXPLAIN PLAN, creating materialized views.

Lab 9: Database Backup and Recovery
Focuses on full/incremental backups, point-in-time recovery, log analysis, and disaster recovery scenarios.

Lab 10: Complete Application Database Design
Focuses on end-to-end database design (ER diagram → normalized schema → complex queries → stored procedures → performance tuning).

Major Specialisation V

Distributed Systems & Microservices (SE501)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Distributed Systems Fundamentals: Characteristics of distributed systems (transparency, openness, scalability), System models (client-server, peer-to-peer, multi-tier), Fallacies of distributed computing (clock synchronization, network failures), Time synchronization (NTP, logical clocks, vector clocks), Distributed mutual exclusion algorithms (Ricart-Agrawala, token ring), CAP theorem and consistency models (strong, eventual, causal).

Unit 2
Communication and Coordination: Message passing vs. shared memory, RPC and RMI principles, REST vs. gRPC communication models, Asynchronous messaging (RabbitMQ, Apache Kafka), Publish-subscribe patterns, Leader election algorithms (Bully, Raft simplified), Group membership and failure detectors, Distributed transactions (2PC, 3PC, Saga pattern).

Unit 3
Replication and Consistency: Replication techniques (primary-backup, multi-master, chain), Consistency models (linearizability, sequential consistency, PRAM), Quorum systems and consistency guarantees, Conflict resolution strategies (last-writer-wins, vector clocks, CRDTs), Paxos consensus algorithm overview, Raft consensus protocol (leader election, log replication), Practical replication systems (ZooKeeper, etcd).

Unit 4
Microservices Architecture Principles: Monolith to microservices decomposition strategies, Domain-Driven Design (bounded contexts, context mapping), Service granularity and cohesion principles, API gateway patterns, Service mesh architecture (Istio, Linkerd), Circuit breaker pattern, Bulkhead pattern, Saga pattern for distributed transactions, Event sourcing and CQRS.

Unit 5
Distributed System Resilience and Observability: Fault tolerance design patterns (retry, timeout, fallback), Chaos engineering principles, Distributed tracing (OpenTelemetry, Jaeger), Metrics collection and alerting (Prometheus, Grafana), Service level objectives (SLOs, SLIs, error budgets), Load balancing strategies (client-side, server-side), Autoscaling and capacity planning, Deployment strategies (blue-green, canary, rolling updates).

Natural Language Processing (NLP) (ML501)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
NLP Fundamentals and Text Preprocessing: Language modeling and n-gram models, Regular expressions for tokenization, Sentence segmentation and normalization, Stemming, lemmatization, and part-of-speech tagging, Stopword removal and text normalization, Bag-of-words and TF-IDF representations, Character-level and subword tokenization (BPE, WordPiece), Unicode handling and multilingual text processing.

Unit 2
Word Embeddings and Sequence Models: Word2Vec (skip-gram, CBOW), GloVe and fastText embeddings, Contextual embeddings (ELMo, Flair), RNN/LSTM/GRU for sequence modeling, Bidirectional encoders, Sequence labeling tasks (NER, POS tagging, chunking), CRF layer for structured prediction, Attention mechanisms (self-attention, multi-head attention).

Unit 3
Transformer Architecture and BERT: Transformer model architecture (encoder-decoder, positional encoding), Self-attention and scaled dot-product attention, Multi-head attention and layer normalization, BERT pretraining objectives (MLM, NSP), RoBERTa, DistilBERT, and ALBERT variants, Fine-tuning strategies and domain adaptation, Sentence-BERT for semantic similarity.

Unit 4
Large Language Models and Prompt Engineering: GPT architecture evolution (GPT-1 to GPT-4), Decoder-only transformers, In-context learning and few-shot prompting, Chain-of-thought reasoning, Retrieval-Augmented Generation (RAG) architecture, Knowledge graphs and dense retrieval, Prompt engineering techniques (zero-shot, few-shot, instruction tuning), Hallucination mitigation strategies.

Unit 5
Advanced NLP Systems and Evaluation: Text generation evaluation (BLEU, ROUGE, BERTScore, human evaluation), Question answering systems (extractive, generative), Conversational AI (dialogue state tracking, response generation), Multilingual NLP (mBERT, XLM-R), Model deployment (TGI, vLLM), Ethical considerations (bias detection, toxicity classification, fairness evaluation), RAG evaluation frameworks.

Practical

Lab 1: Text Preprocessing Pipeline
Focuses on implementing complete text preprocessing with tokenization, normalization, and vectorization techniques.

Lab 2: Word Embeddings and Similarity
Focuses on training Word2Vec/GloVe, analogy tasks, and semantic similarity measurement.

Lab 3: Sequence Models for NER/POS
Focuses on BiLSTM+CRF implementation for named entity recognition and POS tagging.

Lab 4: BERT Fine-tuning
Focuses on fine-tuning BERT/RoBERTa for classification, NER, and question answering tasks.

Lab 5: Transformer from Scratch
Focuses on implementing multi-head self-attention and basic transformer encoder.

Lab 6: LLM Prompt Engineering
Focuses on zero-shot/few-shot prompting, chain-of-thought, and comparative evaluation.

Lab 7: RAG Implementation
Focuses on building complete RAG pipeline with dense retrieval and generation.

Lab 8: Conversational AI Chatbot
Focuses on dialogue systems with context tracking and response generation.

Lab 9: Multilingual NLP Pipeline
Focuses on cross-lingual transfer learning and zero-shot classification.

Lab 10: Production NLP Application
Focuses on end-to-end NLP system deployment with evaluation and monitoring.

Predictive Modeling & Analytics (DS501)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Predictive Modeling Foundations: Supervised vs. unsupervised learning review, Regression vs. classification frameworks, Model evaluation metrics (MAE, RMSE, R^2 for regression; precision, recall, F1, AUC for classification), Cross-validation strategies (k-fold, stratified, time-series), Bias-variance tradeoff revisited, Overfitting detection (learning curves, validation monitoring), Model selection criteria (AIC, BIC, cross-validated scores).

Unit 2
Ensemble Methods and Bagging: Bagging fundamentals and variance reduction, Random Forests (feature bagging, bootstrap aggregating), Extra Trees and feature importance interpretation, Out-of-bag error estimation, Gradient Boosting Machines (GBM) - forward stage-wise additive modeling, XGBoost architecture (regularized objectives, second-order gradients), LightGBM and CatBoost optimizations.

Unit 3
Time Series Forecasting: Time series components (trend, seasonality, cyclicity, irregularity), Stationarity testing (ADF, KPSS tests), ARIMA/SARIMA models (ACF, PACF analysis, differencing), Exponential smoothing methods (Holt-Winters), Prophet forecasting framework, LSTM/GRU for sequential forecasting, Cross-validation for time series (purged/time-series splits), Anomaly detection in temporal data.

Unit 4
Survival Analysis and Imbalanced Learning: Survival function, hazard rates, Kaplan-Meier estimator, Cox proportional hazards model, Time-to-event prediction metrics (C-index, Brier score), Imbalanced classification techniques (SMOTE, ADASYN, undersampling), Cost-sensitive learning, Anomaly detection algorithms (isolation forest, one-class SVM, autoencoders), Class imbalance evaluation (PR-AUC, Matthews correlation coefficient).

Unit 5
Model Deployment, Monitoring, and MLOps: Model serialization (joblib, ONNX), Containerization (Docker for ML models), API serving (Flask, FastAPI, TensorFlow Serving), Model monitoring (drift detection, performance degradation), A/B testing and champion/challenger patterns, Automated retraining pipelines, Explainable AI techniques (SHAP, LIME, partial dependence), Production ML challenges (data versioning, lineage tracking).

Practical

Lab 1: Model Evaluation and Validation
Focuses on implementing comprehensive cross-validation strategies and model selection pipelines.

Lab 2: Ensemble Learning Implementation
Focuses on Random Forest and XGBoost training, hyperparameter tuning, and feature importance analysis.

Lab 3: ARIMA and Exponential Smoothing
Focuses on time series decomposition, stationarity testing, and classical forecasting models.

Lab 4: Deep Learning for Time Series
Focuses on LSTM/GRU forecasting with attention mechanisms and multivariate inputs.

Lab 5: Survival Analysis Implementation
Focuses on Kaplan-Meier curves, Cox models, and time-to-event prediction evaluation.

Lab 6: Imbalanced Classification Techniques
Focuses on SMOTE oversampling, cost-sensitive learning, and PR-AUC optimization.

Lab 7: Anomaly Detection Systems
Focuses on isolation forest, autoencoders, and multivariate anomaly detection pipelines.

Lab 8: Model Deployment APIs
Focuses on containerized model serving with FastAPI and performance benchmarking.

Lab 9: MLOps Pipeline Development
Focuses on automated retraining, drift detection, and A/B testing implementation.

Lab 10: Production Analytics Project
Focuses on complete predictive analytics solution from data ingestion to stakeholder dashboards.

Advanced Cryptography (CY501)

← Back to Scheme

Credits: 3 (3-0-0)

Course Content

Unit 1
Block Cipher Cryptanalysis: Differential cryptanalysis principles and attack complexity, Linear cryptanalysis and correlation matrices, Integral cryptanalysis and multidimensional characteristics, Truncated differential attacks, Impossible differential cryptanalysis, Boomerang and rectangle attacks, Slide attacks and self-similarity properties, Key schedule cryptanalysis techniques.

Unit 2
Stream Ciphers and Sequence Generators: Linear Feedback Shift Registers (LFSRs) and Berlekamp-Massey algorithm, Nonlinear combination generators and correlation immunity, Stream cipher design criteria (period, linear complexity, balance), RC4 analysis (key scheduling biases, WEP attacks), Grain and Trivium stream ciphers, Related-key and distinguishing attacks on stream ciphers.

Unit 3
Public-Key Cryptosystems and Attacks: RSA cryptanalysis (factoring algorithms - trial division, Fermat, quadratic sieve, number field sieve), Rabin cryptosystem and square roots modulo composite, Paillier homomorphic encryption, Elliptic Curve Discrete Logarithm Problem (ECDLP), Index calculus attack on elliptic curves, Pairing-based cryptography (bilinear maps, identity-based encryption).

Unit 4
Hash Function Design and Cryptanalysis: Merkle-Damgård construction and length extension attacks, Sponge construction (Keccak/SHA-3), Hash function security properties (collision, preimage, second preimage resistance), Differential paths in hash functions, Boomerang distinguishers for compression functions, Multicollision attacks and Joux's attack, Hash-based signatures (Lamport, XMSS).

Unit 5
Post-Quantum Cryptography and Advanced Topics: Quantum computing threats (Shor's algorithm, Grover's algorithm), Lattice-based cryptography (Learning With Errors - LWE, Ring-LWE), Code-based cryptography (McEliece cryptosystem), Hash-based signatures (SPHINCS+, LMS), Isogeny-based cryptography, Multivariate polynomial cryptography, NIST PQC standardization process and migration strategies.

Communication & IoT (RB501)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
IoT Communication Fundamentals: IoT reference model (devices, gateway, edge, cloud), Wireless communication principles (modulation, spectrum, bandwidth), Short-range protocols (Bluetooth LE, Zigbee, Z-Wave), LPWAN technologies (LoRaWAN, NB-IoT, LTE-M, Sigfox), Frequency bands and regulatory domains, Power consumption models and duty cycle constraints, Network topologies (star, mesh, tree).

Unit 2
IoT Protocols and Messaging: MQTT protocol (QoS levels, retained messages, last will), CoAP for constrained devices (DTLS security, observe pattern), HTTP/2 and WebSockets for IoT, AMQP and STOMP messaging, DDS for robotics real-time communication, Protocol benchmarking (overhead, latency, throughput), Header compression techniques (6LoWPAN).

Unit 3
IoT Security and Device Management: Device authentication (X.509 certificates, pre-shared keys, token-based), Secure communication (TLS/DTLS, mbedTLS), Firmware Over-The-Air (OTA) updates, Device lifecycle management (provisioning, decommissioning), Secure boot and attestation, Key management in constrained environments, Zero-touch provisioning (ZTP).

Unit 4
Edge Computing and Gateway Architectures: Edge vs. cloud computing trade-offs, Gateway patterns (protocol translation, data aggregation, local processing), EdgeX Foundry and Eclipse IoT frameworks, Time-Sensitive Networking (TSN) for industrial IoT, Containerized edge computing (K3s, microk8s), Data preprocessing at edge (filtering, transformation, analytics).

Unit 5
IoT Platforms and Robotics Integration: AWS IoT Core (Greengrass, Device Shadow), Azure IoT Hub (Device Twin, DPS), Google Cloud IoT (Cloud IoT Edge), ROS2 DDS integration with IoT protocols, Multi-protocol gateways for robotics fleets, Digital twins and simulation-in-the-loop, Industrial IoT protocols (OPC UA, Modbus TCP, PROFINET).

Practical

Lab 1: Bluetooth LE and Zigbee Interfacing
Focuses on BLE GATT services, Zigbee mesh networking, and multi-protocol sensor node development.

Lab 2: LoRaWAN Network Deployment
Focuses on LoRa gateway setup, end-device registration, and long-range sensor data transmission.

Lab 3: MQTT/CoAP Client Implementation
Focuses on QoS testing, secure connections, and protocol interoperability benchmarking.

Lab 4: IoT Security Implementation
Focuses on certificate-based authentication, secure OTA updates, and device attestation.

Lab 5: Edge Gateway Development
Focuses on protocol translation, local data processing, and cloud-edge synchronization.

Lab 6: ROS2-IoT Integration
Focuses on DDS bridge to MQTT, multi-robot fleet communication, and cloud data streaming.

Lab 7: AWS/Azure IoT Platform Integration
Focuses on device shadows, rule engine configuration, and serverless IoT backends.

Lab 8: Industrial Protocol Implementation
Focuses on OPC UA client/server, Modbus gateways, and PROFINET device simulation.

Lab 9: Digital Twin Development
Focuses on ROS-Gazebo simulation synchronization with physical IoT devices.

Lab 10: Complete IoT Robotics Solution
Focuses on multi-device IoT robotics deployment with edge-cloud orchestration.

Major Specialisation VI

Building Scalable Cloud Services (SE601)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Scalability Patterns and Load Balancing: Scalability types (vertical, horizontal, diagonal), Stateless vs. stateful services, Load balancing algorithms (round-robin, least connections, IP hash), Layer 4 vs. Layer 7 load balancing, Health checks and graceful degradation, Auto-scaling policies (CPU/memory utilization, custom metrics, predictive scaling), Capacity planning and right-sizing.

Unit 2
Container Orchestration and Service Mesh: Docker containerization best practices, Kubernetes architecture (control plane, worker nodes, etcd), Pod lifecycle and controllers (Deployment, StatefulSet, DaemonSet), Kubernetes Services (ClusterIP, NodePort, LoadBalancer, ExternalName), Ingress controllers and API gateways, Service mesh patterns (Istio virtual services, traffic management, observability).

Unit 3
Event-Driven and Asynchronous Architectures: Message brokers (Kafka, RabbitMQ, SQS), Pub-sub patterns and fan-out/fan-in, Event sourcing and CQRS principles, Apache Kafka (topics, partitions, consumer groups, exactly-once semantics), Stream processing (Kafka Streams, KSQL), Serverless event sources (Lambda triggers, Cloud Functions events), Saga pattern for distributed transactions.

Unit 4
Distributed Caching and Data Stores: Caching patterns (cache-aside, write-through, read-through), Redis cluster architecture and data sharding, Memcached consistent hashing, CDN integration for static assets, NoSQL selection criteria (DynamoDB, Cassandra, MongoDB Atlas), Relational database scaling (read replicas, sharding), Multi-region database replication.

Unit 5
Observability, Chaos Engineering, and Cost Optimization: The three pillars (metrics, logs, traces), Prometheus/Grafana monitoring stacks, OpenTelemetry instrumentation, Distributed tracing analysis, Chaos engineering (Chaos Monkey, Gremlin), SLO/SLI/SLA definition and error budgets, FinOps practices, Reserved/spot instances, Cost allocation tagging, Multi-cloud cost comparison.

Practical

Lab 1: Load Balancer and Auto Scaling
Focuses on configuring ALB/NLB, auto scaling groups, and blue-green deployments.

Lab 2: Kubernetes Cluster Deployment
Focuses on minikube setup, deployments, services, ingress, and Helm chart packaging.

Lab 3: Kafka Event Streaming Pipeline
Focuses on multi-partition topics, consumer groups, stream processing applications.

Lab 4: Service Mesh Traffic Management
Focuses on Istio installation, virtual services, canary deployments, circuit breakers.

Lab 5: Distributed Caching Patterns
Focuses on Redis cluster setup, cache invalidation strategies, CDN integration.

Lab 6: Microservices Communication
Focuses on gRPC services, API gateway routing, service discovery patterns.

Lab 7: Monitoring and Alerting Stack
Focuses on Prometheus/Grafana dashboards, alerting rules, SLO calculation.

Lab 8: Chaos Engineering Experiments
Focuses on Chaos Mesh experiments, fault injection, resilience testing.

Lab 9: Cost Optimization Analysis
Focuses on cost explorer analysis, rightsizing recommendations, FinOps workflows.

Lab 10: Production Microservices Platform
Focuses on complete scalable cloud service deployment with full observability.

Computer Vision (ML601)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Digital Image Fundamentals and Processing: Image formation and digitization (sampling, quantization), Pixel representations and color spaces (RGB, HSV, Lab, grayscale), Spatial domain filtering (linear, nonlinear filters, smoothing, sharpening), Edge detection (Sobel, Prewitt, Canny, Laplacian of Gaussian), Histogram equalization and contrast enhancement, Morphological operations (erosion, dilation, opening, closing).

Unit 2
Frequency Domain Processing and Transforms: 2D Discrete Fourier Transform (DFT) and properties, Fast Fourier Transform (FFT) implementation, High-pass/low-pass filtering in frequency domain, Homomorphic filtering for illumination correction, Discrete Cosine Transform (DCT) for compression, Wavelet transforms (Haar, Daubechies), Multi-resolution analysis and pyramid representations.

Unit 3
Feature Detection, Extraction, and Description: Corner detection (Harris, Shi-Tomasi), Blob detection (LoG, DoG, Hessian), SIFT (scale-space extrema, keypoint description), SURF and ORB for real-time applications, HOG (Histogram of Oriented Gradients) for pedestrian detection, Feature matching (nearest neighbor, FLANN), RANSAC for robust estimation, Bag-of-visual-words model.

Unit 4
Camera Geometry and 3D Vision: Pinhole camera model and intrinsic/extrinsic parameters, Camera calibration (Zhang's method, checkerboard patterns), Epipolar geometry and fundamental matrix, Stereo vision (disparity maps, rectification), Structure from Motion (SfM) pipeline, SLAM fundamentals (visual odometry, bundle adjustment), Depth estimation from monocular cues.

Unit 5
Deep Learning for Computer Vision: CNN architectures for vision (AlexNet, VGG, ResNet, EfficientNet), Object detection (R-CNN family, YOLO, SSD, RetinaNet), Semantic segmentation (FCN, U-Net, DeepLab), Instance segmentation (Mask R-CNN), Visual transformers (ViT, Swin Transformer), Self-supervised learning (SimCLR, DINO), 3D vision with PointNet/PointNet++, Video analysis (optical flow, 3D CNNs).

Practical

Lab 1: Image Processing Pipeline
Focuses on spatial/frequency domain filtering, histogram equalization, and morphological operations.

Lab 2: Edge Detection and Feature Extraction
Focuses on Canny edge detection, HOG descriptors, and contour analysis.

Lab 3: SIFT/SURF Feature Matching
Focuses on keypoint detection, descriptor extraction, and robust matching with RANSAC.

Lab 4: Camera Calibration
Focuses on checkerboard calibration, intrinsic/extrinsic parameter estimation, and distortion correction.

Lab 5: Stereo Vision and Depth Maps
Focuses on stereo rectification, disparity computation, and 3D point cloud generation.

Lab 6: Object Detection with YOLO
Focuses on YOLO model training, custom dataset annotation, and real-time inference.

Lab 7: Semantic Segmentation
Focuses on U-Net/DeepLab implementation for pixel-wise classification.

Lab 8: Visual SLAM Implementation
Focuses on ORB-SLAM or DSO for real-time visual odometry and mapping.

Lab 9: 3D Vision with Point Clouds
Focuses on PointNet for 3D object classification and segmentation.

Lab 10: Complete CV Robotics Application
Focuses on end-to-end vision system integrating detection, tracking, and 3D perception.

Time Series Analysis & Forecasting (DS601)

← Back to Scheme

Credits: 3 (2-1-0)

Theory

Unit 1
Time Series Fundamentals and Decomposition: Time series components (trend, seasonality, cycle, irregular), Stationarity concepts (weak vs. strict stationarity), Trend estimation (moving averages, polynomial fitting, LOESS), Seasonal decomposition (classical, STL - Seasonal-Trend decomposition using Loess), Autocorrelation (ACF) and partial autocorrelation (PACF) analysis, Differencing and transformations for stationarity.

Unit 2
Classical Time Series Models: ARIMA models (p,d,q parameters, stationarity/invertibility conditions), Box-Jenkins methodology, Model identification (ACF/PACF interpretation), Parameter estimation (MLE, least squares), Model diagnostics (Ljung-Box test, residual analysis), SARIMA for seasonal data, Seasonal differencing and Fourier terms.

Unit 3
Exponential Smoothing and State Space Models: Simple exponential smoothing, Holt's linear trend method, Holt-Winters seasonal method, ETS framework (error, trend, seasonal components), Optimal smoothing parameters (MLE optimization), State space representation, Kalman filter for parameter estimation, Dynamic linear models and interventions.

Unit 4
Modern Machine Learning for Time Series: Feature engineering (lagged variables, rolling statistics, Fourier features), Tree-based methods (XGBoost/LightGBM with time series splits), LSTM/GRU networks for sequential forecasting, Transformer models (Temporal Fusion Transformer, Informer), Ensemble methods (statistical + ML hybrid models), Cross-validation strategies (purged, embargoed, walk-forward).

Unit 5
Advanced Topics and Production Systems: Multivariate time series (VAR, VEC models), Hierarchical forecasting (bottom-up, top-down, optimal reconciliation), Anomaly detection (isolation forest, autoencoders, statistical process control), Probabilistic forecasting (quantile regression, conformal prediction), Automated ML (AutoTS, sktime, Darts), Production deployment (MLOps for time series, concept drift detection).

Practical

Lab 1: Time Series Exploration
Focuses on decomposition, stationarity testing, ACF/PACF analysis, and trend/seasonal extraction.

Lab 2: ARIMA Model Development
Focuses on auto.arima, manual parameter selection, model diagnostics, and forecasting evaluation.

Lab 3: Exponential Smoothing Methods
Focuses on ETS model selection, Holt-Winters implementation, and smoothing parameter optimization.

Lab 4: ML for Univariate Forecasting
Focuses on feature engineering, XGBoost/LightGBM with proper time series validation.

Lab 5: LSTM/GRU Sequence Models
Focuses on multivariate LSTM networks, attention mechanisms, and early stopping.

Lab 6: Hierarchical Forecasting
Focuses on bottom-up/top-down methods and reconciliation techniques.

Lab 7: Anomaly Detection in Time Series
Focuses on statistical methods, isolation forest, and reconstruction-based approaches.

Lab 8: Probabilistic Forecasting
Focuses on quantile regression and conformal prediction intervals.

Lab 9: Automated Forecasting Pipeline
Focuses on AutoTS/Darts implementation with model selection and ensemble.

Lab 10: Production Time Series System
Focuses on complete pipeline from data ingestion through deployment and monitoring.

Web & Application Security (CY601)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Web Security Fundamentals and Threat Modeling: OWASP Top 10 methodology and risk rating, Threat modeling (STRIDE, DREAD, PASTA), Authentication flaws (weak passwords, credential stuffing, broken session management), Authorization bypass techniques (IDOR, missing access control), Security misconfiguration patterns, Sensitive data exposure risks, HTTP security headers (CSP, HSTS, X-Frame-Options).

Unit 2
Injection Attacks and Secure Input Handling: SQL injection attack vectors (in-band, blind, time-based, error-based), Prepared statements and parameterized queries, ORM injection prevention, NoSQL injection (MongoDB, CouchDB), Command injection and OS command execution, LDAP injection, XML External Entity (XXE) attacks, Input validation and sanitization strategies, Web Application Firewall (WAF) rules.

Unit 3
Cross-Site Scripting and CSRF Protection: XSS attack types (reflected, stored/persistent, DOM-based), XSS payloads and filtering evasion techniques, Content Security Policy (CSP) implementation, HttpOnly/Secure cookies, CSRF attack mechanics and token-based mitigation, SameSite cookie attribute, Double-submit cookie pattern, CSRF Guard frameworks and synchronizer token pattern.

Unit 4
Business Logic and API Security: Business logic flaws (race conditions, logic bypass, workflow manipulation), API security (broken object level authorization - BOLA, excessive data exposure), Rate limiting and throttling strategies, JWT security (alg confusion, none algorithm, kid header injection), OAuth 2.0 pitfalls (redirect URI manipulation, insufficient scope validation), GraphQL security (deep recursion, batching attacks).

Unit 5
Application Security Testing and Secure SDLC: Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), Interactive Application Security Testing (IAST), Software Composition Analysis (SCA), Security requirements in Agile, Threat modeling in sprints, Secure code review checklist, Container security (Dockerfile best practices, image scanning), CI/CD security (secrets management, pipeline scanning), RASP and runtime protection.

Practical

Lab 1: Web Vulnerability Scanning
Focuses on OWASP ZAP/Burp Suite scanning, identifying common misconfigurations and security headers.

Lab 2: SQL Injection Exploitation
Focuses on discovering SQLi vectors, blind/time-based exploitation, bypassing WAF filters.

Lab 3: XSS Attack and Defense
Focuses on reflected/stored/DOM XSS payloads, CSP bypass techniques, secure output encoding.

Lab 4: CSRF Exploitation and Mitigation
Focuses on CSRF token analysis, SameSite cookie testing, double-submit cookie implementation.

Lab 5: Authentication Bypass Attacks
Focuses on session fixation, weak password policies, OAuth misconfigurations.

Lab 6: API Security Testing
Focuses on BOLA, broken authentication, rate limiting bypass, GraphQL introspection attacks.

Lab 7: Business Logic Flaws
Focuses on race conditions, price manipulation, workflow bypass vulnerabilities.

Lab 8: JWT and Session Security
Focuses on JWT alg confusion, session hijacking, secure token implementation.

Lab 9: Secure Coding Workshop
Focuses on input validation, parameterized queries, CSP configuration, secure headers.

Lab 10: Complete Web App Pentest
Focuses on full application penetration test with executive/technical reporting.

Autonomous Navigation (RB601)

← Back to Scheme

Credits: 3 (2-0-2)

Theory

Unit 1
Navigation Fundamentals and Localization: Mobile robot kinematics (differential drive, Ackermann steering), Odometry error accumulation and correction, Monte Carlo Localization (MCL/particle filters), Adaptive Monte Carlo Localization (AMCL), Extended Kalman Filter (EKF) for state estimation, UKF and information filters, Sensor fusion architectures (loose/tight coupling).

Unit 2
Simultaneous Localization and Mapping (SLAM): SLAM problem formulation (chicken-egg problem), EKF-SLAM and covariance management, Graph-SLAM (pose graph optimization), FastSLAM (Rao-Blackwellized particle filters), LiDAR-based SLAM (Gmapping, Cartographer, LOAM), Visual SLAM (ORB-SLAM3, DSO), Loop closure detection and bundle adjustment.

Unit 3
Global Path Planning: Configuration space and environment modeling, Cell decomposition (trapezoidal, Boustrophedon), Potential fields (attractive/repulsive), Sampling-based planners (RRT, RRT*, Informed RRT*), Lattice planners and anytime algorithms, A* and variants (D*, Anytime D*, Field D*), Hybrid A* for non-holonomic constraints.

Unit 4
Local Path Planning and Collision Avoidance: Dynamic Window Approach (DWA), Velocity Obstacles (VO), Reciprocal Velocity Obstacles (RVO), Timed Elastic Bands (TEB), Model Predictive Control (MPC) for trajectory tracking, Pure pursuit and Stanley controllers, Emergency obstacle avoidance (reactive behaviors), Risk-aware planning and chance constraints.

Unit 5
ROS Navigation Stack and Multi-Robot: ROS nav2 stack architecture (Planner Server, Controller Server, BT Navigator), Costmaps (global/local, static/dynamic layers), Nav2 Behavior Trees for mission planning, Multi-robot coordination (CBBA, consensus algorithms), Decentralized planning (conflict-based search, prioritized planning), Fleet management and task allocation.

Practical

Lab 1: Robot Odometry and AMCL
Focuses on differential drive odometry, AMCL tuning, and ROS tf integration.

Lab 2: Gmapping SLAM Implementation
Focuses on real-time mapping with laser scanners, map saving/loading, loop closure tuning.

Lab 3: A* and RRT Path Planning
Focuses on global planner implementation, occupancy grid navigation, and path smoothing.

Lab 4: DWA Local Planner Tuning
Focuses on dynamic obstacle avoidance, velocity limiting, and trajectory scoring.

Lab 5: Nav2 Stack Configuration
Focuses on complete ROS2 nav2 deployment with costmap tuning and recovery behaviors.

Lab 6: Visual SLAM with ORB-SLAM3
Focuses on monocular/stereo visual odometry and real-time mapping.

Lab 7: MPC Trajectory Tracking
Focuses on non-linear MPC implementation for smooth trajectory following.

Lab 8: Multi-Robot Path Planning
Focuses on conflict-based search and prioritized planning for robot fleets.

Lab 9: Risk-Aware Navigation
Focuses on chance-constrained planning and uncertainty-aware trajectory generation.

Lab 10: Autonomous Navigation Project
Focuses on complete navigation system integrating SLAM, planning, and control.

Semester VII

Technical Seminar (SM401)

← Back to Scheme

Credits: 2 (0-2-0)

Course Directive

The Technical Seminar is a structured oral and written presentation course conducted in Semester 7. It is designed to develop the student's ability to communicate technical work rigorously and professionally to a peer and faculty audience. There is no fixed syllabus; the content is driven entirely by the student's own technical experience acquired during the preceding summer.

Seminar Topic Selection
Each student must present on *one* of the following two categories:

Category A — Industry or Research Internship: Students who have completed a minimum 6-week internship (industry or research institution) during the summer following Year 3 must present the technical work undertaken during that internship. The presentation must go beyond a descriptive account and must include: the problem statement and its industrial or research context, the technical approach and tools employed, results and quantitative outcomes, and a critical reflection on limitations and future directions.

Category B — Faculty-Supervised Minor Project: Students who have not undertaken an external internship must complete a minor project under the supervision of a faculty member from their department. The project must involve a non-trivial technical deliverable — a working prototype, a simulation study, a literature survey with original analysis, or a dataset and its analysis — and must be initiated no later than the end of Semester 6. The seminar presentation reports the objectives, methodology, implementation, and findings of this project.

Semester VIII

Major Project (PR402)

← Back to Scheme

Credits: 8 (0-0-8)

Course Directive

The Major Project is the capstone academic requirement of the B.Tech programme, carried out in Semester 8. It represents the culmination of the student's four years of technical education and demands the independent application of domain knowledge, software engineering discipline, and research methodology to a substantial, well-defined problem. Every student must complete a Major Project regardless of internship status or industry employment.

Project Allocation
Projects are allocated by the Department Project Coordination Committee at the beginning of Semester 7 to allow adequate preparation time. Each student is assigned a faculty supervisor whose research or consultancy interests align with the student's major specialization track. Students may propose a self-identified project topic, subject to approval by their supervisor and the committee. Industry-sponsored projects and continuation of internship work are permitted provided a faculty co-supervisor is assigned and the project scope satisfies the academic depth requirements defined below.

Scope and Depth Requirements
The Major Project must demonstrate the following minimum technical characteristics:

- A clearly stated problem with justification of its relevance to the student's specialization track and the broader technical landscape.
- A survey of existing approaches or related work demonstrating familiarity with the state of the art.
- An original technical contribution: this may be a novel system design, an implementation benchmarked against baselines, a dataset with analysis, a simulation study with new findings, or a research result supported by experimental evidence.
- A working software, hardware, or simulation artefact that is demonstrable at the time of evaluation.
- Quantitative evaluation: results must be supported by appropriate metrics, baselines, and statistical or empirical validation.

Advance Elective (Software and Cloud Engineering)

DevOps & Site Reliability Engineering (SE-EL1)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
DevOps Philosophy and CI/CD Pipelines: DevOps cultural principles (CAMS - Culture, Automation, Measurement, Sharing), CALMS framework extension, Continuous Integration practices (trunk-based development, feature flags), Continuous Delivery/Deployment pipelines, GitOps declarative deployments, Pipeline as Code (Jenkinsfile, GitHub Actions, GitLab CI), Artifact management and immutable infrastructure.

Unit 2
Infrastructure as Code and Configuration Management: Terraform workflow (init, plan, apply, destroy), HCL syntax and providers, State management (remote backends, locking), Ansible for configuration management (playbooks, roles, inventories), Puppet/Chef declarative models, Kubernetes as Infrastructure-as-Code (Helm charts, Kustomize), Desired State Configuration (DSC).

Unit 3
Site Reliability Engineering Principles: SRE golden signals (latency, traffic, errors, saturation), Service Level Indicators/Objectives/Agreements (SLI/SLO/SLA), Error budgets and toil reduction, Capacity planning and load testing, Incident management (postmortem culture, blameless retrospectives), On-call engineering and incident response playbooks, Reliability engineering vs. traditional ops.

Unit 4
Observability and Monitoring Stack: The three pillars (metrics, logs, traces), Prometheus time-series monitoring (federation, alerting), Grafana dashboards and anomaly detection, Loki for log aggregation, Jaeger/Zipkin distributed tracing, OpenTelemetry instrumentation, Synthetic monitoring and chaos engineering (Chaos Mesh, LitmusChaos), SLO-based alerting.

Unit 5
Production Excellence and Chaos Engineering: Progressive delivery (feature flags, canary releases, blue-green deployments), A/B testing infrastructure, Multi-region failover and disaster recovery, GitOps with ArgoCD/Flux, Chaos engineering experiments (fault injection, resilience testing), Production readiness reviews (PRR), FinOps and cost optimization at scale.

Practical

Lab 1: Complete CI/CD Pipeline Implementation
Focuses on GitHub Actions/GitLab CI pipeline from code commit to production deployment.

Lab 2: Terraform Multi-Cloud Infrastructure
Focuses on declarative infrastructure provisioning across AWS/Azure/GCP with remote state.

Lab 3: Ansible Configuration Management
Focuses on infrastructure provisioning, application deployment, and compliance checking.

Lab 4: Kubernetes GitOps with ArgoCD
Focuses on declarative deployments, rollouts, and automated synchronization.

Lab 5: Golden Signals Monitoring Stack
Focuses on Prometheus/Grafana deployment with custom exporters and alerting rules.

Lab 6: Distributed Tracing Implementation
Focuses on OpenTelemetry instrumentation across microservices with Jaeger analysis.

Lab 7: SLO/SLI Calculation and Error Budgets
Focuses on service level definition, monitoring, and reliability budgeting.

Lab 8: Chaos Engineering Experiments
Focuses on fault injection (pod kills, network latency, resource exhaustion) and recovery analysis.

Lab 9: Progressive Delivery Implementation
Focuses on canary releases, feature flags, and A/B testing infrastructure.

Lab 10: Production SRE Platform
Focuses on complete SRE platform with monitoring, alerting, GitOps, and chaos engineering.

Agile Project Governance & Management (SE-EL2)

← Back to Scheme

Credits: 4 (3-1-0)

Theory

Unit 1
Agile Governance Frameworks and Principles: Agile Manifesto values and principles revisited, Governance vs. control in agile contexts, SAFe (Scaled Agile Framework) levels (team, program, portfolio), LeSS (Large Scale Scrum), Nexus framework, Spotify engineering model, Disciplined Agile Delivery (DAD), Comparative analysis of scaling frameworks, Governance maturity models for agile adoption.

Unit 2
Program and Portfolio Management: Program Increment (PI) planning in SAFe, ART (Agile Release Train) synchronization, Weighted Shortest Job First (WSJF) prioritization, OKR (Objectives and Key Results) alignment, Portfolio Kanban for value stream management, Lean portfolio management, Investment funding and guardrails, Strategic themes and portfolio epics.

Unit 3
Enterprise Agile Metrics and Analytics: Flow metrics (cycle time, lead time, throughput), Value stream mapping and bottleneck analysis, Cumulative Flow Diagrams (CFD), Predictability metrics (planned vs. actual velocity), Quality metrics (defect escape rate, DORA metrics), Business value delivered vs. planned, OKR achievement tracking, Data-driven decision making dashboards.

Unit 4
Risk Management and Compliance in Agile: Agile risk management (product, technical, organizational risks), Risk-adjusted planning and probabilistic forecasting, Monte Carlo simulation for sprint capacity, Compliance by design (regulatory guardrails, audit trails), Governance ceremonies (PI planning, System Demos, Inspect & Adapt), Decision rights and escalation paths, Architectural runway governance.

Unit 5
Organizational Transformation and Change: Agile assessment models (Shu-Ha-Ri, Satir Change Model), Enterprise agile coaching frameworks, Transformation backlogs and roadmaps, Communities of Practice (CoP) establishment, Leadership alignment and executive sponsorship, Metrics for transformation success, Scaling anti-patterns and failure modes, Sustainable agile adoption strategies.

Practical

Lab 1: SAFe PI Planning Simulation
Focuses on Program Increment planning, dependency mapping, and risk assessment workshop.

Lab 2: WSJF Prioritization Framework
Focuses on business value/cost/time criticality analysis for portfolio backlog prioritization.

Lab 3: Flow Metrics Dashboard Development
Focuses on implementing cycle time, throughput, and CFD visualizations using real project data.

Lab 4: Monte Carlo Sprint Forecasting
Focuses on probabilistic velocity prediction and confidence intervals for release planning.

Lab 5: Value Stream Mapping Workshop
Focuses on identifying bottlenecks and waste in software delivery pipelines.

Lab 6: OKR Cascade Implementation
Focuses on organizational goal alignment from portfolio to team level.

Lab 7: Agile Maturity Assessment
Focuses on organizational agile assessment and transformation roadmap development.

Lab 8: Risk-Adjusted Backlog Management
Focuses on incorporating uncertainty and risk into sprint and release planning.

Lab 9: Governance Dashboard Creation
Focuses on executive reporting with key agile metrics and portfolio health indicators.

Lab 10: Enterprise Agile Simulation
Focuses on full-scale agile transformation simulation with multiple teams and dependencies.

Serverless Architecture (SE-EL3)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Serverless Fundamentals and FaaS: Serverless definition and characteristics (no server management, auto-scaling, pay-per-use), Function-as-a-Service (FaaS) architecture, Cold start problem and mitigation strategies, Event-driven execution model, AWS Lambda vs. Azure Functions vs. Google Cloud Functions comparison, Stateless function design principles, Execution environment lifecycle.

Unit 2
Serverless Event Sources and Integration: Event triggers (HTTP API Gateway, S3 events, DynamoDB streams, Kinesis, SQS), Asynchronous invocation patterns (event sources, polling), Synchronous vs. asynchronous execution modes, Dead Letter Queues (DLQ) and retry policies, Destination configurations, Provisioned concurrency, Function versioning and aliases.

Unit 3
Serverless Data Services: DynamoDB design patterns (single-table design, GSI/GSI2, LSI), DAX caching layer, Step Functions workflow orchestration (state machines, choice states, parallel), EventBridge event bus (rules, targets, schema registry), AppSync GraphQL APIs (resolvers, data sources), BaaS services integration patterns.

Unit 4
Advanced Serverless Patterns: Backend-for-Frontend (BFF) pattern, Multi-tenancy strategies, CQRS/Event Sourcing in serverless, Fan-out/fan-in patterns, Saga pattern for distributed transactions, State machines for long-running workflows, Cross-account resource sharing, VPC integration and private endpoints.

Unit 5
Serverless Observability, Security, and Cost: CloudWatch Insights and X-Ray tracing for serverless, Lambda Power Tuning, Distributed tracing across FaaS + BaaS, IAM roles and least privilege for functions, Secrets management integration, WAF and DDoS protection, Cost optimization (provisioned concurrency vs. reserved, architectural patterns), Vendor lock-in mitigation strategies.

Practical

Lab 1: FaaS Event-Driven Applications
Focuses on Lambda/API Gateway/S3 event triggers, cold start analysis, and timeout handling.

Lab 2: DynamoDB Single-Table Design
Focuses on GSI patterns, access patterns mapping, and performance benchmarking.

Lab 3: Step Functions Workflow Orchestration
Focuses on complex state machines, error handling, and long-running business processes.

Lab 4: Serverless GraphQL APIs
Focuses on AppSync resolvers, data source integration, and real-time subscriptions.

Lab 5: Event-Driven Microservices
Focuses on EventBridge patterns, fan-out architectures, and cross-service communication.

Lab 6: Serverless BFF Implementation
Focuses on mobile/web backend optimization and client-specific API layers.

Lab 7: Distributed Tracing Analysis
Focuses on X-Ray integration, performance bottleneck identification, and optimization.

Lab 8: Serverless Security Hardening
Focuses on IAM fine-grained policies, VPC endpoints, and secrets management.

Lab 9: Cost Optimization Workshop
Focuses on Lambda Power Tuning, architectural refactoring for cost efficiency.

Lab 10: Complete Serverless Application
Focuses on production-grade serverless system with CI/CD, monitoring, and multi-region deployment.

Secure Software Development (DevSecOps) (SE-EL4)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
DevSecOps Principles and Secure SDLC: Shift-left security philosophy, Secure by design principles, Threat modeling integration (STRIDE, DREAD, PASTA), Security champions and DevSecOps culture, Secure coding standards (OWASP Secure Coding Practices, CERT Secure Coding), Security requirements definition, Threat modeling in sprint planning, Security Story Points estimation.

Unit 2
Static and Dynamic Application Security Testing: Static Application Security Testing (SAST) tools (SonarQube, Checkmarx, Semgrep), Incremental SAST scanning, False positive management, Dynamic Application Security Testing (DAST) (OWASP ZAP, Burp Suite), Interactive AST (IAST), Software Composition Analysis (SCA) for open source (Dependabot, Snyk), License compliance scanning.

Unit 3
CI/CD Security and Infrastructure Protection: Secrets management (HashiCorp Vault, AWS Secrets Manager, Doppler), GitOps security (protected branches, code owners), Container security (Trivy, Clair, Anchore), Kubernetes security (RBAC, NetworkPolicies, PodSecurityPolicies), Infrastructure as Code scanning (Checkov, Terrascan), Pipeline security (supply chain attacks, SBOM generation).

Unit 4
Cloud Security and Compliance Automation: Cloud security posture management (CSPM), Infrastructure Live (AWS Config, Azure Policy), IAM policy as code (OPA, Kyverno), Compliance as code (Open Policy Agent), Cloud Workload Protection Platforms (CWPP), Data classification and encryption at rest/transit, Multi-account strategies and cross-account access.

Unit 5
Threat Detection, Response, and Continuous Monitoring: Runtime Application Self-Protection (RASP), Behavioral analysis and anomaly detection, Security Information and Event Management (SIEM), SOAR platforms integration, Threat hunting in development environments, Security metrics and dashboards (MTTR, MTTD, vulnerability age), Attack surface management, Zero Trust implementation.

Practical

Lab 1: Secure Coding and SAST Integration
Focuses on SonarQube/Semgrep pipeline integration, secure coding workshops, and false positive triage.

Lab 2: DAST and IAST Deployment
Focuses on OWASP ZAP/Burp automation in CI/CD, shift-left DAST scanning.

Lab 3: Container and Kubernetes Security
Focuses on Docker image scanning, kube-bench audits, NetworkPolicy implementation.

Lab 4: Secrets Management Implementation
Focuses on HashiCorp Vault integration, GitOps secret injection, zero-secret commits.

Lab 5: IaC Security Scanning
Focuses on Checkov/Terrascan for Terraform/CloudFormation, compliance policy enforcement.

Lab 6: SCA and SBOM Generation
Focuses on open source vulnerability management, license compliance, dependency updates.

Lab 7: Cloud IAM Policy as Code
Focuses on OPA Gatekeeper, Kyverno policies, least privilege IAM automation.

Lab 8: Runtime Security Monitoring
Focuses on Falco rules, eBPF-based monitoring, behavioral anomaly detection.

Lab 9: Threat Modeling Workshop
Focuses on STRIDE/DREAD modeling, security story backlog creation.

Lab 10: Complete DevSecOps Pipeline
Focuses on end-to-end secure delivery pipeline from code to production with full observability.

Enterprise Application Architecture (SE-EL5)

← Back to Scheme

Credits: 4 (3-1-0)

Theory

Unit 1
Enterprise Architecture Frameworks and TOGAF: The Open Group Architecture Framework (TOGAF) ADM cycle, Architecture Development Method phases (preliminary, architecture vision, business/data/application/technology architecture), Architecture repository and content framework, Enterprise continuum and solutions landscape, Architecture governance and compliance, Zachman framework comparison, TOGAF certification levels.

Unit 2
Domain-Driven Design for Enterprise Applications: Strategic DDD patterns (bounded contexts, context maps, core/ supporting/ generic domains), Ubiquitous language establishment, Aggregate design principles (consistency boundaries, transactional consistency), Domain events and event storming workshops, Anti-corruption layer patterns, Hexagonal architecture and ports/adapters, CQRS and Event Sourcing implementation.

Unit 3
Enterprise Integration Patterns: Messaging patterns (publish-subscribe, point-to-point, request-reply), Enterprise Service Bus (ESB) vs. modern alternatives, API management lifecycle (design, security, versioning, deprecation), Canonical data models and contract-first design, Transaction strategies (Sagas, choreography, orchestration), Data consistency patterns (eventual consistency, 2PC alternatives).

Unit 4
Microservices and SOA Governance: Service inventory and capability mapping, Service contracts and SLAs, Service versioning strategies (major/minor/semver), Circuit breaker and bulkhead patterns, Service mesh governance (Istio, Consul), Centralized logging and distributed tracing, Golden signals monitoring for enterprise services, Multi-tenancy architecture patterns.

Unit 5
Enterprise Reference Architectures: Java EE reference architecture (EJB, JPA, JSF, CDI), Spring Boot enterprise patterns, .NET Core enterprise architecture, Cloud-native enterprise patterns (12-factor apps, strangler pattern), Legacy modernization strategies (strangler fig, branch by abstraction), Reference architectures (J2EE Petstore, Netflix OSS stack), Architecture decision records (ADRs).

Practical

Lab 1: TOGAF ADM Workshop
Focuses on Architecture Vision creation, stakeholder map development, and governance framework design.

Lab 2: Domain-Driven Design Event Storming
Focuses on bounded context identification, aggregate modeling, and ubiquitous language workshops.

Lab 3: Enterprise Integration Patterns
Focuses on messaging patterns implementation with Apache Camel/Kafka Connect.

Lab 4: Hexagonal Architecture Implementation
Focuses on ports/adapters pattern with domain-driven core and multiple technology adapters.

Lab 5: CQRS/ES Implementation
Focuses on event sourcing, materialized views, and command/query separation.

Lab 6: Service Mesh Enterprise Deployment
Focuses on Istio multi-cluster setup, traffic management, and observability.

Lab 7: API Management Platform
Focuses on API gateway configuration, versioning, rate limiting, and developer portal.

Lab 8: Multi-Tenancy Architecture
Focuses on database-per-tenant vs. shared database strategies implementation.

Lab 9: Legacy Modernization Patterns
Focuses on strangler pattern implementation and branch-by-abstraction techniques.

Lab 10: Enterprise Reference Architecture
Focuses on complete enterprise application following TOGAF/DDD patterns with production deployment.

Software Metrics & Quality Engineering (SE-EL6)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
Software Measurement Theory and Fundamentals: Measurement theory foundations (representational theory, scales of measurement), Product vs. process vs. resource metrics, Goal-Question-Metric (GQM) paradigm, Quality models (ISO 9126, ISO 25010, SQALE), McCall quality factors, Validation of measurement programs, Metric misuse pitfalls and validity criteria.

Unit 2
Size and Complexity Metrics: Lines of Code (LOC) variants and normalization, Function Point Analysis (FPA) - unadjusted/adjusted weights, COSMIC Function Points, Halstead software science (operators/operands, volume, difficulty), Cyclomatic complexity (McCabe), Essential complexity, Cognitive complexity, Maintainability Index (MI).

Unit 3
Coupling, Cohesion, and Design Quality: Coupling metrics (Henry/Kafura, Chidamber/Kemerer), Cohesion measures (LCOM, Lack of Cohesion of Methods), Object-oriented metrics suite (CK metrics - WMC, DIT, NOC, CBO), Class stability/maintainability metrics, Architectural metrics (dependency cycles, component balance), Refactoring impact analysis.

Unit 4
Process and Project Management Metrics: Effort estimation metrics (COCOMO II, planning poker accuracy), Velocity and burndown metrics, Defect density and removal efficiency, Phase defect containment effectiveness, Escaped defects and customer issue rates, Test coverage metrics (statement, branch, MC/DC), DORA metrics (deployment frequency, lead time, MTTR, change failure rate).

Unit 5
Software Reliability and Quality Prediction: Reliability growth models (JM, Musa logarithmic, Duane), Failure intensity prediction, Operational profile construction, Quality prediction models (orthogonal defect classification), Bayesian reliability analysis, Six Sigma defect reduction (DMAIC), Benchmarking against industry standards (ISBSG repository), Actionable insights from metrics dashboards.

Advance Elective (Artificial Intelligence)

Reinforcement Learning (ML-EL1)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Reinforcement Learning Foundations: Markov Decision Processes (MDPs) formalization (states, actions, transition probabilities, rewards), Bellman equations and optimality principle, Model-free vs. model-based RL, Exploration vs. exploitation tradeoff (ϵ-greedy, UCB, entropy regularization), Discounted future rewards and value iteration, Policy iteration algorithm, Generalized Policy Iteration (GPI).

Unit 2
Model-Free Temporal Difference Learning: Monte Carlo methods (first-visit, every-visit, importance sampling), Temporal Difference (TD) learning (TD(θ), TD(λ)), Q-Learning algorithm and off-policy learning, SARSA and on-policy control, Double Q-Learning and expected SARSA, N-step bootstrapping methods, Eligibility traces implementation.

Unit 3
Function Approximation and Deep RL: Linear function approximation (average reward, gradient TD), Tile coding and radial basis functions, Deep Q-Networks (DQN) architecture (experience replay, target networks), Double DQN improvements, Dueling DQN and distributional RL, Rainbow DQN combining multiple improvements, Policy gradient methods (REINFORCE, actor-critic).

Unit 4
Policy Gradient Methods and Actor-Critic: REINFORCE algorithm with baseline subtraction, Advantage Actor-Critic (A2C/A3C), Proximal Policy Optimization (PPO) - clipped objective, Trust Region Policy Optimization (TRPO), Soft Actor-Critic (SAC), Deterministic Policy Gradients (DPG), Twin Delayed DDPG (TD3), Continuous control benchmarks (MuJoCo, PyBullet).

Unit 5
Advanced RL Topics and Applications: Multi-agent RL (MARL) cooperation/competition, Centralized training with decentralized execution (CTDE), Self-play and population-based training, Model-based RL (Dyna, MBPO, Dreamer), Hierarchical RL (options framework, Feudal Networks), Offline RL (conservative Q-learning, behavior regularization), RLHF (RL from Human Feedback) for LLM alignment.

Practical

Lab 1: MDPs and Dynamic Programming
Focuses on value iteration, policy iteration, and MDP environment implementation.

Lab 2: Monte Carlo and TD Learning
Focuses on MC/TD methods comparison on classic environments (Cliff Walking, Taxi).

Lab 3: Q-Learning and SARSA Implementation
Focuses on off-policy/on-policy control, exploration strategies, and convergence analysis.

Lab 4: Deep Q-Network (DQN) Training
Focuses on Atari games, experience replay, target networks, and hyperparameter tuning.

Lab 5: Policy Gradient Methods
Focuses on REINFORCE implementation for discrete/continuous action spaces.

Lab 6: Actor-Critic Algorithms
Focuses on A2C/A3C parallel training and advantage estimation techniques.

Lab 7: PPO and TRPO Implementation
Focuses on constrained policy optimization and sample efficiency comparison.

Lab 8: Continuous Control with SAC/TD3
Focuses on MuJoCo robotics environments and state-of-the-art continuous RL.

Lab 9: Multi-Agent RL Scenarios
Focuses on cooperative/competitive MARL with self-play training.

Lab 10: Custom RL Environment Project
Focuses on complete RL agent development for robotics/navigation application.

MLOps & Production AI (ML-EL2)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
MLOps Fundamentals and ML Lifecycle: ML project lifecycle stages (data, training, validation, deployment, monitoring), MLOps maturity levels (manual, ML automation, continuous ML), Model drift types (concept drift, data drift, upstream drift), Golden ML pipelines, Experiment tracking and reproducibility, Versioning strategies (data, models, code, environment), ML metadata stores and lineage tracking.

Unit 2
Data Management and Feature Engineering: Data versioning (DVC, Delta Lake), Data quality gates and schema validation, Feature stores (Feast, Tecton, Hopsworks), Online/offline feature serving, Feature drift detection, Data lineage and impact analysis, Automated data validation (Great Expectations, Deequ), PII/PHI data anonymization and compliance.

Unit 3
Model Development and Experimentation: MLflow tracking server and experiment management, Weights & Biases (W&B) sweeps and hyperparameter optimization, Ray Tune distributed tuning, Model registries and staging environments, A/B testing frameworks (Optimizely, Flagr), Canary deployments and shadow testing, Model performance baselines and champion/challenger patterns.

Unit 4
Model Deployment and Serving: Model packaging (ONNX, TorchServe, TensorFlow Serving), Containerization best practices (Docker multi-stage builds), Kubernetes inference servers (KFServing, Seldon Core, KServe), Serverless ML (AWS SageMaker Serverless, Cloud Run), Edge deployment (TensorFlow Lite, ONNX Runtime), Multi-model serving and GPU sharing, Batch vs. real-time inference.

Unit 5
ML Monitoring, Governance, and Continuous Training: Model monitoring (KServe model monitoring, Prometheus integration), Drift detection algorithms (KS2 test, PSI, MMD), Automated retraining triggers and CI/CD for ML, Human-in-the-loop validation, Model explainability (SHAP, LIME integration), Bias/fairness monitoring (AIF360, Fairlearn), Compliance frameworks (GDPR, HIPAA ML controls), Blue-green ML deployments.

Practical

Lab 1: ML Pipeline with MLflow
Focuses on complete ML lifecycle tracking, experiment comparison, and model registry usage.

Lab 2: Feature Store Implementation
Focuses on Feast feature store setup, online/offline serving, and feature drift monitoring.

Lab 3: Automated Hyperparameter Tuning
Focuses on Ray Tune sweeps, parallel training, and optimization visualization.

Lab 4: Model Deployment with KServe
Focuses on Kubernetes inference serving, traffic splitting, and autoscaling.

Lab 5: A/B Testing Infrastructure
Focuses on canary deployments, statistical significance testing, and rollout analysis.

Lab 6: Model Monitoring and Drift Detection
Focuses on drift detection pipelines, alerting configuration, and performance degradation analysis.

Lab 7: Continuous Training Pipeline
Focuses on automated retraining triggers, model validation, and champion/challenger selection.

Lab 8: Explainable AI Integration
Focuses on SHAP/LIME integration with production models and drift-aware explanations.

Lab 9: Multi-Model Serving Platform
Focuses on GPU sharing, dynamic model loading, and inference cost optimization.

Lab 10: Production ML Platform
Focuses on complete MLOps platform with data versioning, automated training, deployment, and monitoring.

Generative AI & Large Language Models (ML-EL3)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Generative Modeling Foundations: Generative vs. discriminative models, Maximum likelihood estimation for density estimation, Autoregressive models and sequential generation, Latent variable models, Evaluation metrics (log-likelihood, FID, IS scores), Mode collapse and posterior collapse problems, Training instabilities and divergence measures.

Unit 2
Variational Autoencoders and Diffusion Models: VAE architecture (encoder-decoder, KL divergence), Reparameterization trick and β-VAE variants, Hierarchical VAEs and VQ-VAE, Denoising Diffusion Probabilistic Models (DDPM), Forward/reverse diffusion process, Denoising score matching, Classifier-free guidance, Latent diffusion models (Stable Diffusion).

Unit 3
Large Language Model Architectures: Decoder-only transformers (GPT series evolution), Rotary Position Embeddings (RoPE), Grouped-Query Attention (GQA), Mixture of Experts (MoE) scaling, Context window extension techniques (RoPE scaling, ALiBi), FlashAttention and memory-efficient attention, Speculative decoding and key-value caching optimizations.

Unit 4
Pretraining, Alignment, and Scaling Laws: Self-supervised pretraining objectives (causal LM, masked LM, prefix LM), Chinchilla scaling laws and compute-optimal training, Instruction tuning and dataset curation, Reinforcement Learning from Human Feedback (RLHF), Proximal Policy Optimization (PPO) for alignment, Direct Preference Optimization (DPO), Constitutional AI and self-improvement.

Unit 5
Generative AI Applications and Deployment: Retrieval-Augmented Generation (RAG) architectures, Agentic workflows (ReAct, Toolformer), Multimodal generation (CLIP, DALL-E, Flamingo), Text-to-image/video/speech generation, Guardrails and content moderation, Model evaluation (human preference, Elo rankings), Production deployment (vLLM, TGI, quantization, distillation), Ethical considerations and safety measures.

Practical

Lab 1: VAE Implementation
Focuses on complete VAE training, latent space visualization, and β-VAE experimentation.

Lab 2: Diffusion Model Training
Focuses on DDPM implementation, sampling visualization, and classifier guidance.

Lab 3: Transformer from Scratch
Focuses on GPT-like decoder implementation with RoPE and FlashAttention.

Lab 4: Fine-tuning Small Language Models
Focuses on instruction tuning datasets and LoRA parameter-efficient fine-tuning.

Lab 5: RLHF with PPO/DPO
Focuses on preference dataset creation and alignment training.

Lab 6: RAG Pipeline Development
Focuses on dense retrieval, reranking, and generation with knowledge augmentation.

Lab 7: Multimodal Generation
Focuses on CLIP-guided diffusion and text-to-image synthesis.

Lab 8: LLM Agent Development
Focuses on ReAct agents with tool calling and reasoning capabilities.

Lab 9: Production LLM Serving
Focuses on vLLM/TGI deployment, quantization, and batching optimization.

Lab 10: Generative AI Application
Focuses on complete production generative system with evaluation and safety measures.

Multi-Agent Systems (ML-EL4)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
Multi-Agent System Foundations: Agent definitions and properties (autonomy, reactivity, proactivity, social ability), Multi-agent vs. single-agent systems, Agent architectures (reactive, deliberative, hybrid, BDI), Environments (accessible, deterministic, dynamic, continuous, multi-agent), Coordination vs. cooperation vs. competition, Communication languages (KQML, FIPA-ACL), Agent platforms (JADE, SPADE).

Unit 2
Distributed Problem Solving: Task decomposition and allocation (Contract Net Protocol, auction-based), Centralized vs. distributed planning, Partial global planning and approximate coordination, Robustness to agent failures, Negotiation protocols (monotonic concession, argumentation-based), Game-theoretic negotiation (Nash equilibria, subgame perfect equilibrium), Incentive-compatible mechanisms.

Unit 3
Cooperative Multi-Agent Learning: Centralized Training with Decentralized Execution (CTDE), Independent Q-Learning (IQL) limitations, Value Decomposition Networks (VDN), QMIX monotonic factorization, MADDPG for continuous control, Counterfactual Multi-Agent Policy Gradients (COMA), Communication in MARL (differentiable communication channels, gated attention), Credit assignment problem.

Unit 4
Competitive Multi-Agent Systems: Zero-sum and general-sum games, Minimax and alpha-beta pruning, Nash equilibria computation (fictitious play, no-regret learning), Policy gradient methods in games (self-play, population-based training), AlphaGo/AlphaZero algorithms, MuZero model-based planning, OpenAI Five and multi-agent Dota 2, Elo rating systems for agent evaluation.

Unit 5
Applications and Emerging Topics: Swarm robotics coordination (flocking, formation control), Traffic management and autonomous vehicle platooning, Smart grids and energy management MAS, Financial market multi-agent simulation, Human-agent collaboration (shared workspace, mixed-initiative), Hierarchical multi-agent systems, Safe AI and value alignment in MAS, Decentralized AI governance.

Explainable AI (XAI) & AI Governance (ML-EL5)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
XAI Fundamentals and Taxonomy: Explainability vs. interpretability vs. transparency, Scope of explanations (local/global, model-agnostic/model-specific), Explanation types (counterfactuals, prototypes, saliency maps), Human-centered XAI design principles, Mental models and explanation satisfaction, Trade-offs between accuracy and explainability, Regulatory requirements driving XAI (GDPR right to explanation, AI Act).

Unit 2
Model-Agnostic Explanation Methods: LIME (Local Interpretable Model-agnostic Explanations) algorithm, Kernel density weighting and sparse linear models, SHAP (SHapley Additive exPlanations) - Shapley values from game theory, Kernel SHAP vs. Tree SHAP, Anchors for high-precision rule extraction, Partial Dependence Plots (PDP) and Individual Conditional Expectation (ICE), Accumulated Local Effects (ALE) plots.

Unit 3
Model-Specific Interpretability Techniques: Gradient-based attribution methods (Vanilla gradients, Integrated Gradients, Saliency maps), Layer-wise Relevance Propagation (LRP), DeepLIFT and Deep Taylor decomposition, Attention visualization and interpretability, Decision tree surrogate models, Prototypical networks and case-based explanations, Intrinsic interpretable models (rule lists, generalized additive models).

Unit 4
AI Fairness, Bias, and Ethics: Bias sources (representation, historical, measurement bias), Fairness metrics (demographic parity, equal opportunity, equalized odds), Group fairness vs. individual fairness, Counterfactual fairness, Algorithmic discrimination quantification, Fairness interventions (pre/pre/post-processing), Intersectional fairness and multi-attribute discrimination, Responsible AI frameworks.

Unit 5
AI Governance Frameworks and Risk Management: AI governance maturity models, AI risk management frameworks (NIST AI RMF, ISO/IEC 42001), Algorithmic accountability and auditability, Model cards and datasheets for datasets, AI ethics boards and review processes, Impact assessments (Data Protection Impact Assessment), International AI regulations (EU AI Act, US Executive Order), Deployment governance (human oversight, contestability rights).

Probabilistic Graphical Models (ML-EL6)

← Back to Scheme

Credits: 4 (3-1-0)

Theory

Unit 1
Probabilistic Graphical Models Fundamentals: Directed (Bayesian networks) vs. undirected (Markov Random Fields) models, Factorization of joint distributions, Conditional independence and d-separation, Markov properties and Hammersley-Clifford theorem, Graphical model representation advantages, Exponential family distributions and sufficiency, Common probability distributions in graphical models.

Unit 2
Inference Algorithms: Exact inference (variable elimination, belief propagation), Complexity analysis of inference, Approximate inference (importance sampling, likelihood weighting), Markov Chain Monte Carlo (MCMC) methods (Metropolis-Hastings, Gibbs sampling), Sequential Monte Carlo (particle filters), Loopy belief propagation and mean field approximation.

Unit 3
Learning Graphical Models: Parameter learning in Bayesian networks (maximum likelihood, MAP estimation), Dirichlet priors and Dirichlet-multinomial models, Structure learning (score-based, constraint-based), K2 algorithm and greedy hill-climbing, Markov blanket discovery, PC algorithm for DAG structure learning, Learning Markov Random Fields (pseudolikelihood estimation).

Unit 4
Dynamic Bayesian Networks and Hidden Markov Models: Hidden Markov Models (HMM) - three fundamental problems (evaluation, decoding, learning), Forward-backward algorithm, Viterbi algorithm, Baum-Welch EM algorithm, Dynamic Bayesian Networks (DBN) for time series, Switching Kalman filters, Continuous-time graphical models.

Unit 5
Advanced Topics and Applications: Gaussian graphical models and precision matrix estimation, Conditional Random Fields (CRF) for structured prediction, Factor graphs and sum-product algorithm, Nonparametric Bayesian models (Indian Buffet Process), Deep generative models with graphical structure, Causal discovery from observational data, Applications in bioinformatics, NLP, and robotics.

Practical

Lab 1: Bayesian Network Inference
Focuses on exact inference using variable elimination and belief propagation.

Lab 2: MCMC Sampling Methods
Focuses on Metropolis-Hastings, Gibbs sampling convergence diagnostics.

Lab 3: Hidden Markov Model Implementation
Focuses on Viterbi path decoding, forward-backward probabilities, EM learning.

Lab 4: Parameter Learning in BNs
Focuses on MLE/MAP estimation with missing data and Dirichlet priors.

Lab 5: Structure Learning Algorithms
Focuses on score-based and constraint-based DAG learning from data.

Lab 6: Dynamic Bayesian Networks
Focuses on time series modeling and filtering/smoothing applications.

Lab 7: Conditional Random Fields
Focuses on sequence labeling (NER, POS tagging) with CRF models.

Lab 8: Gaussian Graphical Models
Focuses on precision matrix estimation and partial correlation networks.

Lab 9: Factor Graph Inference
Focuses on sum-product algorithm for complex graphical structures.

Lab 10: Graphical Models Application
Focuses on complete PGM solution for domain-specific prediction task.

Advance Elective (Data Science)

High-Dimensional Data Analysis (DS-EL1)

← Back to Scheme

Credits: 4 (3-1-0)

Theory

Unit 1
High-Dimensional Phenomena and Curse of Dimensionality: Distance concentration and emptiness phenomenon, Concentration of measure inequality, Nearest neighbor distances in high dimensions, Sparsity in high-dimensional data, Double descent phenomenon, Blessing of dimensionality vs. curse, Dimension reduction necessity, Johnson-Lindenstrauss lemma for random projections.

Unit 2
Principal Component Analysis and Classical Methods: PCA mathematical formulation (covariance matrix eigendecomposition, SVD), PCA variants (Kernel PCA, Sparse PCA, Robust PCA), Incremental/online PCA, Factor analysis and independent component analysis (ICA), Multidimensional scaling (MDS - classical, non-metric), Isomap and geodesic distances.

Unit 3
Nonlinear Dimensionality Reduction: t-SNE algorithm (student-t divergence, perplexity tuning), UMAP (uniform manifold approximation and projection), LargeVis and PHATE, Autoencoder architectures (vanilla, variational, denoising), Deep belief networks for representation learning, Self-supervised contrastive learning (SimCLR, MoCo), Manifold learning assumptions and topology preservation.

Unit 4
High-Dimensional Statistics and Regularization: Multiple testing problem and FDR control (Benjamini-Hochberg procedure), Sparsity-inducing regularization (Lasso, Elastic Net, Group Lasso), Stability selection and knockoffs framework, High-dimensional covariance estimation (graphical models, covariance shrinkage), Robust high-dimensional regression, Sure independence screening (SIS).

Unit 5
Scalable Algorithms and Embeddings: Random projection methods (CountSketch, Johnson-Lindenstrauss transforms), Locality-Sensitive Hashing (LSH) families (random hyperplanes, p-stable distributions), Approximate nearest neighbors (HNSW, FAISS), Word embeddings scaling (GloVe, fastText), Graph embeddings (Node2Vec, DeepWalk), Tensor decomposition for multi-modal data.

Practical

Lab 1: PCA and Classical DR Comparison
Focuses on eigendecomposition vs. SVD, explained variance analysis, reconstruction error.

Lab 2: Nonlinear DR Visualization
Focuses on t-SNE/UMAP parameter tuning, perplexity analysis, cluster separation metrics.

Lab 3: Autoencoder Implementation
Focuses on latent space visualization, reconstruction quality, anomaly detection use cases.

Lab 4: High-Dimensional Regression
Focuses on Lasso path algorithms, cross-validation, stability selection analysis.

Lab 5: LSH and ANN Search
Focuses on FAISS/HNSW indexing, recall@K evaluation, query time benchmarking.

Lab 6: Random Projections Scalability
Focuses on JL lemma verification, dimensionality reduction performance scaling.

Lab 7: Graph Embeddings
Focuses on Node2Vec parameter tuning, link prediction evaluation.

Lab 8: Multiple Testing Control
Focuses on FDR procedures, power analysis, volcano plots for genomics data.

Lab 9: Sparse Covariance Estimation
Focuses on graphical lasso, precision matrix visualization, conditional independence.

Lab 10: High-Dimensional Pipeline
Focuses on complete analysis workflow from raw HD data to interpretable low-D representations.

Business Intelligence & Data Warehousing (DS-EL2)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Data Warehousing Concepts and Architecture: Operational Data Store (ODS) vs. data warehouse, Bill Inmon vs. Ralph Kimball approaches (normalized vs. star schema), Data warehouse architecture (staging, ETL, presentation layer), conformed dimensions and fact granularity, Slowly Changing Dimensions (SCD Type 1-6), Surrogate keys and dimensionality modeling.

Unit 2
OLAP and Multidimensional Analysis: Online Analytical Processing (OLAP) operations (slice/dice, drill-down/up, pivot), MOLAP vs. ROLAP vs. HOLAP, OLAP cube construction and sparse cube optimization, Bitmap indexing for high-cardinality dimensions, Aggregation hierarchies and roll-up strategies, Materialized views and summary tables.

Unit 3
BI Tools and Visualization: Self-service BI platforms (Tableau, Power BI, Looker), Dashboard design principles and storytelling, KPI definition and cascading, Advanced visualizations (small multiples, sparklines, heatmaps), Geographic visualization and custom polygons, Interactive filtering and parameter actions, Embedded analytics and white-labeling.

Unit 4
ETL/ELT Pipelines and Data Quality: Extract-Transform-Load vs. Extract-Load-Transform paradigms, Change Data Capture (CDC) techniques (log-based, timestamp, triggers), Data quality dimensions (accuracy, completeness, consistency, timeliness), Data profiling and anomaly detection, Master Data Management (MDM) strategies, Data lineage and impact analysis.

Unit 5
Modern Data Platforms and Lakehouse: Data lake architectures and schema-on-read challenges, Lakehouse paradigm (Delta Lake, Apache Iceberg, Hudi), Semantic layer abstraction (dbt, LookML), Headless BI and metric stores (Metrics Layer), Real-time analytics (Kafka Streams, Flink SQL), Federated query engines (Presto, Trino, Athena).

Practical

Lab 1: Dimensional Modeling Workshop
Focuses on star/snowflake schema design, SCD implementation, surrogate key generation.

Lab 2: OLAP Cube Development
Focuses on multidimensional cube construction, aggregation hierarchies, slice/dice operations.

Lab 3: BI Dashboard Creation
Focuses on Tableau/Power BI storytelling, KPI dashboards, interactive parameters/filters.

Lab 4: Advanced Visualizations
Focuses on custom polygons, small multiples, geographic heatmaps, trend analysis.

Lab 5: ETL Pipeline Implementation
Focuses on Apache Airflow DAGs, CDC with Debezium, data transformation patterns.

Lab 6: Data Quality Framework
Focuses on Great Expectations validation, anomaly detection, data profiling.

Lab 7: Lakehouse Architecture Setup
Focuses on Delta Lake tables, time travel, ACID transactions, schema evolution.

Lab 8: Semantic Layer Development
Focuses on dbt transformations, LookML models, metric definitions.

Lab 9: Real-Time Analytics Pipeline
Focuses on Kafka + Flink streaming analytics and materialized views.

Lab 10: Complete BI Solution
Focuses on end-to-end data warehouse to executive dashboard delivery.

Mining Massive Datasets (DS-EL3)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Scalable Data Mining Frameworks: MapReduce programming model revisited, Beyond MapReduce (Spark, Dask), Data stream mining challenges (concept drift, memory constraints), Sliding window models, Reservoir sampling, Massive data partitioning strategies, Approximate query processing and sampling guarantees, Sketching algorithms fundamentals.

Unit 2
Frequent Itemset Mining and Association Rules: Apriori algorithm and candidate generation, FP-growth and FP-tree structure, Eclat and vertical data format, Sampling-based frequent itemset mining, Parallel/distributed FP-growth, Association rule interestingness measures (support, confidence, lift, conviction), Sequential pattern mining (GSP, SPADE, PrefixSpan).

Unit 3
Graph Mining and Network Analysis: Graph representation (adjacency lists, CSR format), PageRank algorithm and variants (weighted, personalized), HITS algorithm, Triangle counting and clustering coefficients, Community detection (Louvain, spectral clustering, label propagation), Graph sampling and streaming algorithms, Subgraph isomorphism and motif discovery.

Unit 4
Dimensionality Reduction and Clustering at Scale: Mini-batch k-means and scalable EM, Canopy clustering, BIRCH hierarchical clustering, Spectral clustering approximation, t-SNE/UMAP for large datasets, Random projection trees, Streaming PCA and incremental SVD, CUR matrix decomposition for interpretability.

Unit 5
Near-Duplicates and Recommendation Systems: MinHash and Locality Sensitive Hashing (LSH) for Jaccard similarity, SimHash for text documents, Near-duplicate detection at web scale, Matrix factorization (SVD, ALS, NTF), Neighborhood-based collaborative filtering, Content-based recommendation, Hybrid recommenders, Scalable bandit algorithms (LinUCB, Thompson sampling).

Practical

Lab 1: Scalable MapReduce/Spark Implementation
Focuses on distributed word count, log analysis, and custom partitioners.

Lab 2: Frequent Itemset Mining
Focuses on Apriori/FP-growth comparison, parallel implementation, association rules.

Lab 3: PageRank and Graph Analytics
Focuses on GraphFrames/GraphX implementation, triangle counting, community detection.

Lab 4: LSH for Near-Duplicates
Focuses on MinHash/LSH indexing, recall-precision tradeoff analysis.

Lab 5: Streaming Data Processing
Focuses on Spark Streaming, sliding window aggregations, anomaly detection.

Lab 6: Scalable Clustering
Focuses on mini-batch k-means, BIRCH, spectral clustering approximation.

Lab 7: Matrix Factorization Recommenders
Focuses on ALS implementation, cold-start handling, evaluation metrics.

Lab 8: Graph Sampling and Streaming
Focuses on reservoir sampling, random walk sampling, streaming PageRank.

Lab 9: Approximate Query Processing
Focuses on sampling guarantees, error bounds, progressive sampling.

Lab 10: Massive Dataset Mining Pipeline
Focuses on complete scalable mining solution from ingestion to insights.

Data Governance, Privacy & Ethics (DS-EL4)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
Data Governance Frameworks and Maturity: Data governance definition and business value, DAMA-DMBOK framework domains, Data governance maturity models (Gartner, IBM, EDM Council), Roles and responsibilities (data stewards, custodians, owners), Data governance council structure, Policy development and enforcement, Data classification schemes and labeling strategies.

Unit 2
Data Quality Management and Lineage: Data quality dimensions (accuracy, completeness, consistency, timeliness, validity, uniqueness), Data profiling techniques and anomaly detection, Data quality rules engines, Master Data Management (MDM) strategies, Data lineage capture (technical, business, operational), Columnar lineage and impact analysis, Data cataloging and metadata management.

Unit 3
Privacy Engineering and Regulations: Privacy by design principles (proactive, embedded, data minimization), GDPR principles and data subject rights, CCPA/CPRA requirements, Privacy Impact Assessments (PIA/DPIA), Data Protection Officers responsibilities, Cross-border data transfers (SCCs, BCRs), Privacy seals and certifications (ISO 27701, TrustArc).

Unit 4
Data Anonymization and Privacy-Preserving Techniques: k-Anonymity, l-Diversity, t-Closeness, Differential privacy fundamentals (ϵ, δ privacy budgets), Local vs. central differential privacy, Synthetic data generation (CTGAN, TVAE), Homomorphic encryption use cases, Secure Multi-Party Computation (SMPC), Federated learning privacy guarantees.

Unit 5
AI Ethics, Fairness, and Responsible Data Practices: Algorithmic bias sources and amplification, Fairness metrics (demographic parity, equal opportunity), Group vs. individual fairness, Counterfactual fairness, Intersectional discrimination, Explainable AI for regulatory compliance, AI ethics frameworks (IEEE, Partnership on AI), Algorithmic accountability and audit trails, Data ethics review boards.

Real-Time Analytics & Stream Processing (DS-EL5)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Stream Processing Fundamentals: Data stream characteristics (infinite, unbounded, out-of-order), Lambda vs. Kappa architectures, Time concepts (event time, processing time, ingestion time, watermarks), Windowing strategies (tumbling, hopping, sliding, session), Late data handling and allowed lateness, Exactly-once vs. at-least-once semantics, Fault tolerance in streaming systems.

Unit 2
Apache Kafka Ecosystem: Kafka architecture (topics, partitions, leaders/followers), Producer/consumer APIs and configurations, Kafka Streams DSL vs. Processor API, KSQL for stream processing SQL, Kafka Connect for data integration, MirrorMaker and cluster federation, Schema Registry and Avro/Protobuf serialization, Exactly-once guarantees with transactions.

Unit 3
Apache Flink Deep Dive: Flink execution model (DataStream/DataSet APIs), Stateful stream processing and keyed state, Checkpointing and savepoints, Event-time processing with watermarks, Complex Event Processing (CEP), Table/SQL API for unified batch/streaming, FlinkML and Gelly for streaming ML, Side outputs and dynamic scaling.

Unit 4
Stream Processing Patterns and Analytics: Pattern detection (match_recognize, CEP patterns), Sessionization and funnel analysis, Real-time aggregations and materialized views, Join strategies (stream-stream, stream-table), Time-windowed joins and interval joins, Anomaly detection (isolation forests, statistical methods), Real-time dashboards (Grafana + Prometheus).

Unit 5
Advanced Topics and Production Systems: Stream-table duality and changelog semantics, Upserts and primary key handling, Change Data Capture (CDC) with Debezium, Streaming ETL/ELT pipelines, Backpressure handling and resource management, Multi-tenancy and resource isolation, Monitoring and observability (metrics, tracing), Deployment strategies (Kubernetes operators).

Practical

Lab 1: Kafka Fundamentals
Focuses on multi-partition topics, producers/consumers, exactly-once transactions.

Lab 2: Kafka Streams Applications
Focuses on windowed aggregations, joins, stateful processing, KSQL queries.

Lab 3: Flink Event-Time Processing
Focuses on watermarks, late data, tumbling/sliding windows, event-time joins.

Lab 4: Flink State and Checkpointing
Focuses on keyed state, operator state, savepoints, fault recovery.

Lab 5: Complex Event Processing
Focuses on Flink CEP patterns, sequence detection, anomaly patterns.

Lab 6: Streaming SQL and Tables
Focuses on Flink Table API, dynamic tables, changelog processing.

Lab 7: CDC and Streaming ETL
Focuses on Debezium + Kafka Connect, Flink CDC connectors.

Lab 8: Real-Time ML Pipelines
Focuses on FlinkML feature extraction, model serving, prediction.

Lab 9: Production Monitoring
Focuses on Prometheus/Grafana dashboards, alerting rules.

Lab 10: Complete Streaming Pipeline
Focuses on end-to-end IoT/telemetry/real-time analytics solution.

Graph Mining & Social Network Analysis (DS-EL6)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Graph Theory Foundations for Analytics: Graph representations (adjacency matrix, edge list, CSR/CSC), Directed/undirected/multigraphs, Basic metrics (degree, density, diameter, average path length), Connected components and strongly connected components, Graph isomorphism and canonical labeling, Bipartite graphs and projections.

Unit 2
Network Centrality and Ranking: Degree centrality variants (in-degree, out-degree, weighted), Closeness centrality and harmonic mean, Betweenness centrality (Brandes algorithm), Eigenvector centrality and Katz matrix, PageRank (power iteration, personalized, topic-sensitive), HITS (authority/hub scores), SALSA improvement.

Unit 3
Community Detection and Clustering: Modularity optimization and Louvain method, Spectral clustering (normalized Laplacian), Label propagation algorithm, Infomap (random walks), Clique percolation method, Overlapping communities (CPM, SLPA), Hierarchical clustering (single-linkage dendrograms), Benchmark networks (LFR generator).

Unit 4
Graph Mining Algorithms: Frequent subgraph mining (Apriori-based, gSpan), Graph motif discovery and network motifs, Graphlet counting and degree distributions, Triangle counting and enumeration, Graph kernels (graphlet, shortest-path, Weisfeiler-Lehman), Subgraph matching and isomorphism testing, Graph alignment.

Unit 5
Social Network Analysis and Applications: Small-world phenomenon and clustering coefficient, Scale-free networks and power-law distributions, Link prediction (common neighbors, Jaccard, Adamic-Adar, Katz, matrix factorization), Influence maximization (greedy algorithm, TIM/CELF), Cascade models (Independent Cascade, Linear Threshold), Signed networks and balance theory, Temporal networks and dynamic analysis.

Practical

Lab 1: Graph Construction and Basic Metrics
Focuses on NetworkX/Graph-tool basics, degree distributions, centrality computation.

Lab 2: PageRank and Ranking Algorithms
Focuses on power iteration implementation, personalized PageRank, convergence analysis.

Lab 3: Community Detection Methods
Focuses on Louvain, spectral clustering, modularity evaluation comparison.

Lab 4: Graph Sampling Techniques
Focuses on random walk, forest fire, node/edge sampling, estimation accuracy.

Lab 5: Frequent Subgraph Mining
Focuses on gSpan, FSG, subgraph enumeration efficiency.

Lab 6: Link Prediction
Focuses on supervised/unsupervised methods, ROC-AUC evaluation, temporal validation.

Lab 7: Influence Maximization
Focuses on greedy algorithm, CELF optimization, simulation-based evaluation.

Lab 8: Graph Neural Networks Basics
Focuses on GCN, GAT, node classification, link prediction tasks.

Lab 9: Dynamic Network Analysis
Focuses on temporal snapshots, streaming algorithms, evolution tracking.

Lab 10: Social Network Analysis Pipeline
Focuses on complete analysis workflow from raw data to insights/visualization.

Advance Elective (Cybersecurity)

Digital Forensics & Incident Response (CY-EL1)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Digital Forensics Fundamentals: Forensic science principles (Locard's exchange principle, scientific method), Digital evidence types and characteristics (volatile, non-volatile, latent), Chain of custody requirements, Legal considerations (admissibility, Daubert standard), Incident response lifecycle (NIST, SANS models), Forensic readiness planning, Roles (first responders, investigators, analysts).

Unit 2
Incident Response Processes: Preparation phase (policies, tools, teams), Detection and analysis (indicators of compromise, triage), Containment strategies (short-term, long-term), Eradication and recovery procedures, Post-incident activities (lessons learned, reporting), Incident classification (levels, severity), Communication protocols and stakeholder management.

Unit 3
Disk and File System Forensics: Acquisition methods (dead disk: dd, FTK Imager; live response), Write-blockers and validation (hashing: MD5, SHA-256), File system analysis (FAT, NTFS, ext4), Timeline reconstruction (MFT, $LogFile, superblocks), File carving (scalpel, foremost), Slack space and unallocated clusters, Metadata extraction (EXIF, MAC times).

Unit 4
Memory and Application Forensics: Memory acquisition (AVML, DumpIt, LiME), Volatility framework (imageinfo, pslist, psscan, hashdump), Process injection detection (hollow process, DLL injection), Registry analysis (userassist, amcache, shimcache), Browser forensics (SQLite artifacts: history, downloads, cache), Email forensics (Outlook PST/OST, MBOX), Anti-forensics techniques (timestomping, overwriting).

Unit 5
Network and Advanced Forensics: Packet capture analysis (Wireshark, tcpdump), Network forensics (IoCs: C2 beacons, anomalies), Log analysis (Windows Event Logs, Sysmon, firewall logs), Cloud forensics (AWS/Azure artifacts), Mobile forensics (Android logical/physical, iOS backups), Malware forensics (static/dynamic analysis), SIEM integration (Splunk, ELK Stack).

Practical

Lab 1: Forensic Acquisition and Hashing
Focuses on disk imaging with FTK Imager/dd, chain of custody forms, hash verification.

Lab 2: File System Analysis
Focuses on Autopsy timeline reconstruction, file carving, metadata extraction.

Lab 3: Incident Response Simulation
Focuses on NIST IR lifecycle, IOC identification, containment planning.

Lab 4: Memory Forensics with Volatility
Focuses on process analysis, network connections, registry hive extraction.

Lab 5: Registry and Application Artifacts
Focuses on userassist, recent documents, browser history extraction.

Lab 6: Network Forensics
Focuses on Wireshark packet analysis, protocol dissection, anomaly detection.

Lab 7: Log Analysis and Correlation
Focuses on Sysmon/Event Logs parsing, timeline correlation.

Lab 8: Mobile Device Forensics
Focuses on ADB backup, Cellebrite/ Oxygen extraction, app data analysis.

Lab 9: Malware Reverse Engineering Basics
Focuses on strings analysis, YARA rules, sandbox detonation.

Lab 10: Complete Incident Investigation
Focuses on end-to-end forensics report generation, court-ready evidence package.

Cloud Security Architecture (CY-EL2)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Cloud Computing and Security Fundamentals: Cloud service models (IaaS, PaaS, SaaS) and deployment models (public, private, hybrid, multi-cloud), Shared responsibility model across providers (AWS, Azure, GCP), Cloud security principles (CIA triad extension, defense-in-depth), Cloud Security Alliance (CSA) Cloud Controls Matrix (CCM), NIST SP 800-53 cloud controls, Threat landscape (misconfiguration, IAM abuse, data exfiltration).

Unit 2
Identity and Access Management (IAM) in Cloud: Identity federation (SAML, OIDC, SCIM), Multi-factor authentication (MFA) enforcement, Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC), AWS IAM policies/service control policies, Azure AD/Entra ID PIM, GCP IAM/service accounts, Least privilege principle, Break-glass accounts and privileged access workstations, Cloud Access Security Brokers (CASB).

Unit 3
Network Security Architecture: Virtual Private Clouds (VPCs) and subnet segmentation, Hub-and-spoke/landing zone architectures, Network Access Control Lists (NACLs) and Security Groups, AWS Transit Gateway/VPC peering, Azure Virtual WAN/Hub-Spoke, GCP Shared VPC/VPC Peering, Web Application Firewalls (WAF), DDoS protection services, Zero Trust Network Access (ZTNA), Micro-segmentation strategies.

Unit 4
Data Protection and Encryption: Data classification and labeling, Encryption at rest (server-side, client-side), Key Management Services (AWS KMS, Azure Key Vault, GCP KMS), Envelope encryption and customer-managed keys, Data Loss Prevention (DLP) policies, Secure object storage (S3 bucket policies, private endpoints), Database encryption (TDE, column-level), Data residency and sovereignty compliance.

Unit 5
Cloud Operations, Monitoring, and Compliance: Cloud Security Posture Management (CSPM), Infrastructure as Code (IaC) security (Terraform Sentinel, CloudFormation Guard), Logging and monitoring (CloudWatch, Azure Monitor, Cloud Logging), SIEM integration and centralized aggregation, Continuous compliance auditing (AWS Config, Azure Policy, GCP Organization Policies), DevSecOps integration (shift-left security), Incident response in cloud environments.

Practical

Lab 1: Cloud IAM Configuration
Focuses on policy creation, RBAC testing, federation setup across AWS/Azure/GCP.

Lab 2: VPC Networking Architecture
Focuses on hub-spoke deployment, NACL/SG rules, private endpoint configuration.

Lab 3: Encryption and KMS Management
Focuses on key policies, envelope encryption, cross-account key access.

Lab 4: Threat Modeling Workshop
Focuses on STRIDE modeling for cloud workloads, risk prioritization.

Lab 5: IaC Security Scanning
Focuses on Checkov/Terraform validate, policy-as-code enforcement.

Lab 6: Monitoring and Alerting
Focuses on CloudWatch dashboards, GuardDuty integration, anomaly detection.

Lab 7: WAF and DDoS Protection
Focuses on rule configuration, rate limiting, bot mitigation.

Lab 8: Compliance Auditing
Focuses on Config rules, policy compliance reports, remediation workflows.

Lab 9: Multi-Cloud Federation
Focuses on cross-cloud IAM, resource sharing, centralized logging.

Lab 10: Secure Architecture Design
Focuses on complete landing zone deployment with security controls.

Malware Analysis & Reverse Engineering (CY-EL3)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Malware Fundamentals and Lifecycle: Malware types classification (viruses, worms, trojans, ransomware, APTs), Infection vectors (phishing, drive-by, supply chain), Malware evolution (polymorphic, metamorphic, fileless), Kill chain models (Cyber Kill Chain, MITRE ATT&CK), Indicators of Compromise (IOCs), Malware triage methodologies, Dynamic vs. static vs. behavioral analysis.

Unit 2
Static Analysis Techniques: PE/ELF file format dissection (headers, sections, imports/exports), Packers and crypters detection (UPX, custom), Strings analysis and YARA rules, Hashing and fuzzy hashing (SSDEEP, imphash), Resource extraction (icons, embedded files), Static disassemblers (Ghidra, IDA Free), Control flow graphs and function identification.

Unit 3
Dynamic Analysis and Sandboxing: Safe execution environments (VMware snapshots, Cuckoo Sandbox), API monitoring (APIMonitor, ProcMon), Behavioral analysis (network, filesystem, registry), Dynamic instrumentation (Pin, DynamoRIO), Memory forensics integration, Anti-analysis evasion detection (timing checks, debugger detection, VM artifacts), Sandbox escape techniques.

Unit 4
Reverse Engineering Core Concepts: Assembly language essentials (x86/x64, ARM), Disassembly and decompilation, Calling conventions (stdcall, fastcall, thiscall), Function prologue/epilogue patterns, Data flow vs. control flow analysis, Symbolic execution fundamentals, Binary patching and code caves, Obfuscation reversal (control flow flattening, opaque predicates).

Unit 5
Advanced Malware Techniques: Rootkit analysis (SSDT hooking, DKOM), Kernel-mode malware, Anti-forensics (timestomping, log wiping), C2 communication protocols (HTTP, DNS tunneling, custom), Ransomware (crypto APIs, payment portals), APT persistence mechanisms, Firmware/BIOS malware, Mobile malware (APK analysis, Frida scripting).

Practical

Lab 1: Malware Triage and Sandbox Setup
Focuses on Cuckoo deployment, sample submission, behavioral reports.

Lab 2: Static Analysis Workshop
Focuses on PEStudio, strings/YARA, import reconstruction.

Lab 3: Dynamic Behavioral Analysis
Focuses on ProcMon/RegShot, network capture, API hooking.

Lab 4: x86 Assembly and Disassembly
Focuses on debuggers (x64dbg, OllyDbg), breakpoint strategies.

Lab 5: Ghidra Reverse Engineering
Focuses on decompiler usage, script automation, function renaming.

Lab 6: Unpacking and Deobfuscation
Focuses on manual unpacking, debugger tricks, entropy analysis.

Lab 7: Rootkit Detection and Analysis
Focuses on GMER/RootkitRevealer, kernel debugging (WinDbg).

Lab 8: Network Malware Analysis
Focuses on Wireshark dissect, C2 emulation, protocol decoding.

Lab 9: Advanced Evasion Reversal
Focuses on anti-VM bypass, packer identification, script malware.

Lab 10: Complete Malware Report
Focuses on IOC extraction, MITRE mapping, remediation recommendations.

Blockchain & Decentralized Identity (CY-EL4)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Blockchain Fundamentals and Cryptography: Distributed ledger technology principles, Cryptographic hash functions (SHA-256, Keccak), Digital signatures (ECDSA, EdDSA), Merkle trees and Patricia tries, Consensus mechanisms (PoW, PoS, PBFT, Tendermint), Byzantine Generals Problem, 51% attacks and chain reorganizations, Blockchain trilemma (decentralization, security, scalability).

Unit 2
Smart Contracts and Ethereum Ecosystem: EVM architecture and gas model, Solidity programming (contracts, modifiers, events), Smart contract vulnerabilities (reentrancy, integer overflow, front-running), Formal verification (Mythril, Slither, Certora), ERC standards (20, 721, 1155, 4337 account abstraction), Layer 2 scaling (Optimism, Arbitrum, zkSync), Oracles (Chainlink, Band Protocol).

Unit 3
Decentralized Identity (DID) Standards: Self-Sovereign Identity (SSI) principles, DID method specification (did:ethr, did:key, did:web), Verifiable Credentials (VC) and Presentations (VP) data model, W3C standards (DID Core, VC Data Model), Decentralized Identifiers resolution, Issuer/Holder/Verifier model, Zero-knowledge proofs for selective disclosure.

Unit 4
DID Ecosystems and Interoperability: DIDComm protocol for secure messaging, DIF standards (Universal Resolver, Sidetree protocol), Hyperledger Indy/Aries for permissioned identity, uPort/SeraphId implementations, Verifiable Data Registries (VDRs), Revocation and expiration handling (status registries), Cross-chain identity (Polkadot, Cosmos IBC).

Unit 5
Blockchain Security and Privacy: 51% attacks mitigation, Eclipse attacks, Smart contract auditing methodologies, Flash loan attacks and MEV, Privacy-preserving techniques (zk-SNARKs, Bulletproofs, Mimblewimble), Confidential transactions, Threshold signatures and MPC wallets, Quantum threats (lattice-based crypto, hash-based signatures), Regulatory compliance (MiCA, travel rule).

Practical

Lab 1: Blockchain Cryptography Basics
Focuses on ECDSA signing/verification, Merkle tree construction, transaction serialization.

Lab 2: Ethereum Smart Contracts
Focuses on Remix IDE development, Hardhat testing, gas optimization.

Lab 3: Smart Contract Vulnerability Exploitation
Focuses on reentrancy demos, Slither static analysis, remediation.

Lab 4: DID Creation and Resolution
Focuses on did:key/ethr generation, Universal Resolver usage.

Lab 5: Verifiable Credentials Issuance
Focuses on VC signing, VP presentation, selective disclosure.

Lab 6: DIDComm Secure Messaging
Focuses on Aries agents, pairwise DIDs, encrypted messages.

Lab 7: Layer 2 Deployment
Focuses on Optimism/Arbitrum bridges, L2 contract deployment.

Lab 8: zk-SNARK Implementation
Focuses on Semaphore/circom circuits, proof generation/verification.

Lab 9: Blockchain Forensics
Focuses on transaction tracing, tainted address analysis.

Lab 10: SSI Application Prototype
Focuses on complete KYC/credential verification dApp.

Mobile & Wireless Security (CY-EL5)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Mobile Platform Security Architecture: Android security model (sandboxing, app permissions, SELinux), iOS security (code signing, sandbox, XNU kernel), App lifecycle and inter-process communication (Intents, AIDL, XPC), Secure boot chain and verified boot, Mobile device management (MDM) frameworks, Root/jailbreak detection bypasses, Hardware security (TEE, ARM TrustZone).

Unit 2
Mobile Application Security: Static analysis (MobSF, QARK), Dynamic analysis (Frida, Objection), Reverse engineering (APKTool, Hopper, IDA), Common vulnerabilities (insecure storage, hard-coded secrets, improper IPC), OWASP Mobile Top 10, Certificate pinning bypass, Runtime hooking and SSL pinning evasion, Binary protection (obfuscation, RASP).

Unit 3
Wireless Network Security Protocols: 802.11 wireless fundamentals (WEP, WPA, WPA2, WPA3), Wi-Fi attack vectors (evil twin, KRACK, PMKID), Enterprise security (802.1X/EAP methods: PEAP, EAP-TLS), Bluetooth Low Energy (BLE) security (pairing modes, Just Works vulnerability), BLE attacks (knob turning, spoofing), Zigbee/IoT protocol security, Wireless IDS/IPS.

Unit 4
Cellular Network Security: GSM/CDMA vulnerabilities (A5/1 cracking, IMSI catchers), LTE/4G security (EPS-AKA, SUCI encryption), 5G security architecture (SUPI concealment, SEAF authentication), Mobile network attacks (SS7 exploitation, Diameter signaling), IMS/SIP security, Roaming security risks, SIM toolkit and OTA attacks, eSIM security.

Unit 5
Mobile Threat Landscape and Forensics: Mobile malware evolution (banking trojans, spyware), Phishing and social engineering vectors, MDM bypass and enterprise compromise, Mobile forensics (logical, file system, physical extraction), JTAG/ISP chip-off methods, Anti-forensics (app shredders, secure erase), Mobile security frameworks (MASA, GSMA NESAS).

Practical

Lab 1: Android App Reverse Engineering
Focuses on APK decompilation, manifest analysis, permission abuse.

Lab 2: iOS App Analysis
Focuses on IPA cracking, runtime hooking with Frida/Cycript.

Lab 3: Wi-Fi Security Assessment
Focuses on Aircrack-ng suite, WPA2 cracking, evil twin setup.

Lab 4: Bluetooth/BLE Exploitation
Focuses on GATT attacks, Ubertooth, gattacker, BLE recon.

Lab 5: Mobile Dynamic Analysis
Focuses on MobSF automation, Objection scripting, SSL pinning bypass.

Lab 6: Cellular Network Recon
Focuses on OpenBTS/IMSI catcher simulation, SS7 basics.

Lab 7: Root Detection Bypass
Focuses on MagiskHide, Frida anti-root scripts.

Lab 8: Mobile Forensics Acquisition
Focuses on Cellebrite/Grayshift, ADB backup analysis.

Lab 9: Wireless IDS Deployment
Focuses on Kismet, Snort wireless rules.

Lab 10: Mobile Pentest Report
Focuses on complete app/network assessment with remediation.

Cyber Law, Risk & Compliance (CY-EL6)

← Back to Scheme

Credits: 4 (4-0-0)

Course Content

Unit 1
Cybersecurity Legal Framework: Information Technology Act 2000 (amendments), Section 43/43A/65/66/66A-66F offenses, Digital Personal Data Protection Act 2023 (DPDP), Reasonable security practices and SOPs, Cyber Appellate Tribunal, Electronic signatures and digital certificates, Contract law in cyberspace, Jurisdiction issues (cross-border cybercrimes).

Unit 2
Data Protection and Privacy Laws: GDPR extraterritorial applicability, Legitimate interest vs. consent, Data Protection Impact Assessments (DPIA), Data processor obligations, Breach notification timelines (72 hours), Data Protection Officer (DPO) mandate, Privacy by Design/Default, India data localization requirements (critical personal data), Cross-border transfer mechanisms (SCCs, adequacy decisions).

Unit 3
Cybercrime Investigation and Evidence: Indian Evidence Act Section 65B (electronic records), First Information Report (FIR) for cybercrimes, Chain of custody preservation, Digital evidence admissibility, CERT-In incident reporting mandate (6-hour timeline), Forensic readiness planning, Mutual Legal Assistance Treaties (MLAT), Budapest Convention on Cybercrime.

Unit 4
Enterprise Risk Management Frameworks: NIST Cybersecurity Framework (Identify, Protect, Detect, Respond, Recover), ISO 27001:2022 controls and Annex A domains, Risk assessment methodologies (qualitative/quantitative), Threat modeling (STRIDE, PASTA), Risk treatment plans (avoid, mitigate, transfer, accept), Key Risk Indicators (KRIs), Third-party risk management, Supply chain security.

Unit 5
Compliance Auditing and Governance: PCI-DSS requirements (cardholder data environment), SOC 2 Type II criteria (security, availability, processing integrity), HIPAA/HITECH for healthcare, COBIT 2019 governance framework, GRC platforms integration, Audit logging and retention policies, Continuous compliance monitoring, Regulatory reporting (SEBI cybersecurity framework), Board-level cyber governance responsibilities.

Advance Elective (Robotics)

Advanced Control Systems (RB-EL1)

← Back to Scheme

Credits: 4 (3-1-0)

Theory

Unit 1
State-Space Analysis and Design: State-space representation (controllable canonical, observable canonical, diagonal forms), Controllability and observability tests (Popov-Belevitch-Hautus eigenvalues), Pole placement by state feedback, Ackermann's formula, State observers (Luenberger full-order/reduced-order), Separation principle, Riccati equation fundamentals.

Unit 2
Optimal Control Systems: Linear Quadratic Regulator (LQR) design (finite/infinite horizon), Algebraic Riccati Equation (ARE) solution, Steady-state LQR and optimal gain computation, Linear Quadratic Gaussian (LQG) control, Kalman filtering for state estimation, Loop transfer recovery (LQR/LQG trade-offs), Disturbance decoupling and regulation.

Unit 3
Robust Control Design: H-infinity control fundamentals (small gain theorem, bounded real lemma), Mixed-sensitivity optimization, mu-synthesis for structured uncertainty, Loop-shaping design procedure, Robust stability/performance margins, Kharitonov theorem for interval plants, Quantitative feedback theory (QFT) basics.

Unit 4
Adaptive Control Systems: Model Reference Adaptive Control (MRAC: MIT rule, Lyapunov-based), Self-tuning regulators (explicit/implicit), Parameter convergence analysis, Persistent excitation conditions, Dead-zone modification for noise robustness, Gain scheduling strategies, Switching and supervisory control, Adaptive pole placement.

Unit 5
Nonlinear and Modern Control: Feedback linearization (input-state, input-output), Sliding mode control (reaching law, boundary layer), Backstepping design for strict-feedback systems, Lyapunov stability analysis (direct/indirect methods), Passivity-based control, Model Predictive Control (MPC) fundamentals, Constrained optimization (QP solvers).

Practical/Simulations

Session 1: State-Space Modeling
Focuses on MATLAB state-space conversion, controllability/observability analysis.

Session 2: LQR/LQG Controller Design
Focuses on Riccati equation solution, Kalman filter tuning, simulation validation.

Session 3: H-Infinity Synthesis
Focuses on hinfsyn toolbox, mixed-sensitivity design, robust stability margins.

Session 4: Adaptive Control Simulation
Focuses on MRAC implementation, parameter tracking, robustness testing.

Session 5: Nonlinear Control Design
Focuses on feedback linearization, sliding mode simulation, Lyapunov validation.

Session 6: MPC Implementation
Focuses on quadprog solver, receding horizon control, constraint handling.

Session 7: Robust Control Applications
Focuses on QFT toolbox, uncertainty modeling, performance specification.

Session 8: Real-Time Control Experiments
Focuses on dSPACE/Simulink Real-Time, hardware-in-loop validation.

Session 9: System Identification Integration
Focuses on subspace methods feeding control design.

Session 10: Control System Performance Analysis
Focuses on comparative benchmarking, sensitivity functions, robustness metrics.

Deep Learning for Robot Perception (RB-EL2)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
CNN Architectures for Visual Perception: Convolutional layers and feature hierarchies, AlexNet to ResNet/VGG, Residual learning and skip connections, Depthwise separable convolutions (MobileNet), Dilated convolutions (DeepLab), Grouped convolutions (ShuffleNet), Attention mechanisms (CBAM, SENet), EfficientNet scaling and compound coefficients.

Unit 2
Object Detection and Segmentation: Two-stage detectors (R-CNN, Fast/Faster R-CNN), One-stage detectors (YOLO v3-v8, SSD, RetinaNet), Anchor-free methods (CornerNet, FCOS), Instance segmentation (Mask R-CNN), Semantic segmentation (FCN, U-Net, DeepLab series), Panoptic segmentation (Panoptic FPN), Real-time detection trade-offs.

Unit 3
3D Perception and Depth Estimation: Monocular depth estimation (MiDaS, Depth Anything), Stereo vision and disparity maps, LiDAR point cloud processing (voxelization, PointNet/PointNet++), Multi-view fusion (MVSNet), 3D object detection (VoxelNet, SECOND, PointRCNN), BEV representations (Lift-Splat-Shoot), Sensor fusion (camera-LiDAR-Radar).

Unit 4
Visual Odometry and SLAM: Feature-based VO (ORB-SLAM, DSO), Learning-based VO (DeepVO, VIO), Direct methods vs. indirect, Bundle adjustment optimization, Loop closure detection, Visual-inertial odometry (OKVIS, VINS-Mono), Neural radiance fields (NeRF) for dense reconstruction, Semantic SLAM integration.

Unit 5
Robust Perception and Domain Adaptation: Sim-to-real transfer (CycleGAN, domain randomization), Uncertainty estimation (Bayesian NNs, Monte Carlo dropout), Test-time adaptation, Continual learning for perception, Multi-modal fusion (BEVFusion, UniFusion), Edge deployment optimization (TensorRT, ONNX Runtime), Anomaly detection in perception pipelines.

Practical

Lab 1: CNN Implementation from Scratch
Focuses on PyTorch convolutional layers, training on CIFAR/ImageNet subsets.

Lab 2: Object Detection Pipeline
Focuses on YOLOv8/MMDetection, custom dataset annotation/training.

Lab 3: Semantic Segmentation
Focuses on DeepLabv3/HRNet, Cityscapes/KITTI datasets.

Lab 4: 3D Point Cloud Processing
Focuses on Open3D/PointNet, LiDAR segmentation/detection.

Lab 5: Visual Odometry
Focuses on ORB-SLAM3 implementation, KITTI/TUM RGB-D evaluation.

Lab 6: Sensor Fusion
Focuses on nuScenes dataset, camera-LiDAR fusion networks.

Lab 7: Domain Adaptation
Focuses on CycleGAN sim-to-real, BDD100K/Syn2Real experiments.

Lab 8: Real-Time ROS2 Integration
Focuses on perception nodes, NVIDIA Jetson deployment.

Lab 9: Uncertainty Quantification
Focuses on evidential deep learning, out-of-distribution detection.

Lab 10: End-to-End Perception System
Focuses on complete robotic pipeline with custom hardware.

Swarm Intelligence & Multi-Robot Systems (RB-EL3)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Swarm Intelligence Foundations: Biological inspiration (ants, bees, birds, fish schooling), Boids model (separation, alignment, cohesion), Reynolds rules and flocking behavior, Particle Swarm Optimization (PSO: velocity update, inertia weight), Ant Colony Optimization (ACO: pheromone trails, evaporation), Artificial Bee Colony (ABC), Swarm stability analysis.

Unit 2
Multi-Robot Communication and Coordination: Communication topologies (centralized, decentralized, hybrid), Consensus algorithms (average, leader election), Task allocation (CBBA, auction-based), Formation control (virtual structures, leader-follower), Potential field methods (artificial potentials, collision avoidance), Graph-theoretic approaches (Riemannian manifolds).

Unit 3
Swarm Robotics Algorithms: Coverage control (Voronoi partitions, Lloyd algorithm), Foraging and patrolling strategies, Flocking with obstacles, Self-assembly and shape formation, Density control and deployment, Scalable coordination (local interactions, emergent behavior), Scalability analysis (communication overhead, convergence time).

Unit 4
Distributed Estimation and Mapping: Multi-robot SLAM (centralized, decentralized, DDF-SLAM), Data association (JPDDA, Murty's algorithm), Cooperative localization (EKF/UKF fusion), Relative pose estimation (anchor nodes), Graph SLAM optimization (pose graph, factor graphs), Outlier rejection and robust estimation.

Unit 5
Advanced Topics and Applications: Heterogeneity in swarms (specialized roles), Fault tolerance and resilience, Learning in multi-agent systems (MARL, QMIX), Swarm reinforcement learning, Real-world applications (search-rescue, agriculture, warehouse automation), Hardware platforms (Khepera, Kilobot, TurtleBot swarm), Standardization (ROS2 multi-robot, MQTT/IoT protocols).

Practical

Lab 1: Boids Flocking Simulation
Focuses on Unity/Gazebo implementation, parameter tuning.

Lab 2: Particle Swarm Optimization
Focuses on PyTorch/DEAP for robotics path planning.

Lab 3: Ant Colony Path Finding
Focuses on graph-based ACO for multi-robot navigation.

Lab 4: Formation Control
Focuses on leader-follower, virtual structure simulations.

Lab 5: Task Allocation (CBBA)
Focuses on auction algorithm implementation, conflict resolution.

Lab 6: Multi-Robot Coverage
Focuses on Voronoi-based area partitioning in Gazebo.

Lab 7: Consensus and Synchronization
Focuses on average consensus, phase synchronization.

Lab 8: Cooperative SLAM
Focuses on ROS2 multi-robot mapping, loop closure.

Lab 9: Swarm Hardware Experiment
Focuses on TurtleBot3 swarm, real-world flocking.

Lab 10: Complete Swarm Application
Focuses on search-and-rescue scenario with 10+ robots.

Human-Robot Interaction (HRI) (RB-EL4)

← Back to Scheme

Credits: 4 (3-1-0)

Theory

Unit 1
HRI Fundamentals and Models: HRI paradigms (physical, teleoperation, shared autonomy), Levels of autonomy (SAE 0-5 for robotics), Cognitive models (NASA TLX workload, Situation Awareness), Trust in automation (Lee & See model), Uncanny valley effect, Robot appearance (anthropomorphic, zoomorphic, functional), Social robot design principles.

Unit 2
Natural User Interfaces: Multimodal interaction (speech, gesture, gaze, haptics), Speech recognition (ASR: HMM, end-to-end DNNs), Natural Language Understanding (NLU: intent classification, slot filling), Dialogue management (POMDP, rule-based), Gesture recognition (MediaPipe, OpenPose), Gaze estimation and attention modeling.

Unit 3
Social and Affective HRI: Emotion recognition (facial FER, physiological signals: EDA, HRV), Expressive behavior generation (gesture timing, facial animation), Rapport building strategies, Theory of Mind in robots, Personalization and user modeling, Cultural adaptation in HRI, Ethical considerations (deception, privacy, attachment).

Unit 4
Shared Autonomy and Assistive Robotics: HRI in manipulation (shared control, intent prediction), Haptic feedback and teleoperation (passivity-based control), Assistive technologies (exoskeletons, wheelchairs), Legibility and predictability in motion, Human-aware navigation (social costs, pedestrian avoidance), Safety standards (ISO 13482, UL 1740).

Unit 5
HRI Evaluation and Applications: Experimental design (within/between subjects, counterbalancing), Metrics (success rate, time-to-task, user satisfaction SUS/NASA-TLX), Field vs. lab studies, HRI in domains (healthcare, education, manufacturing, service robots), Emerging trends (AR/VR HRI, brain-computer interfaces), Accessibility and inclusive design.

Practical/Simulations

Session 1: ROS2 Human Detection
Focuses on YOLO + OpenCV for person tracking.

Session 2: Speech Recognition Interface
Focuses on Whisper ASR, intent classification with HuggingFace.

Session 3: Gesture Recognition
Focuses on MediaPipe hands/pose, custom classifier training.

Session 4: Emotion Recognition
Focuses on FER datasets, lightweight CNN deployment.

Session 5: Dialogue System
Focuses on Rasa NLU + dialogue policies.

Session 6: Shared Autonomy Demo
Focuses on admittance control, human force input.

Session 7: Human-Aware Navigation
Focuses on social navigation layers in ROS Nav2.

Session 8: HRI User Study
Focuses on SUS questionnaire, A/B testing design.

Session 9: Multimodal Integration
Focuses on fusion of speech/gesture/gaze inputs.

Session 10: HRI Application Prototype
Focuses on assistive robot with voice/gesture control.

Advanced Motion Planning (RB-EL5)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Geometric Motion Planning: Configuration space (C-space obstacles), Workspace vs. C-space, Collision detection (GJK algorithm, bounding volume hierarchies), Sampling-based planners (PRM: probabilistic roadmap, lazy evaluation), RRT family (RRT-Connect, RRT*: asymptotic optimality), Informed sampling (elliptical projections).

Unit 2
Optimization-Based Planning: Trajectory optimization (CHOMP, STOMP, TrajOpt), Convex optimization (convex MPC, SOCP), Differential flatness for drones/manipulators, Whole-body planning (centroidal dynamics), Contact-rich manipulation (MuJoCo, QP trajectory optimization), Real-time receding horizon control.

Unit 3
Learning for Motion Planning: Imitation learning (BC, DAGGER, GAIL), Reinforcement learning planners (AlphaPilot, Dreamer), Neural motion policies (MPNet, MotionPolicy), Scene-aware planning (LEGO, SPN), Foundation models for planning (RT-2, Octo), Fast planning via regression (Latent Planners), Sample efficiency techniques.

Unit 4
Multi-Robot and Human-Aware Planning: Decentralized planning (conflict-based search CBS), Priority planning, Velocity obstacles (ORCA), Human-robot cohabitation (social costs, reciprocal collision avoidance), Dynamic environments (DWA, TEb), Legible and predictable trajectories for HRI.

Unit 5
Advanced Topics: Nonholonomic planning (dubins paths, Reeds-Shepp curves), Underactuated systems (acrobot, pendubot), Legged locomotion (MPC, WBF), Soft robotics planning, Uncertainty-aware planning (POMDPs, belief space), Integrated perception-planning (VPG, PEPLAN), Real-time certification (iCORA, anytime algorithms).

Practical

Lab 1: Sampling-Based Planners
Focuses on OMPL implementation (RRT*, PRM), MoveIt! integration.

Lab 2: Optimization Trajectory Generation
Focuses on TOPP-RA, CHOMP in Drake/MuJoCo.

Lab 3: Manipulation Planning
Focuses on Panda arm pick-place, collision-free IK.

Lab 4: Quadrotor Motion Planning
Focuses on minimum-snap trajectories, PX4 SITL.

Lab 5: Multi-Robot Coordination
Focuses on CBS/ORCA in Gazebo swarm.

Lab 6: Neural Planners
Focuses on MPNet fine-tuning, scene-conditioned planning.

Lab 7: Human-Aware Navigation
Focuses on Nav2 social layers, pedestrian datasets.

Lab 8: Legged Robot Locomotion
Focuses on ANYmal MPC, contact sequence optimization.

Lab 9: Real-Time Hardware Demo
Focuses on Jetson deployment, latency optimization.

Lab 10: Complete Planning Pipeline
Focuses on perception-motion-action loop validation.

Manipulation & Grasping Mechanisms (RB-EL6)

← Back to Scheme

Credits: 4 (3-0-2)

Theory

Unit 1
Rigid Body Kinematics and Statics: Forward/inverse kinematics for manipulators, Denavit-Hartenberg (DH) parameters, Product of exponentials (POE) formulation, Jacobian matrices (linear/angular velocity), Singularity analysis (manipulability ellipsoid), Static equilibrium (wrench closure), Force/torque relationships.

Unit 2
Grasp Planning and Analysis: Grasp quality metrics (force closure, form closure), Grasp wrench space (GWS) and manipulability, 2D/3D grasp synthesis (independent contact regions), Friction cone constraints (LMI formulation), Optimal grasp planning (convex optimization), Parallel jaw vs. multifinger grasps, Underactuated hands.

Unit 3
Dexterous Manipulation: In-hand manipulation (finger gaiting, regrasp), Rolling contact kinematics, Part mating (chamferless assembly), Fixturing and form closure, Hybrid position/force control, Impedance/admittance control, Variable admittance for compliant manipulation.

Unit 4
Contact Dynamics and Control: Coulomb friction model (stick-slip), Complementarity formulations (LCP, NCP), Time-stepping integrators (MuJoCo, Drake), Hybrid zero dynamics for manipulation, Tactile sensing and feedback, Slip detection and grip adjustment, Soft robotics grasping (continuum models).

Unit 5
Learning-Based Grasping: Data-driven grasp prediction (DexNet, GraspNet), 6-DoF grasp detection (deep CNNs, PointNet++), Sim-to-real transfer (domain randomization, system ID), Reinforcement learning for manipulation (QT-Opt, RLBench), Vision-based servoing (IBVS, 6D pose estimation), Model-based planning (MPC with contacts).

Practical

Lab 1: Manipulator Kinematics
Focuses on DH modeling, Jacobian computation, singularity analysis.

Lab 2: Grasp Quality Metrics
Focuses on GWS computation, force closure testing.

Lab 3: Parallel Jaw Grasping
Focuses on MoveIt! grasp planning, Panda gripper simulation.

Lab 4: Dexterous Hand Simulation
Focuses on Shadow Hand, finger gaiting in Gazebo.

Lab 5: Contact-Rich Manipulation
Focuses on peg insertion, force control tuning.

Lab 6: Vision-Based Grasping
Focuses on DexNet-4, RGB-D grasp detection.

Lab 7: Tactile Sensing Integration
Focuses on DigiTac sensors, slip detection algorithms.

Lab 8: RL Manipulation Policies
Focuses on Stable-Baselines3, custom gym environments.

Lab 9: Real Robot Experiment
Focuses on Franka Panda, pick-and-place validation.

Lab 10: Complete Manipulation Pipeline
Focuses on perception-planning-execution with uncertainty handling.

Exploratory Minor (Level 1)

Introduction to Bioinformatics (MN-BIO)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
The Language of Life and Its Digital Representation: The Central Dogma of Molecular Biology (DNA → RNA → Protein); Nucleotides, Codons, and the Genetic Code as an information encoding scheme; Biological sequences as discrete strings over a finite alphabet ($\Sigma = \{A, T, G, C\}$); FASTA and FASTQ file formats as data structures for sequence storage; Introduction to biological databases: NCBI, UniProt, and the PDB as large-scale structured repositories; Querying and retrieving biological records programmatically.

Unit 2
Sequence Alignment — The Core Computational Problem: The biological motivation for sequence comparison: finding functional and evolutionary similarity; Pairwise alignment: global alignment (Needleman-Wunsch) and local alignment (Smith-Waterman) as dynamic programming problems; Scoring matrices: PAM and BLOSUM as probabilistic substitution models; Gap penalties and their effect on alignment quality; Heuristic alignment with BLAST: seed-and-extend strategy, E-values, and statistical significance; Applications in identifying homologous genes and annotating unknown sequences.

Unit 3
Genome Assembly and Sequence Graphs: The genome sequencing problem: reads, coverage, and the assembly challenge; Overlap-Layout-Consensus (OLC) paradigm for long-read assembly; De Bruijn graphs as the foundational data structure for short-read assembly: k-mers, nodes, and edges; Eulerian path formulation of the assembly problem; Challenges: repeats, sequencing errors, and heterozygosity; Introduction to reference genomes and read mapping as an alternative to de-novo assembly.

Unit 4
Gene Finding and Biological Pattern Recognition: The gene prediction problem: identifying coding regions within a raw genome sequence; Hidden Markov Models (HMMs) as the canonical tool for gene structure modeling: states for exons, introns, and intergenic regions; The Viterbi algorithm for decoding the most likely gene structure; Motif finding: representing regulatory signals (promoters, binding sites) as Position Weight Matrices (PWMs); Searching for motifs using information-theoretic scoring; Introduction to sequence logos as a visualization of motif conservation.

Unit 5
Molecular Phylogenetics and Evolutionary Computation: Phylogenetic trees as data structures representing evolutionary history: leaves, internal nodes, branch lengths; Distance-based tree construction: UPGMA and Neighbor-Joining algorithms; Character-based methods: Maximum Parsimony as a combinatorial optimization problem; Bootstrapping as a statistical validation technique for tree topology; Multiple Sequence Alignment (MSA) with ClustalW and MUSCLE as a prerequisite for phylogenetics; Biological insight: tracing the origin of pathogens and studying gene family evolution.

Biomedical Signals & Systems (MN-MED)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
The Body as a Data Source: Biomedical signals as time-series data generated by physiological processes: electrical, mechanical, acoustic, and chemical origins; Classification of biomedical signals: deterministic vs. stochastic, continuous vs. discrete, stationary vs. non-stationary; Key signal types and their clinical context: ECG (cardiac electrical activity), EEG (brain waves), EMG (muscle activity), PPG (blood volume pulse), and PCG (heart sounds); Sampling and digitization: the Nyquist criterion applied to physiological bandwidths; Signal quality metrics: SNR, baseline wander, and motion artifacts as noise modeling problems; Introduction to standard biomedical data formats: EDF, HL7, and DICOM headers for time-series records.

Unit 2
Signal Processing Foundations for Physiological Data: The Fourier Transform as the bridge between time-domain waveforms and frequency-domain physiology: interpreting spectral content of ECG and EEG; Short-Time Fourier Transform (STFT) and spectrograms for non-stationary signals; Discrete Wavelet Transform (DWT) for multi-resolution analysis: decomposing EEG into delta, theta, alpha, beta, and gamma bands; Digital filtering: FIR and IIR filter design for artifact removal (powerline interference at 50/60 Hz, baseline wander); Convolution as the computational primitive of linear filtering; Feature extraction from time-domain (mean, variance, zero-crossing rate) and frequency-domain (band power, spectral entropy) for downstream analysis.

Unit 3
Physiological Rhythms and Event Detection: Cardiac cycle anatomy and the PQRST complex as a structured pattern in ECG; Pan-Tompkins algorithm for QRS detection: bandpass filtering, differentiation, squaring, and moving-window integration as a classical signal processing pipeline; R-R interval time series and Heart Rate Variability (HRV) as a window into the autonomic nervous system; Seizure detection in EEG: energy-based and threshold-based detectors; Apnea detection from respiratory signals using autocorrelation and spectral analysis; The general paradigm: segmentation → feature extraction → event classification as a reusable computational template across all biosignal types.

Unit 4
Biosensor Systems and Data Acquisition Pipelines: Transducer principles: converting physiological quantities into electrical signals (electrodes, pressure transducers, optical sensors); Instrumentation amplifiers, differential amplification, and common-mode rejection as analog front-end design concepts; Analog-to-Digital conversion pipeline: anti-aliasing filters, ADC resolution, and dynamic range; Wearable sensor systems: IMUs, pulse oximeters, and continuous glucose monitors as constrained embedded data acquisition devices; Data streaming and buffering: circular buffers and real-time processing constraints; Introduction to IoT-based patient monitoring architectures: edge preprocessing, MQTT-based transmission, and cloud aggregation.

Unit 5
Clinical Decision Support and Signal Intelligence: The clinical decision support system (CDSS)

as a software pipeline: signal ingestion → preprocessing → feature engineering → classification → alert generation; Arrhythmia classification from ECG as a canonical supervised learning problem: feature-based classifiers vs. end-to-end approaches; Stress and fatigue detection from HRV and EEG features; Alarm fatigue in ICUs: the false-positive problem and precision-recall tradeoffs in safety-critical systems; Introduction to interoperability standards: FHIR as a RESTful API standard for exchanging clinical data; Regulatory and ethical constraints: FDA classification of Software as a Medical Device (SaMD) and data privacy under HIPAA.

Human-Computer Interaction (HCI) (MN-DES)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Human Perception and the Cognitive Model of Interaction: HCI as an intersection of cognitive psychology, ergonomics, and software engineering; The human information processing model: sensory memory, working memory (Miller's Law: 7 ± 2 chunks), and long-term memory as constraints on interface design; Perception principles: Gestalt laws (proximity, similarity, continuity, closure) as the perceptual grammar of visual layout; Fitts' Law as a predictive model of pointing time: $T = a + b \log_2\left(\frac{2D}{W}\right)$ and its application to button sizing and menu placement; Hick's Law: decision time as a function of the number of choices; Cognitive load theory and its implications for interface complexity; Mental models: the gap between the user's conceptual model and the system's implementation model.

Unit 2
Interaction Design Paradigms and UI Architectures: Evolution of interaction paradigms: command-line, WIMP (Windows, Icons, Menus, Pointer), touch, voice, and gesture interfaces; The Model-View-Controller (MVC) pattern reframed as a separation between data, presentation, and user intent; Event-driven programming as the computational backbone of all interactive systems: event loops, listeners, and callbacks; Affordances, signifiers, and feedback as the vocabulary of interface semantics (Norman's design principles); Information architecture: hierarchy, hub-and-spoke, and sequential navigation models; Design patterns for UI: card layouts, progressive disclosure, and contextual menus as reusable interaction solutions.

Unit 3
User Research and Requirements Engineering for Interfaces: The user-centered design (UCD) process: discover → define → design → evaluate as an iterative loop; Qualitative research methods: contextual inquiry, semi-structured interviews, and think-aloud protocols as data collection techniques; Quantitative methods: surveys, A/B testing, and log analysis for behavioral data at scale; Personas as abstract data models representing user archetypes; User journey maps as state-machine diagrams of the user's experience over time; Task analysis: Hierarchical Task Analysis (HTA) as a tree decomposition of user goals into subtasks; Translating research findings into functional and non-functional requirements.

Unit 4
Prototyping, Wireframing, and Usability Evaluation: The fidelity spectrum: paper sketches, wireframes, and interactive prototypes as progressively higher-cost representations; Information hierarchy and visual weight: typography scale, color contrast ratios (WCAG 2.1 AA: 4.5:1), and whitespace as layout algorithms; Usability heuristics: Nielsen's 10 principles as a checklist-based static analysis of interfaces; Usability testing: task completion rate, error rate, and time-on-task as quantitative usability metrics; System Usability Scale (SUS) as a standardized psychometric instrument; Eye-tracking and heatmaps as data visualization of visual attention patterns; Accessibility as a correctness constraint: ARIA roles, keyboard navigability, and screen reader compatibility.

Unit 5
Intelligent and Adaptive Interfaces: Recommender systems as a form of proactive UI: col-

laborative filtering and content-based filtering as personalization engines; Adaptive interfaces: rule-based and ML-driven systems that modify layout, content, or interaction modality based on user context; Conversational UI: dialogue state machines as the control flow model for chatbots and voice assistants; Affective computing: inferring user emotional state from facial expression, keystroke dynamics, and physiological signals to adapt interface behavior; Dark patterns as adversarial HCI: taxonomizing manipulative design and their detection as a classification problem; Ethical dimensions of persuasive technology: nudging, attention economics, and the responsibility of interface engineers.

IoT for Industry (MN-MFG)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
The Industrial IoT Stack and Smart Factory Architecture: Industry 4.0 as the convergence of physical production systems with digital computation: cyber-physical integration, connectivity, and data-driven decision making; The IIoT reference architecture: field devices → edge gateways → fog layer → cloud platform as a layered data pipeline; Operational Technology (OT) vs. Information Technology (IT): SCADA systems, PLCs, and DCS as the legacy computational infrastructure of the factory floor; Industrial sensors and actuators: temperature, pressure, vibration, flow, and proximity sensors as the raw data sources of manufacturing processes; Sensor data as time-series streams: sampling rates, resolution, and the tradeoff between data fidelity and transmission cost; Introduction to industrial communication protocols: Modbus, OPC-UA, and PROFINET as the fieldbus standards enabling device interoperability.

Unit 2
Embedded Systems and Edge Computing in Manufacturing: Microcontrollers vs. microprocessors as the hardware dichotomy at the network edge: real-time constraints, determinism, and the role of RTOSes (FreeRTOS); Embedded firmware architecture: interrupt-driven programming, hardware abstraction layers (HAL), and peripheral drivers for ADC, UART, SPI, and I2C; Energy harvesting and power management: duty cycling, sleep modes, and the energy budget of a battery-operated field sensor; Edge computing rationale: latency reduction, bandwidth conservation, and local closed-loop control that cannot tolerate cloud round-trip delays; Edge inference: deploying quantized ML models (TensorFlow Lite, ONNX Runtime) on resource-constrained hardware for on-device anomaly detection; Containerization at the edge: Docker and Kubernetes-based orchestration (K3s) for managing heterogeneous edge device fleets.

Unit 3
Industrial Networking, Protocols, and Data Ingestion: Wireless technologies for industrial environments: ISA-100.11a, WirelessHART, LoRaWAN, and 5G NR-U as a spectrum of range-bandwidth-latency tradeoffs; Time-Sensitive Networking (TSN): IEEE 802.1 extensions for deterministic Ethernet in motion control and robotics; MQTT as the dominant publish-subscribe messaging protocol for IIoT: broker architecture, QoS levels, and retained messages; AMQP and Kafka for high-throughput industrial event streaming: topics, partitions, and consumer groups as the data ingestion backbone; Data serialization for constrained environments: Protocol Buffers and CBOR as compact binary alternatives to JSON; OPC-UA PubSub as the unified information model enabling semantic interoperability across heterogeneous vendor equipment.

Unit 4
Industrial Data Processing and Condition Monitoring: Time-series databases for manufacturing data: InfluxDB and TimescaleDB as purpose-built storage engines for high-frequency sensor streams; Stream processing for real-time manufacturing analytics: Apache Flink and Spark Streaming for sliding-window aggregations, threshold alerting, and event detection; Vibration signal analysis for rotating machinery health: FFT-based spectral analysis, envelope detection, and bearing fault frequency identification (BPFO, BPFI); Statistical Process Control (SPC): control charts (X-bar, R-chart) as an online monitoring algorithm for detecting process drift; Remaining Useful Life (RUL) estimation: degradation modeling from run-to-failure datasets

as a regression problem; Overall Equipment Effectiveness (OEE) as a KPI computed from availability, performance, and quality data streams.

Unit 5

Industrial Security, Standards, and the Road to Autonomy: The expanded attack surface of connected manufacturing: IT/OT convergence risks, Purdue Model segmentation, and the Stuxnet case as a landmark in ICS cybersecurity; IEC 62443 as the industrial cybersecurity standard: security levels, zones, and conduits as a network segmentation framework; Secure boot, firmware signing, and certificate-based device identity as the hardware root of trust for IIoT devices; Predictive maintenance as a business case: comparing reactive, preventive, and predictive maintenance strategies through cost and downtime modeling; Introduction to autonomous manufacturing: closed-loop feedback from sensor data to actuator commands without human intervention; Digital thread concept: the unbroken data linkage from product design through manufacturing to field operation as the informational backbone of smart factories.

Computational Materials Science (MN-MAT)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1

Matter as Data: Representing Materials Computationally: The materials science length scale hierarchy: electrons → atoms → microstructure → macroscopic properties as a multiscale computational challenge; Crystal structures as periodic data structures: lattice vectors, basis atoms, unit cells, and the Bravais lattice classification; Crystallographic information files (CIF) as the standard schema for storing atomic structure data; Reciprocal space and the Brillouin zone as the Fourier dual of the crystal lattice: intuition for why periodicity enables tractable computation; Point defects, dislocations, and grain boundaries as structural irregularities that dominate material behavior: representing them as perturbations to a reference periodic system; Introduction to open materials databases: the Materials Project, AFLOW, and OQMD as large-scale, queryable repositories of computed material properties.

Unit 2

Atomistic Simulation: Molecular Dynamics and Monte Carlo: Classical Molecular Dynamics (MD) as a numerical ODE integration problem: the Verlet and velocity-Verlet algorithms for propagating Newton's equations of motion; Interatomic potentials as the force-field approximation: Lennard-Jones, embedded atom method (EAM), and Tersoff potentials as parameterized energy functions replacing quantum mechanics; Periodic boundary conditions as the computational trick that simulates bulk material with a finite simulation box; Thermostats and barostats: the Nosé-Hoover and Parrinello-Rahman algorithms for sampling the NVT and NPT ensembles; Monte Carlo methods for equilibrium sampling: the Metropolis-Hastings algorithm applied to atomic configuration space; Structural analysis of MD trajectories: radial distribution functions, mean squared displacement, and Voronoi tessellation as the post-processing toolkit.

Unit 3

Quantum Mechanical Simulation and Density Functional Theory: The quantum mechanical many-body problem: the electronic Schrödinger equation and why it is computationally intractable for systems beyond a few electrons; The Hohenberg-Kohn theorems as the theoretical foundation of DFT: ground-state energy as a functional of electron density; The Kohn-Sham equations: mapping the interacting many-electron problem onto a tractable non-interacting system with an exchange-correlation functional; Plane-wave basis sets and pseudopotentials as the computational representation enabling periodic DFT calculations; The self-consistent field (SCF) iteration as a fixed-point algorithm for solving the Kohn-Sham equations; Outputs of a DFT calculation: total energy, atomic forces, electronic band structure, and density of states as the primary computed quantities linked to observable material properties.

Unit 4

Microstructure Modeling and Continuum Methods: The microstructure as the mesoscale link between atomic structure and macroscopic performance: grain size, phase distribution, porosity, and their effect on mechanical and transport properties; Phase-field method as a continuum PDE framework for simulating microstructure evolution: order parameters, free energy functionals, and the Cahn-Hilliard and Allen-Cahn equations; Finite Difference and Finite Element discretization of the phase-field equations: stability, convergence, and the computational cost of 3D microstructure simulations; Voxel-based microstructure representation: 3D arrays as the

discrete data structure for storing phase, orientation, and composition fields; CALPHAD (CALculation of PHAse Diagrams) as a thermodynamic database approach to predicting equilibrium phase stability; Synthetic microstructure generation: random grain growth algorithms and their use as training data for downstream ML models.

Unit 5

High-Performance Computing for Materials Simulations: The computational cost landscape: DFT scales as $O(N^3)$, classical MD as $O(N \log N)$ with neighbor lists, and phase-field as $O(N_{\text{grid}})$; Parallelization strategies for materials codes: domain decomposition in MD (spatial decomposition of the simulation box), k-point parallelism in DFT, and OpenMP threading for shared-memory acceleration; GPU acceleration for MD: neighbor list construction and force evaluation as massively parallel operations; VASP, Quantum ESPRESSO, LAMMPS, and MOOSE as the canonical open-source codes for DFT, MD, and FEM respectively: their input file formats as domain-specific configuration languages; Workflow automation for high-throughput computation: FireWorks and AiiDA as job management frameworks for running thousands of DFT calculations programmatically; Data provenance and reproducibility: tracking simulation inputs, software versions, and pseudopotential choices as a scientific software engineering discipline.

Algorithmic Financial Engineering (MN-FIN)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Financial Markets as Computational Systems: Financial markets as information-processing machines: price as a signal aggregating distributed knowledge; Asset classes and their data representations: equities, bonds, derivatives, and currencies as structured records with tick-by-tick time-series; Market microstructure: the limit order book (LOB) as a priority-queue data structure; bid-ask spread, market depth, and order matching engines as real-time algorithmic systems; OHLCV (Open, High, Low, Close, Volume) as the canonical compressed representation of price history; Financial data formats and sources: FIX protocol for order messaging, Bloomberg and Quandl APIs, and the data engineering challenges of survivorship bias and point-in-time correctness; Return series, log-returns, and volatility as the derived statistical quantities on which all quantitative finance is built.

Unit 2
Stochastic Processes and the Mathematics of Price: Random walks and the Efficient Market Hypothesis as the null model of price dynamics; Brownian motion as the continuous-time limit of a random walk: the Wiener process and its properties; Geometric Brownian Motion (GBM) as the standard model of equity price evolution: $dS = \mu S\,dt + \sigma S\,dW_t$; Itô's Lemma as the chain rule of stochastic calculus: deriving the dynamics of functions of stochastic processes; The Black-Scholes-Merton (BSM) model: assumptions, the heat equation reduction, and the closed-form option pricing formula; The Greeks (Δ, Γ, Θ, $\mathcal{V}$, ρ) as partial derivatives of option price with respect to market variables: their role in hedging and risk management; Monte Carlo simulation of GBM paths as a numerical pricing engine for path-dependent exotic derivatives.

Unit 3
Algorithmic Trading Strategy Design: The algorithmic trading pipeline: data ingestion → signal generation → position sizing → order execution → risk management as a software architecture; Technical indicators as computable functions of price history: moving averages (SMA, EMA), RSI, MACD, and Bollinger Bands as feature engineering for price signals; Mean reversion strategies: the Ornstein-Uhlenbeck process as the stochastic model for cointegrated asset pairs; statistical arbitrage and pairs trading as a market-neutral strategy; Momentum strategies: cross-sectional and time-series momentum as systematic factor exposures; Backtesting as the empirical validation framework for trading strategies: walk-forward testing, transaction cost modeling, and the multiple comparisons problem (backtest overfitting); Performance metrics: Sharpe ratio, maximum drawdown, Calmar ratio, and the information ratio as the quantitative scorecard of a strategy.

Unit 4
Portfolio Optimization and Risk Modeling: Modern Portfolio Theory (MPT): the mean-variance optimization problem as a quadratic program over portfolio weights; The efficient frontier and the Capital Market Line; the Sharpe ratio as the objective of tangency portfolio construction; Covariance matrix estimation challenges: the curse of dimensionality, the Ledoit-Wolf shrinkage estimator, and factor models (CAPM, Fama-French) as structured covariance approximations; Risk factor decomposition: attributing portfolio risk to systematic factors vs. idiosyncratic residuals; Value at Risk (VaR) and Conditional Value at Risk (CVaR) as tail risk measures:

historical simulation, parametric, and Monte Carlo estimation methods; The Black-Litterman model: incorporating analyst views as a Bayesian update to the market equilibrium portfolio.

Unit 5

High-Frequency Trading and Market Systems Engineering: The latency arms race: co-location, FPGA-based order routing, and kernel bypass networking (DPDK, RDMA) as systems engineering for microsecond-level execution; Market making as an inventory management problem: the Avellaneda-Stoikov model for optimal bid-ask quote placement under adverse selection risk; Order execution algorithms: TWAP, VWAP, and Implementation Shortfall as strategies for minimizing market impact when liquidating large positions; Regulation and market stability: circuit breakers, kill switches, and the Flash Crash of 2010 as a case study in emergent systemic risk from algorithmic interaction; Alternative data in quantitative finance: satellite imagery, credit card transaction flows, and NLP-derived sentiment scores as non-traditional alpha signals; Regulatory technology (RegTech): transaction monitoring, trade surveillance, and anti-money laundering (AML) detection as classification problems on financial event streams.

Digital System Design (MN-SEM)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Boolean Algebra and the Logic of Computation: Digital abstraction: the reduction of continuous voltage levels to discrete binary values as the foundational engineering decision of all computing hardware; Boolean algebra as the mathematical system governing binary logic: axioms, theorems (De Morgan's, Shannon's expansion), and duality; Canonical forms: Sum of Products (SOP) and Product of Sums (POS) as normal forms for any Boolean function; Karnaugh maps as a graphical prime implicant grouping algorithm for two-, three-, four-, and five-variable minimization; Quine-McCluskey algorithm as the tabular, automatable generalization of K-map minimization; Hazards in combinational logic: static-1, static-0, and dynamic hazards as timing-dependent glitches and their elimination through consensus terms; Logic families and electrical characteristics: TTL vs. CMOS noise margins, fan-out, and propagation delay as the physical constraints on digital abstraction.

Unit 2
Combinational Building Blocks and Arithmetic Circuits: Standard combinational modules as reusable hardware abstractions: multiplexers, demultiplexers, encoders, decoders, and priority encoders; Binary arithmetic: unsigned and two's complement signed representation, addition, subtraction, and overflow detection; Ripple Carry Adder as the baseline implementation; Carry Lookahead Adder (CLA) as a parallel prefix computation that trades gate area for logarithmic depth; Booth's algorithm for signed multiplication as a recoding scheme reducing partial product count; Array multipliers and Wallace tree reduction as the structural parallelism underlying hardware multipliers; Comparators, barrel shifters, and ALU design as the composition of arithmetic and logic primitives into a general-purpose execution unit; Hazard-free multiplexer trees for implementing arbitrary functions as a universal combinational template.

Unit 3
Sequential Logic and Finite State Machines: The need for memory: feedback, bistability, and the SR latch as the minimal memory element; Flip-flop types: D, JK, T, and master-slave configurations; setup time, hold time, and clock-to-Q delay as the timing contracts of sequential elements; Registers and shift registers as parallel and serial data storage structures; Finite State Machines (FSMs) as the formal computational model of sequential digital systems: Moore vs. Mealy machines, state diagrams, and state tables; FSM synthesis: state encoding (binary, one-hot, Gray code) and its effect on area and timing; Synchronous design discipline: the single-clock domain assumption, metastability, and synchronizer circuits for crossing clock domain boundaries; Counters as specialized FSMs: ripple counters, synchronous binary counters, and Johnson counters.

Unit 4
Hardware Description Languages and FPGA Implementation: The abstraction gap: from schematic capture to Hardware Description Languages as the software engineering revolution of digital design; VHDL and Verilog/SystemVerilog as parallel, concurrent description languages: the fundamental difference between hardware concurrency and software sequential execution; Synthesizable vs. simulation-only constructs: the subset of HDL that maps to physical gates; RTL (Register Transfer Level) design style: always blocks, sensitivity lists, blocking vs. non-blocking

assignments as the source of the most common HDL bugs; FPGA architecture: configurable logic blocks (CLBs), look-up tables (LUTs), flip-flops, block RAMs, and DSP slices as the physical resources a synthesizer maps RTL onto; The synthesis-place-and-route-bitstream flow as the compilation pipeline from HDL source to configured FPGA hardware; Timing analysis: setup and hold slack, critical path identification, and timing constraints as the hardware analog of performance profiling.

Unit 5

RISC-V ISA and Processor Microarchitecture: Instruction Set Architecture as the hardware-software contract: CISC vs. RISC design philosophies and the motivations behind RISC-V as an open, modular, royalty-free ISA; RISC-V base integer ISA (RV32I/RV64I): register file, instruction encoding formats (R, I, S, B, U, J), and the minimal but complete instruction set; Single-cycle processor datapath: instruction fetch, decode, execute, memory access, and writeback as a pipeline of combinational logic and register state; Control unit design: generating datapath control signals from opcode and funct fields as a ROM-based or logic-based decoder; The performance equation: CPU time = Instruction Count × CPI × Clock Period and how microarchitectural choices trade these three factors; Introduction to pipelining: the five-stage RISC-V pipeline, structural hazards, data hazards (forwarding and stalling), and control hazards (branch prediction) as the fundamental challenges of instruction-level parallelism.

Fundamentals of GIS & Spatial Analysis (MN-GEO)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1

The Earth as a Database: Spatial Data Models and Coordinate Systems: Geographic Information Systems (GIS) as a computational framework for storing, querying, analyzing, and visualizing spatially referenced data; The Earth's shape problem: geoids, ellipsoids, and datums (WGS84, NAD83) as the reference surfaces underpinning all coordinate systems; Geographic coordinate systems (latitude, longitude) vs. projected coordinate systems (UTM, Web Mercator): the cartographic projection problem as a mathematical mapping from a curved surface to a flat plane with unavoidable distortion tradeoffs (area, shape, distance, direction); Vector data model: points, polylines, and polygons as geometric primitives with associated attribute tables — the shapefile and GeoJSON as standard serialization formats; Raster data model: georeferenced grids of cell values as the representation of continuous spatial phenomena (elevation, temperature, land cover); Spatial resolution, extent, and the modifiable areal unit problem (MAUP) as fundamental data quality and scale considerations.

Unit 2

Spatial Data Structures and Efficient Query: The spatial indexing problem: why standard B-tree indexes fail for two-dimensional range queries; R-trees as the canonical spatial index: minimum bounding rectangles, node splitting strategies, and the query algorithm for range and nearest-neighbor searches; Quadtrees as recursive spatial partitioning: region quadtrees for raster data and point quadtrees for vector data; Geohash and S2 cell hierarchies as space-filling curve encodings that map 2D spatial proximity to 1D string prefixes for use in standard key-value stores; Spatial joins: point-in-polygon, line intersection, and buffer overlap as the fundamental binary spatial predicates; Topological relationships formalized: the DE-9IM (Dimensionally Extended 9-Intersection Model) as the mathematical framework defining contains, intersects, touches, crosses, and disjoint; PostGIS as a spatial extension to PostgreSQL: geometry types, spatial indexes (GiST), and ST_ function families as a production spatial query engine.

Unit 3

Coordinate Transformations and Map Projections as Linear Algebra: Affine transformations: translation, rotation, scaling, and shearing as the 3×3 homogeneous matrix operations applied to georeferencing raster imagery; Georeferencing: establishing the correspondence between pixel coordinates and geographic coordinates using ground control points (GCPs) and least-squares polynomial fitting; Datum transformations: Helmert (7-parameter similarity) transformation for converting between geodetic reference frames; Raster resampling methods: nearest-neighbor, bilinear interpolation, and cubic convolution as tradeoffs between accuracy and computational cost when reprojecting grids; Vector reprojection: applying coordinate transformation functions point-by-point to convert geometries between coordinate reference systems; Error propagation in spatial data: positional accuracy, attribute accuracy, and lineage as the components of data quality metadata standards (ISO 19157).

Unit 4

Spatial Analysis: Proximity, Networks, and Surface Modeling: Buffer analysis, Voronoi diagrams, and Delaunay triangulation as the foundational proximity computation primitives; Network analysis on road graphs: shortest path (Dijkstra, A*), service area computation

(isochrones), and the traveling salesman problem as spatial optimization; Digital Elevation Models (DEMs): slope, aspect, hillshade, curvature, and viewshed analysis as raster-based terrain derivatives computed via finite-difference kernels; Interpolation of scattered point data to continuous surfaces: IDW (Inverse Distance Weighting), spline interpolation, and Kriging as a geostatistical interpolator that provides uncertainty estimates alongside predictions; Map algebra: local, focal, zonal, and global operations as a raster processing algebra analogous to array broadcasting; Spatial autocorrelation: Moran's I statistic as the formal test of whether nearby locations have more similar values than expected by chance, and its connection to the First Law of Geography.

Unit 5

Remote Sensing and Satellite Data Processing: The electromagnetic spectrum as the data source of satellite remote sensing: optical (multispectral, hyperspectral), SAR (synthetic aperture radar), and LiDAR as complementary sensing modalities; Radiometric calibration: converting raw digital numbers to at-sensor radiance and top-of-atmosphere reflectance as the preprocessing pipeline for optical imagery; Spectral indices as engineered features: NDVI (vegetation), NDWI (water), NDBI (built-up area) as normalized band ratios that isolate physical phenomena; Image classification: supervised (maximum likelihood, SVM) and unsupervised (k-means, ISODATA) approaches to assigning land cover labels to image pixels; Accuracy assessment: confusion matrix, overall accuracy, producer's and user's accuracy, and the Kappa coefficient as the validation framework for classified maps; Cloud computing for remote sensing: Google Earth Engine as a planetary-scale geospatial analysis platform where computation moves to the data rather than the data to the computation.

Urban Informatics & Smart Infrastructure (MN-CIV)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
The City as a Computational System: Urban informatics defined: the application of data science, sensing, and computation to understand and manage cities as complex adaptive systems; The smart city stack: physical infrastructure → sensor and actuator layer → communication networks → data platforms → analytics and applications as a layered architecture; Urban data taxonomy: administrative records (census, permits), transactional data (transit ticketing, utility billing), sensor streams (traffic loops, air quality monitors), and participatory data (citizen reports, social media) as heterogeneous sources with different latency and quality profiles; City-scale data platforms: urban operating system (Urban OS) architectures and open data portals as the infrastructure for data governance and third-party application development; The digital divide and data equity: uneven sensor deployment, demographic underrepresentation in datasets, and the risk of algorithmic systems reinforcing spatial inequality; Key urban indicators: population density, land use mix, walkability index, and heat island intensity as computed metrics derived from raw urban data.

Unit 2
Urban Sensing Networks and Data Acquisition: Fixed sensing infrastructure: inductive loop detectors, pneumatic tube counters, CCTV-based vehicle detection, and environmental sensor networks as the permanent data collection layer of a city; Mobile sensing: probe vehicles (taxis, buses with GPS), crowdsourced pothole detection via accelerometers, and participatory sensing apps as opportunistic data collection paradigms; Communication infrastructure for smart cities: LoRaWAN for low-power wide-area sensor networks, NB-IoT for deep-indoor metering, 5G for latency-sensitive applications (V2X, remote surgery), and DSRC/C-V2X for vehicle-to-infrastructure communication; Smart metering for utilities: AMI (Advanced Metering Infrastructure) for electricity, water, and gas; interval data as a high-frequency time series enabling disaggregation of end-use consumption; Data fusion from heterogeneous sources: temporal alignment, spatial co-registration, and handling missing data in multi-sensor urban streams; Edge-cloud continuum: processing latency-sensitive urban data locally (traffic signal control) vs. aggregating historical data centrally (urban planning analytics).

Unit 3
Urban Mobility and Transportation Systems: Transportation networks as weighted directed graphs: nodes as intersections, edges as road segments with travel time, capacity, and incident attributes; Traffic flow theory: the fundamental diagram relating flow, density, and speed; the LWR (Lighthill-Whitham-Richards) model as a PDE description of traffic dynamics; Signal control algorithms: fixed-time plans, actuated control, and the Webster formula for optimal cycle length as the classical control-theoretic approach to intersection management; Public transit networks: GTFS (General Transit Feed Specification) as the standard data schema for routes, stops, trips, and schedules; transit assignment and the stochastic user equilibrium problem; Shared mobility systems: bike-share and ride-hailing as queuing systems; station-based rebalancing as an optimization problem over demand prediction; Multimodal journey planning as a time-expanded graph shortest-path problem: integrating walking, transit, cycling, and ride-hailing into a unified routing engine.

Unit 4

Urban Infrastructure Management and Resilience: Urban infrastructure systems as interdependent networks: power grids, water distribution, wastewater, telecommunications, and transportation as coupled graphs where failure in one propagates to others; Structural health monitoring (SHM): embedding vibration, strain, and tilt sensors in bridges and buildings; anomaly detection in sensor streams as the early-warning system for infrastructure degradation; Water distribution network modeling: pipe networks as hydraulic graphs governed by conservation of mass and energy; leak detection and pipe burst localization as inverse problems on sensor data; Smart grid fundamentals: the bidirectional power flow problem introduced by distributed renewable generation; demand response as a real-time optimization of load shedding under supply constraints; Resilience metrics for urban infrastructure: redundancy, robustness, rapidity, and resourcefulness as the 4R framework; cascading failure simulation using percolation theory on interdependent networks; Urban flood modeling: drainage network capacity, impervious surface coverage, and DEM-based inundation simulation as a coupled hydraulic-GIS computational problem.

Unit 5

Urban Governance, Open Data, and Civic Technology: Open government data as a public good: data licensing (CC-BY, ODbL), machine-readable formats (CSV, GeoJSON, SPARQL endpoints), and the five-star open data maturity model; Urban dashboards and decision support systems: real-time KPI visualization for city operators as a data pipeline from sensor ingestion to front-end rendering; Participatory platforms and e-governance: 311 service request systems, participatory budgeting platforms, and fix-my-street applications as citizen-government data exchange interfaces; Algorithmic accountability in smart city systems: auditing predictive policing models, automated permit systems, and dynamic pricing algorithms for fairness and transparency; Smart city standards and frameworks: ISO 37120 (city indicators), ITU-T Y.4000 (IoT for smart cities), and the ISO/IEC 30145 smart city ICT reference framework as the interoperability layer enabling vendor-neutral city platforms; Privacy by design in urban sensing: anonymization of mobility traces, federated analytics for utility data, and the legal landscape of public surveillance under GDPR.

Fundamentals of Automotive Embedded Systems (MN-EVM)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1

The Vehicle as a Networked Embedded System: Modern vehicles as distributed embedded computers: the evolution from mechanical linkages to software-defined functions and the role of Electronic Control Units (ECUs) as the computational nodes of the vehicle; ECU architecture: microcontroller selection criteria (real-time capability, flash/RAM, peripheral set), the hardware abstraction layer (HAL), and the separation of application software from hardware drivers; The automotive software stack: AUTOSAR (AUTomotive Open System ARchitecture) Classic and Adaptive as the standardized layered middleware enabling ECU software portability across hardware platforms; In-vehicle network topology evolution: point-to-point wiring → shared bus (CAN) → domain architecture → zonal Ethernet architecture as the progression driven by bandwidth and latency demands; Key automotive data formats: DBC files as the schema defining CAN message IDs, signal names, bit positions, scaling factors, and units; UDS (Unified Diagnostic Services, ISO 14229) as the diagnostic communication protocol for ECU flashing and fault code reading.

Unit 2

Automotive Communication Protocols: CAN (Controller Area Network) as the foundational automotive bus: frame structure (arbitration ID, DLC, data, CRC, ACK), non-destructive bitwise arbitration via CSMA/CR, error detection mechanisms, and the bus-off recovery state machine; CAN FD (Flexible Data Rate) as the evolution addressing bandwidth limitations: variable data phase bit rate up to 8 Mbps and 64-byte payload; LIN (Local Interconnect Network) as a low-cost single-wire sub-bus for body electronics: master-slave scheduling, break field synchronization, and checksum variants; FlexRay as the deterministic, fault-tolerant protocol for safety-critical chassis applications: static and dynamic segments, cycle structure, and the clock synchronization algorithm; Automotive Ethernet (100BASE-T1, 1000BASE-T1) and the SOME/IP service-oriented middleware as the backbone of high-bandwidth ADAS sensor data transport; CAN matrix analysis: decoding raw hex CAN logs using DBC definitions as a data engineering task fundamental to vehicle software development and testing.

Unit 3

Electric Powertrain: Computation and Control: Electric vehicle powertrain architecture: battery pack, power electronics (inverter, DC-DC converter, OBC), electric motor, and the thermal management system as the four subsystems governed by embedded software; Battery electrochemistry from a systems perspective: cell voltage characteristics, State of Charge (SoC) as a hidden state variable, and State of Health (SoH) as a degradation metric; Battery Management System (BMS) algorithms: coulomb counting and open-circuit voltage lookup as SoC estimation baselines; cell balancing topologies (passive vs. active) and their firmware implementations; Field-Oriented Control (FOC) of permanent magnet synchronous motors (PMSM): the Clarke and Park transforms as coordinate system rotations that decouple torque and flux control into two independent DC current controllers; Space Vector PWM (SVPWM) as the modulation strategy converting voltage references to inverter switch timing; Regenerative braking: the torque blending algorithm coordinating hydraulic and electric braking to maximize

energy recovery within ABS and stability control constraints.

Unit 4

Real-Time Operating Systems and Functional Safety: Hard real-time constraints in automotive embedded systems: the consequence of missed deadlines in brake-by-wire, steer-by-wire, and motor control as safety-critical failures; RTOS concepts: task scheduling (rate-monotonic, earliest-deadline-first), priority inversion, the priority inheritance protocol, and inter-task communication via queues and semaphores; AUTOSAR OS as the automotive RTOS standard: tasks, alarms, schedule tables, and the distinction between basic and extended tasks; ISO 26262 (Road Vehicles — Functional Safety) as the automotive safety standard: the safety lifecycle, Automotive Safety Integrity Levels (ASIL A–D) as risk classification, and the hardware and software requirements imposed at each level; Diagnostic Trouble Codes (DTCs) and fault management: the permanent fault, pending fault, and confirmed fault state machine as the embedded software pattern for robust fault detection and reporting; Watchdog timers, redundant computation, and voting logic as the hardware mechanisms for achieving ASIL-D fault tolerance in safety-critical ECUs.

Unit 5

ADAS Sensing Fundamentals and the Software-Defined Vehicle: Advanced Driver Assistance Systems (ADAS) as the sensing and actuation stack beneath full autonomy: ACC, AEB, LKA, BSD, and surround-view as the Level 1–2 automation building blocks; Sensor modalities for ADAS: cameras (monocular, stereo, fisheye), radar (77 GHz FMCW), ultrasonic, and LiDAR — their operating principles, range-resolution-angular resolution tradeoffs, and failure modes in adverse weather; Sensor time synchronization: hardware timestamps, PPS (Pulse Per Second) signals, and PTP (IEEE 1588 Precision Time Protocol) as the mechanisms ensuring multi-sensor data fusion is temporally coherent; Over-the-Air (OTA) software updates: the A/B partition scheme, rollback safety mechanisms, and delta patching as the software deployment infrastructure of the software-defined vehicle; Cybersecurity in automotive systems: the TARA (Threat Analysis and Risk Assessment) methodology under ISO/SAE 21434, attack surfaces (OBD port, telematics unit, V2X radio), and the SecOC (Secure Onboard Communication) protocol for authenticating CAN messages.

Advanced Minor (Level 2)

Genomic Data Science (MN-BIO-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
High-Throughput Sequencing and the Genomics Data Pipeline: Next-Generation Sequencing (NGS) technologies and the scale of modern genomic data: millions of short reads, terabytes per experiment; The NGS analysis pipeline as a software engineering problem: quality control (FastQC), trimming, alignment (BWA, STAR), and variant calling (GATK); SAM/BAM file formats as compressed, indexed data structures for aligned reads; The VCF (Variant Call Format) as a standardized schema for genomic variants; Scalability challenges: processing whole-genome sequencing cohorts with thousands of samples; Introduction to workflow management systems (Snakemake, Nextflow) for reproducible pipelines.

Unit 2
Statistical Genomics and Variant Analysis: Single Nucleotide Polymorphisms (SNPs) and structural variants as the raw signal of genomic studies; Genome-Wide Association Studies (GWAS): the linear regression model linking genotype to phenotype at millions of loci simultaneously; The multiple testing problem and Bonferroni correction in the genomic context; Linkage disequilibrium and haplotype blocks as a dimensionality reduction phenomenon; Population stratification as a confounding factor and Principal Component Analysis (PCA) as its computational solution; Polygenic risk scores as a predictive modeling application.

Unit 3
Transcriptomics and Differential Expression Analysis: RNA-Seq as a quantitative measurement of gene activity: reads as a proxy for transcript abundance; The count matrix as the central data object: genes $\times$ samples; Normalization strategies: RPKM, FPKM, TPM, and DESeq2's size-factor normalization to remove technical bias; Differential expression analysis as a statistical hypothesis testing problem (negative binomial model); Multiple testing correction (Benjamini-Hochberg FDR); Downstream interpretation: Gene Ontology (GO) enrichment and pathway analysis (KEGG) as knowledge-graph queries on expression results.

Unit 4
Machine Learning on Genomic Data: Feature engineering for genomic data: encoding sequences as one-hot vectors, k-mer frequency spectra, and embeddings; Supervised learning for genomic tasks: splice site prediction, promoter classification, and variant effect prediction; Deep learning architectures tailored to sequences: 1D Convolutional Neural Networks for motif detection, recurrent models for long-range dependencies; The challenge of interpretability: attention mechanisms and saliency maps for discovering learned biological motifs; Transfer learning with pre-trained genomic language models (DNABERT, Nucleotide Transformer); Class imbalance and data leakage as critical pitfalls specific to genomic classification.

Unit 5
Single-Cell Genomics and Emerging Frontiers: Single-cell RNA sequencing (scRNA-seq): motivation, library preparation, and the cell $\times$ gene count matrix as a sparse, high-dimensional data object; Dimensionality reduction for single-cell data: PCA, t-SNE, and UMAP as visualization and clustering tools; Cell type annotation as an unsupervised clustering problem (Louvain, Leiden algorithms); Trajectory inference: modeling differentiation paths as a graph problem on

the cell manifold; Introduction to multi-omics data integration: combining genomics, transcriptomics, and epigenomics as a multi-view learning problem; Ethical and privacy considerations in genomic data sharing: re-identification risks and federated learning as a privacy-preserving solution.

Medical Image Analysis (MN-MED-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Medical Imaging Modalities and Data Representation: Physics and clinical purpose of major imaging modalities: X-ray (attenuation contrast), CT (Hounsfield units and volumetric reconstruction), MRI (spin relaxation and tissue contrast), Ultrasound (acoustic impedance), and PET (metabolic activity via radiotracer decay); DICOM as the universal standard: file structure, metadata tags, pixel data encoding, and series/study/patient hierarchy as a relational schema; 2D images, 3D volumes, and 4D time series as tensors: voxel spacing, orientation matrices, and coordinate systems (patient, scanner, world); Intensity histograms, windowing, and level adjustment as the first-order data exploration tools; Challenges specific to medical image data: class imbalance (rare pathologies), annotation scarcity, and inter-rater variability as dataset construction problems.

Unit 2
Classical Image Processing for Medical Contexts: Image enhancement: histogram equalization, CLAHE for local contrast improvement in low-dose CT and mammography; Edge detection: Sobel, Canny, and Laplacian of Gaussian (LoG) operators applied to boundary delineation; Mathematical morphology: erosion, dilation, opening, and closing for noise removal and shape refinement in binary masks; Region growing and watershed segmentation as classical unsupervised partitioning methods; Geometric transformations: rigid (translation, rotation) and deformable registration as the problem of aligning two images of the same patient across time or modality; Mutual information as a modality-independent similarity metric for multi-modal registration (CT-MRI fusion).

Unit 3
Deep Learning Architectures for Medical Image Segmentation: The segmentation task formalized: pixel-wise (2D) or voxel-wise (3D) classification as a dense prediction problem; U-Net architecture: encoder-decoder structure with skip connections as the dominant paradigm for biomedical segmentation; Loss functions for imbalanced segmentation: Dice loss, focal loss, and their combination; 3D U-Net and V-Net for volumetric CT and MRI segmentation; Patch-based training strategies to handle GPU memory constraints with large 3D volumes; nnU-Net as an automated self-configuring segmentation framework: dataset fingerprinting and hyperparameter selection without manual tuning; Evaluation metrics: Dice Similarity Coefficient (DSC), Hausdorff Distance, and volumetric overlap.

Unit 4
Computer-Aided Detection and Diagnostic Classification: CAD systems as a second-reader paradigm: screening mammography, lung nodule detection (LUNA16), and diabetic retinopathy grading; Transfer learning from ImageNet to medical imaging: domain shift challenges and fine-tuning strategies; Vision Transformers (ViT) and hybrid CNN-Transformer architectures for global context modeling in histopathology; Multiple Instance Learning (MIL) for weakly supervised classification from slide-level labels in whole-slide imaging (WSI); Uncertainty quantification: Monte Carlo Dropout and deep ensembles for communicating prediction confidence in clinical settings; Grad-CAM and attention rollout for explainability: generating heatmaps that localize the image regions driving a diagnosis.

Unit 5

Clinical Deployment, Federated Learning, and Emerging Directions: The gap between research accuracy and clinical deployment: distribution shift, site-specific variation, and continuous monitoring post-deployment; Federated learning as the canonical solution to multi-site medical AI: the FedAvg algorithm, communication efficiency, and the non-IID data problem across hospital cohorts; Differential privacy in federated medical imaging: noise mechanisms and the privacy-utility tradeoff; Synthetic data generation with Diffusion Models and GANs for data augmentation and rare disease simulation; Foundation models in medical imaging: SAM (Segment Anything Model) adaptation, MedSAM, and universal segmentation; Regulatory pathway for AI-based medical devices: FDA 510(k) clearance, algorithm change protocols, and post-market surveillance requirements.

Spatial Computing & AR/VR (MN-DES-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
The Geometry of Virtual Worlds: Spatial computing defined: the shift from 2D screen-based interaction to 3D world-anchored computation; Coordinate systems and transformations: local, world, and camera spaces; the model-view-projection (MVP) matrix pipeline as the mathematical engine of all 3D rendering; Homogeneous coordinates and the role of the 4×4 transformation matrix for unified translation, rotation, and scaling; Quaternions as a rotation representation: advantages over Euler angles (gimbal lock avoidance) and interpolation via SLERP; Scene graphs as hierarchical tree data structures for managing spatial relationships between objects; Frustum culling and spatial partitioning (octrees, BVH) as algorithmic optimizations for real-time rendering performance.

Unit 2
Rendering Pipelines and Real-Time Graphics: The GPU rendering pipeline: vertex shading, rasterization, fragment shading, and output merging as a massively parallel dataflow architecture; Physically Based Rendering (PBR): the Cook-Torrance BRDF model, metallic-roughness workflow, and image-based lighting (IBL) as the standard for photorealistic real-time materials; Shadow mapping and screen-space ambient occlusion (SSAO) as depth-buffer-based approximation techniques; Level of Detail (LoD) systems and mesh simplification as adaptive quality-performance tradeoff mechanisms; Foveated rendering: exploiting the human eye's non-uniform acuity to concentrate GPU budget at the gaze center, critical for VR headset performance; The VR rendering constraints: >90 Hz refresh rate, sub-20ms motion-to-photon latency, and stereoscopic rendering as hard real-time requirements.

Unit 3
Tracking, Sensing, and World Understanding: Inside-out tracking: Simultaneous Localization and Mapping (SLAM) as the core algorithm enabling headsets to understand their position without external infrastructure; Visual-Inertial Odometry (VIO): fusing camera frames with IMU data via an Extended Kalman Filter for robust 6-DoF pose estimation; Plane detection and scene reconstruction: fitting geometric primitives to depth sensor point clouds; Hand and body tracking: MediaPipe and skeletal model fitting as a graph-based pose estimation problem; Eye tracking: Purkinje image-based gaze estimation and its dual role in interaction (gaze-as-input) and rendering optimization (foveated rendering); Spatial anchors as persistent coordinate system attachments: storing and retrieving AR content relative to physical landmarks across sessions.

Unit 4
Spatial Interaction Design and User Experience in 3D: The interaction design vocabulary of spatial computing: gaze, pinch, voice, ray casting, and direct hand manipulation as input modalities; Degrees of freedom (DoF): 3-DoF (orientation only) vs. 6-DoF (full position and orientation) as a fundamental UX constraint; Comfort and presence: vergence-accommodation conflict as the optical basis of VR-induced discomfort; Locomotion paradigms: teleportation, arm-swinging, and continuous movement as tradeoffs between immersion and simulator sickness; Spatial audio: HRTF (Head-Related Transfer Function) convolution as the signal processing technique for 3D positional sound; Designing for shared AR spaces: multi-user session syn-

chronization, conflict resolution for overlapping virtual objects, and social proxemics in mixed reality.

Unit 5

Computer Vision and AI for Spatial Intelligence: Object detection and instance segmentation in AR: YOLO and Mask R-CNN as real-time scene understanding backends; Neural Radiance Fields (NeRF): volumetric scene representation learned from posed images as a photorealistic 3D reconstruction technique; 3D Gaussian Splatting as a faster, rasterizable alternative to NeRF for real-time novel view synthesis; Semantic scene understanding: assigning object-class labels to point cloud regions as a 3D classification problem; Foundation models for spatial AI: SAM 2 for video object segmentation and its integration into AR object persistence; Digital twins in spatial computing: synchronizing a real-time physical environment with its virtual counterpart using sensor fusion, SLAM, and simulation for industrial and architectural applications.

Digital Twins & Cyber-Physical Systems (MN-MFG-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Cyber-Physical Systems: Formal Foundations: Cyber-Physical Systems (CPS) defined: the tight coupling of computation, communication, and physical dynamics where software correctness has physical consequences; Hybrid systems as the mathematical model of CPS: continuous ODEs governing physical evolution interrupted by discrete computational events; Hybrid automata: states, invariants, guards, and reset maps as the formal specification language for CPS behavior; Reachability analysis: computing the set of states a hybrid system can reach as a safety verification problem; Timed automata and real-time constraints: modeling and verifying timing guarantees in control loops; Introduction to model-based design: using Simulink/Modelica as executable specification environments where the physical plant and the controller co-evolve before any hardware is built.

Unit 2
Digital Twin Architecture and Synchronization: Digital twin taxonomy: descriptive (as-built model), predictive (simulation-driven forecast), and prescriptive (optimization-driven recommendation) twins; The synchronization problem: state estimation from noisy, incomplete sensor streams using the Kalman Filter and its nonlinear extensions (EKF, UKF) to keep the virtual model consistent with physical reality; Asset Administration Shell (AAS) as the standardized data model for digital twins in Industry 4.0: submodels, properties, and operations; Ontologies for manufacturing: describing assets, processes, and relationships in a machine-readable knowledge graph (OWL, RDF); Twin composition: federating multiple asset-level twins into a system-level simulation; Versioning and configuration management for twin models: treating simulation code and physical parameters as software artifacts under version control.

Unit 3
Physics-Based Simulation and Surrogate Modeling: Finite Element Analysis (FEA) as a discretization of continuous physical domains: mesh generation, boundary conditions, and the stiffness matrix as a sparse linear algebra problem; Computational Fluid Dynamics (CFD) for thermal and flow simulation in manufacturing processes: Reynolds-Averaged Navier-Stokes (RANS) equations and their numerical solution; Multibody dynamics simulation for robotic and mechanical systems: rigid body kinematics, contact mechanics, and constraint solving; The surrogate modeling problem: replacing expensive physics simulations with fast data-driven approximations; Gaussian Process Regression as a probabilistic surrogate: posterior mean as prediction and posterior variance as uncertainty quantification; Physics-Informed Neural Networks (PINNs): embedding governing PDEs as soft constraints in the loss function to produce physically consistent surrogates with sparse data.

Unit 4
Closed-Loop Control, Optimization, and Autonomous CPS: Classical control recap in the CPS context: PID controllers, stability (Lyapunov methods), and the cost of instability in physical systems; Model Predictive Control (MPC) as the canonical optimization-based controller: receding horizon, constraint handling, and the role of the digital twin as the internal prediction

model; Reinforcement Learning for CPS control: Sim-to-Real transfer using the digital twin as a safe training environment before deployment on physical hardware; Multi-objective optimization of manufacturing processes: Pareto frontiers over competing objectives (throughput, energy consumption, defect rate) using evolutionary algorithms; Collaborative robotics (cobots): task and motion planning (TAMP) as a combined discrete-continuous search problem; Digital twin-driven process optimization: closed-loop parameter tuning where the twin evaluates candidate settings before they are applied to the physical line.

Unit 5

Industrial AI, Scalability, and the Future of Manufacturing: Federated learning for manufacturing: training anomaly detection models across multiple factory sites without centralizing proprietary process data; Large-scale digital twin platforms: AWS IoT TwinMaker, Azure Digital Twins, and NVIDIA Omniverse as cloud-native twin infrastructures and their architectural patterns; Graph Neural Networks for modeling complex manufacturing systems: representing machines, materials, and processes as nodes and their dependencies as edges for holistic system-level inference; Generative design: using diffusion models and topology optimization to synthesize manufacturable part geometries that satisfy structural constraints; Standards and interoperability: RAMI 4.0 as the reference architecture model for Industry 4.0, and its relationship to IEC 61512 (ISA-88) and ISO 23247 (Digital Twin for Manufacturing); Sustainability and energy-aware manufacturing: digital twins as instruments for carbon footprint modeling, energy flow optimization, and lifecycle assessment.

Material Informatics (MN-MAT-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1

Featurization: Turning Atoms into Numbers: The central challenge of materials ML: representing a crystal structure or molecule as a fixed-length numerical vector without losing physical symmetries; Symmetry constraints as hard requirements on any valid materials representation: invariance to translation, rotation, reflection, and permutation of identical atoms; Compositional descriptors: Magpie and Matminer feature sets encoding elemental statistics (electronegativity, atomic radius, valence) as tabular features; Structural descriptors: radial distribution functions, Smooth Overlap of Atomic Positions (SOAP), and Many-Body Tensor Representation (MBTR) as rotationally invariant fingerprints of local atomic environments; Graph representation of crystals: atoms as nodes, bonds as edges, and edge attributes encoding interatomic distances and angles; The atomistic simulation database as training data: parsing CIF files and VASP output with Pymatgen and ASE as the data engineering layer.

Unit 2

Machine Learning Interatomic Potentials: The core motivation: replacing $O(N^3)$ DFT force evaluations with a fast ML surrogate that maintains quantum accuracy at classical MD cost; Behler-Parrinello Neural Network Potentials (NNPs): decomposing total energy into atomic energy contributions, symmetry function inputs, and feedforward network architecture; Gaussian Approximation Potentials (GAP): Gaussian process regression on SOAP descriptors as a probabilistic force field with uncertainty quantification; Message Passing Neural Networks for potentials: the NequIP and MACE architectures using equivariant features that respect $SO(3)$ symmetry exactly; Active learning for potential training: uncertainty-driven selection of new DFT reference calculations to iteratively expand the training set toward completeness; Validation protocols: force, energy, and stress MAE on held-out test sets; phonon dispersion and elastic constants as physics-based sanity checks beyond statistical metrics.

Unit 3

Property Prediction and High-Throughput Screening: Supervised learning for materials property prediction: formation energy, bandgap, bulk modulus, and ionic conductivity as regression targets from the Materials Project database; Crystal Graph Convolutional Neural Networks (CGCNN) and MatErials Graph Network (MEGNet) as the canonical graph neural network architectures for crystal property prediction; Transfer learning in materials: pre-training on large DFT datasets and fine-tuning on expensive experimental targets with limited labels; Uncertainty quantification for property prediction: conformal prediction and deep ensembles as methods for producing calibrated prediction intervals; The high-throughput screening funnel: using cheap descriptors to filter a large chemical space before applying expensive models, analogous to a multi-stage database query; Pareto-optimal materials discovery: navigating conflicting property objectives (high conductivity, low cost, chemical stability) as a multi-objective optimization problem.

Unit 4

Generative Models for Materials Design: Inverse materials design as a generative problem: given a target property, synthesize a valid crystal structure that achieves it; Variational Autoencoders (VAEs) for materials: encoding crystal structures into a continuous latent space and

decoding latent vectors into new candidate structures; Diffusion models for crystal structure generation: CDVAE and DiffCSP as denoising diffusion frameworks operating on fractional atomic coordinates and lattice parameters; Validity constraints in generative materials design: charge neutrality, electronegativity balance, and coordination number rules as hard physical filters on generated candidates; Reinforcement learning for materials optimization: formulating the atom substitution and structural relaxation sequence as a Markov Decision Process; Synthesizability prediction as a binary classification bottleneck: distinguishing computationally stable from experimentally accessible materials.

Unit 5

Foundation Models, Experimental Integration, and the Autonomous Lab: Materials foundation models: GNoME (Google DeepMind), CHGNet, and MACE-MP-0 as universal interatomic potentials trained on tens of millions of DFT calculations; The autonomous materials discovery loop: computational prediction → robotic synthesis → automated characterization → ML model update as a closed-loop active learning system; Self-driving laboratories: integrating liquid handling robots, automated XRD and spectroscopy, and Bayesian optimization into a fully autonomous experimental pipeline; Multi-fidelity learning: combining cheap low-fidelity data (classical MD, GGA-DFT) with expensive high-fidelity data (hybrid DFT, experiment) in a hierarchical model to maximize information per compute dollar; Natural language processing for materials science: extracting structured property data from the unstructured literature using named entity recognition and relation extraction; FAIR data principles (Findable, Accessible, Interoperable, Reusable) applied to materials databases: metadata standards, persistent identifiers, and open APIs as the infrastructure of collaborative materials informatics.

Blockchain & DeFi Architectures (MN-FIN-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1

Cryptographic Foundations of Distributed Trust: The trust problem in financial systems: why traditional finance requires central intermediaries and the cryptographic primitives that eliminate this requirement; Cryptographic hash functions: SHA-256 and Keccak-256 as collision-resistant, one-way compression functions; their role in data integrity, Merkle trees, and proof-of-work; Digital signatures: ECDSA over the secp256k1 curve as the authentication mechanism for blockchain transactions; public-private key pairs as identity in a trustless system; Merkle trees and Merkle Patricia Tries as the authenticated data structures underpinning Bitcoin's UTXO set and Ethereum's world state; Commitment schemes and zero-knowledge proofs: the intuition behind proving knowledge of a secret without revealing it; The Byzantine Generals Problem as the formal statement of the distributed consensus challenge that blockchain solves.

Unit 2

Consensus Mechanisms and Blockchain Architecture: The blockchain data structure: linked hash-chained blocks as a tamper-evident append-only log; Nakamoto consensus and Proof-of-Work: the longest-chain rule, the 51% attack threshold, and the energy cost as an intentional computational barrier; Proof-of-Stake: validator selection by staked collateral, slashing conditions, and the nothing-at-stake problem; Ethereum's Gasper consensus: combining LMD-GHOST fork choice with Casper-FFG finality as a hybrid BFT-Nakamoto protocol; Byzantine Fault Tolerant (BFT) consensus for permissioned chains: PBFT, Tendermint, and HotStuff as the family of protocols tolerating up to $\lfloor (n-1)/3 \rfloor$ faulty nodes; Blockchain trilemma: the fundamental tension between decentralization, security, and scalability as the architectural constraint shaping all design decisions.

Unit 3

Smart Contracts and the Programmable Settlement Layer: The Ethereum Virtual Machine (EVM) as a quasi-Turing-complete, deterministic, sandboxed state machine replicated across all nodes; Gas as the metering mechanism: opcode pricing, the gas limit as a DoS prevention tool, and the EIP-1559 fee market as a dynamic base fee algorithm; Solidity contract architecture: state variables as persistent storage slots, functions as state transition triggers, events as an indexed log for off-chain consumption; Common smart contract patterns: Ownable, Pausable, Upgradeable Proxy (EIP-1967), and the Factory pattern as reusable building blocks; Smart contract security vulnerabilities: reentrancy (the DAO hack), integer overflow, front-running (MEV), and oracle manipulation as a taxonomy of attack vectors with their mitigation patterns; Formal verification of smart contracts: using model checkers (Certora Prover, Echidna) to prove invariant preservation as a correctness guarantee.

Unit 4

Decentralized Finance Protocols: DeFi as the re-implementation of financial primitives (exchange, lending, derivatives) as permissionless smart contract systems; Automated Market Makers (AMMs): Uniswap's constant product formula $x \cdot y = k$ as a bonding curve that replaces the order book; concentrated liquidity (Uniswap v3) as a capital efficiency optimization; Decentralized lending protocols: Aave and Compound as over-collateralized lending pools with algorithmic interest rate models (utilization-rate curves); Liquidation mechanisms as on-

chain margin calls: health factor computation and the liquidation bonus as an incentive for third-party liquidators; Decentralized stablecoins: MakerDAO's DAI as a CDP (Collateralized Debt Position) system; algorithmic stablecoins and the Terra/UST collapse as a case study in reflexive death spiral dynamics; Yield aggregators and composability: DeFi protocols as Lego blocks composed via standard interfaces (ERC-20, ERC-4626) to construct complex financial strategies.

Unit 5

Scalability, Interoperability, and the Evolving Architecture: The scalability bottleneck: Ethereum mainnet throughput (∼15 TPS) vs. Visa (∼24,000 TPS) and the architectural solutions to close this gap; Optimistic rollups (Arbitrum, Optimism): executing transactions off-chain, posting compressed calldata on-chain, and the fraud proof challenge window as a security mechanism; ZK-rollups (zkSync, StarkNet): validity proofs via SNARKs and STARKs as cryptographic certificates of correct off-chain execution that require no challenge period; Data availability: EIP-4844 (proto-danksharding) and blob-carrying transactions as the L1 reform enabling cheap rollup data posting; Cross-chain bridges: lock-and-mint and liquidity pool architectures, and bridge hacks as the dominant DeFi attack vector; Maximal Extractable Value (MEV): block proposer reordering, sandwich attacks, and PBS (Proposer-Builder Separation) as the political economy of blockchain transaction ordering.

SoC Design & Verification (MN-SEM-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Advanced Processor Microarchitecture: Superscalar execution: multiple issue slots, instruction dispatch, and the out-of-order execution engine as a hardware scheduler; Tomasulo's algorithm: reservation stations, the common data bus, and register renaming as the solution to WAR and WAW hazards that limit in-order pipelines; Reorder Buffer (ROB) and precise exceptions: maintaining the illusion of sequential execution in a speculatively executing processor; Branch prediction: bimodal, two-level adaptive (gshare), and TAGE predictors as increasingly sophisticated pattern matchers for control flow; Memory hierarchy design: cache associativity, replacement policies (LRU, PLRU), write policies (write-back, write-through), and the inclusion property in multi-level caches; Hardware prefetching: stride prefetchers and stream buffers as latency-hiding mechanisms; NUMA architectures and cache coherence protocols: MSI, MESI, and MOESI state machines as the distributed consensus protocol for shared memory correctness.

Unit 2
System-on-Chip Architecture and On-Chip Interconnects: SoC as a heterogeneous integration of processor cores, accelerators, memory controllers, and peripheral IP blocks on a single die; IP-based design methodology: hard IP (fixed layout), soft IP (synthesizable RTL), and firm IP as the three tiers of reusable hardware components; On-chip bus protocols: AMBA AXI4 as the dominant high-performance interconnect standard; AXI channels (AW, W, B, AR, R), handshaking, and outstanding transaction support; AXI4-Lite for low-bandwidth register-mapped peripherals and AXI4-Stream for unidirectional data flow; Network-on-Chip (NoC) architectures: mesh, torus, and fat-tree topologies; wormhole routing, virtual channels, and deadlock avoidance as the packet-switched alternative to shared buses at scale; Memory-mapped I/O and the address map as the software interface to hardware: base address registers, BAR allocation, and device tree descriptions.

Unit 3
Hardware Accelerator Design and RISC-V Extensions: The case for hardware acceleration: Amdahl's Law and the end of Dennard scaling driving the shift from general-purpose cores to domain-specific architectures; Systolic arrays as the canonical accelerator architecture for matrix multiplication: data flow patterns, PE design, and the connection to TPU and neural network inference engines; RISC-V custom instruction extensions: the X (non-standard extension) space, adding application-specific instructions to the base ISA without breaking compatibility; Tightly-coupled accelerators vs. memory-mapped coprocessors: latency, bandwidth, and programming model tradeoffs; Chisel as a hardware construction language embedded in Scala: generators, parameterization, and the Rocket Chip SoC generator as the RISC-V reference implementation; High-Level Synthesis (HLS): compiling C/C++ to RTL with Vitis HLS and understanding the scheduling, binding, and allocation steps that bridge software and hardware.

Unit 4
Functional Verification and Formal Methods: The verification gap: why simulation alone cannot guarantee correctness and why verification consumes over 70% of modern chip design effort; SystemVerilog as the verification language: interfaces, clocking blocks, program blocks, and the layered testbench architecture; Universal Verification Methodology (UVM): the component

hierarchy (sequencer, driver, monitor, scoreboard, agent), factory pattern for component overriding, and the transaction-level modeling (TLM) abstraction; Constrained-random stimulus generation: SystemVerilog constraints as a declarative specification of the legal input space solved by an SMT-based constraint solver; Functional coverage: covergroups, coverpoints, and cross-coverage as a metric for measuring verification completeness; Formal property verification: writing SVA (SystemVerilog Assertions) properties and using model checkers (JasperGold, SymbiYosys) to prove or disprove them exhaustively over all reachable states.

Unit 5

Physical Design, Semiconductor Manufacturing, and the Tapeout Flow: The RTL-to-GDSII flow as the complete compilation pipeline from hardware description to mask data: synthesis → floorplanning → placement → clock tree synthesis → routing → sign-off; Standard cell libraries: the relationship between process node (28nm, 7nm, 3nm), cell height, drive strength, and the power-performance-area (PPA) tradeoff space; Static Timing Analysis (STA): the graph-based algorithm for computing worst-case path delays, accounting for process-voltage-temperature (PVT) corners; CMOS fabrication fundamentals: photolithography, ion implantation, CVD, CMP, and the metal interconnect stack as the physical realization of the GDSII layout; Design for Manufacturability (DFM): fill insertion, double patterning constraints, and lithography-friendly design rules as the interface between design and process; Open-source EDA and the democratization of chip design: OpenROAD, Magic VLSI, and the SKY130 PDK from Google/SkyWater as a fully open tapeout-capable toolchain enabling academic silicon.

Geospatial Data Science & AI (MN-GEO-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Large-Scale Geospatial Data Engineering: The volume problem in modern geospatial data: terabyte-scale satellite archives (Sentinel, Landsat, Planet), billion-point LiDAR datasets, and continuous GPS trajectory streams as the data engineering challenge; Cloud-Optimized GeoTIFF (COG) and Zarr as chunked, indexed raster formats enabling range-request-based partial reads from object storage without full file download; SpatioTemporal Asset Catalogs (STAC) as the RESTful metadata standard for discovering and querying geospatial datasets across providers; Apache Sedona (formerly GeoSpark) and Dask-GeoPandas as distributed spatial dataframe frameworks for parallelizing vector analysis across clusters; Point cloud formats: LAS/LAZ and the Open Point Cloud (OPC) format; spatial tiling schemes (COPC) enabling streaming access to massive LiDAR archives; Data cube architectures: Open Data Cube and xcube as multidimensional array frameworks organizing satellite time series as (time × band × row × col) tensors for efficient temporal analysis.

Unit 2
Spatial Statistics and Geostatistical Modeling: Geostatistics as the branch of spatial statistics dealing with continuous spatial phenomena modeled as realizations of random fields; The variogram as the fundamental tool of geostatistics: empirical variogram computation, theoretical model fitting (spherical, exponential, Gaussian), and the nugget-sill-range parameterization; Kriging as a Best Linear Unbiased Predictor (BLUP): ordinary, universal, and co-kriging variants; the kriging system as a linear algebra problem whose solution weights nearby observations by their spatial covariance structure; Spatial regression models: the spatial lag model and spatial error model as extensions of OLS that account for spatial autocorrelation in the residuals; Geographically Weighted Regression (GWR) as a locally adaptive regression where coefficients vary continuously across space; Point process models: Poisson, Thomas cluster, and Matérn inhibition processes as stochastic models of spatial event locations with applications in crime mapping, ecology, and epidemiology.

Unit 3
Deep Learning for Earth Observation: The computer vision pipeline applied to satellite imagery: the unique challenges of multi-spectral inputs (beyond RGB), large tile sizes, and extreme class imbalance (rare features in vast backgrounds); Semantic segmentation of satellite imagery: U-Net variants with multi-scale feature fusion for land cover mapping, flood extent delineation, and building footprint extraction; Object detection in aerial imagery: rotated bounding box detection (oriented R-CNN) for vehicles, ships, and infrastructure as a rotation-equivariance challenge; Change detection as a spatio-temporal comparison problem: Siamese networks and difference-image classification for detecting deforestation, urban expansion, and disaster damage; Self-supervised pre-training for satellite imagery: masked autoencoders (SatMAE) and contrastive learning on unlabeled image archives to overcome labeled data scarcity; Foundation models for Earth observation: Prithvi (NASA/IBM), Scale-MAE, and GFM as geospatial vision transformers pre-trained on multi-temporal, multi-spectral global archives.

Unit 4
Trajectory Analysis and Movement Data Mining: GPS trajectories as the canonical mobility

data type: raw track compression (Douglas-Peucker algorithm), stay-point detection, and semantic enrichment with map-matching; Map matching: the Hidden Markov Model formulation of snapping noisy GPS points to a road network graph using Viterbi decoding; Mobility pattern mining: frequent route extraction, origin-destination matrix construction, and the DBSCAN-based trajectory clustering algorithm; Urban mobility modeling: gravity models and radiation models as spatial interaction frameworks predicting flow volumes between locations; Activity recognition from trajectory data: inferring transport mode (walking, driving, cycling) from speed, acceleration, and heading features as a time-series classification problem; Privacy in mobility data: the re-identification risk of trajectory datasets, k-anonymity via spatial generalization, and differential privacy mechanisms for publishing aggregate mobility statistics.

Unit 5

Spatial AI for Urban and Environmental Intelligence: Urban sensing as a data fusion problem: integrating satellite imagery, street-level imagery (Mapillary, GSV), social media check-ins, and administrative records into a unified urban analytics platform; Graph Neural Networks for spatial prediction: encoding the city as a graph where nodes are regions and edges encode adjacency or functional similarity for traffic speed forecasting, crime prediction, and real estate valuation; Spatio-temporal forecasting architectures: DCRNN and STGCN as graph-convolutional recurrent models for traffic flow; diffusion convolutional signal propagation on road network graphs; Environmental monitoring with AI: downscaling coarse climate model outputs to fine spatial resolution using super-resolution CNNs; estimating air quality at unmonitored locations via spatial interpolation with satellite covariates; Digital Earth and the Metaverse of infrastructure: CityGML and 3D Tiles as the data standards for city-scale digital twin representations used in urban planning, disaster simulation, and autonomous vehicle HD map construction.

AI & Big Data for Urban Systems (MN-CIV-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1
Urban Big Data Engineering and Platforms: The velocity-volume-variety challenge in urban data: millions of connected devices generating continuous streams across heterogeneous modalities at city scale; Lambda and Kappa architectures for urban data pipelines: batch layer (Hadoop, Spark) for historical reprocessing vs. speed layer (Kafka, Flink) for real-time analytics; Urban data lakes: organizing raw, processed, and curated data tiers in object storage with schema-on-read; Apache Spark for distributed urban analytics: RDD and DataFrame abstractions, spatial UDFs with Sedona, and partitioning strategies for geospatial workloads; Stream processing for urban event detection: sliding and tumbling window aggregations on traffic, utility, and environmental sensor streams; Data quality at scale: automated schema validation, outlier flagging, and lineage tracking in urban data pipelines as a software engineering discipline; Federated urban data ecosystems: data spaces, sovereign data sharing (Gaia-X, FIWARE), and the IDSA connector architecture enabling cross-city data exchange without centralization.

Unit 2
Spatio-Temporal Prediction and Urban Forecasting: Urban forecasting as a spatio-temporal regression problem: demand, flow, and environmental variables as functions of both location and time; Spatio-temporal kriging and covariance separability as the geostatistical baseline for urban prediction; Recurrent architectures for urban time series: LSTM and GRU for multivariate sensor forecasting with spatial covariates; Graph Convolutional Networks (GCN) for network-structured urban data: diffusion convolution on the road graph adjacency matrix for traffic speed and flow forecasting; Spatial transformer networks for demand forecasting: treating the city grid as an image and learning spatial dependencies via convolutional filters; Uncertainty quantification in urban forecasts: conformal prediction intervals for traffic ETAs and energy demand as inputs to risk-aware operational decisions; Hierarchical forecasting: reconciling predictions at neighborhood, district, and city scales using bottom-up and optimal reconciliation methods (MinT).

Unit 3
Reinforcement Learning for Urban Control: Urban control problems as Markov Decision Processes: the state space, action space, reward function, and transition dynamics for traffic signal control, energy dispatch, and shared mobility rebalancing; Multi-agent reinforcement learning (MARL) for city-scale control: decentralized execution with centralized training (CTDE) for coordinating hundreds of intersections simultaneously; Deep Q-Network (DQN) and Proximal Policy Optimization (PPO) applied to adaptive traffic signal control: outperforming Webster's fixed-time plans in simulation benchmarks (CityFlow, SUMO); Demand-responsive transit: RL-based dynamic routing and stop insertion for on-demand bus services; Smart grid demand response as a multi-period stochastic optimization: model predictive control vs. RL for real-time load scheduling under renewable uncertainty; Sim-to-real transfer challenges in urban RL: the distribution shift between calibrated city simulators and the physical complexity of real urban environments.

Unit 4
Urban AI Applications: Vision, NLP, and Multimodal Sensing: Computer vision for urban

intelligence: vehicle and pedestrian detection from CCTV using YOLO-family detectors; crowd density estimation via density map regression; anomaly detection for public safety; Street-level imagery analytics: semantic segmentation of Mapillary and Google Street View imagery for automated urban audit (sidewalk quality, green cover, facade condition) as a scalable alternative to field surveys; NLP for urban governance: topic modeling (LDA, BERTopic) on 311 service requests and social media to extract citizen-reported infrastructure issues and sentiment toward city services; Multimodal fusion for urban situational awareness: combining satellite imagery, street-level video, social media text, and sensor telemetry in a unified representation for emergency response coordination; Noise and air quality mapping: fusing sparse sensor networks with land-use regression models and deep learning to produce high-resolution pollution maps at city scale.

Unit 5

Urban Simulation, Digital Twins, and the Future City: Agent-based models (ABMs) as the simulation paradigm for emergent urban phenomena: agents, behaviors, environments, and the MATSim transport simulation as a city-scale activity-based mobility model; Urban digital twins: the city-scale extension of asset-level twins integrating real-time sensor data, 3D city models (CityGML, 3D Tiles), and simulation engines for scenario planning and operational monitoring; Generative AI for urban design: diffusion models and GANs conditioned on zoning constraints and demographic targets for synthesizing urban layouts, building massing, and land use configurations; Equity-aware urban AI: algorithmic fairness metrics (demographic parity, equalized odds) applied to urban resource allocation, and participatory AI methods for incorporating community input into model design; Autonomous urban systems: self-driving vehicle fleet management, drone delivery network optimization, and robotic last-mile logistics as the next wave of AI integration into urban infrastructure; Climate-adaptive cities: using big data and AI for urban heat island mitigation, flood early-warning systems, and optimizing green infrastructure placement as responses to climate-driven urban risk.

Autonomous Driving & Vehicle Intelligence (MN-EVM-A)

← Back to Scheme

Credits: 2 (2-0-0)

Course Content

Unit 1

The Autonomy Stack: Perception, Prediction, Planning, and Control: SAE Levels 0–5 as a taxonomy of automation capability and the corresponding shift in computational responsibility from driver to system; The modular autonomy stack architecture: perception → object detection and tracking → prediction → planning → control as a sequential dataflow pipeline; End-to-end learning as the alternative paradigm: mapping raw sensor inputs directly to steering and throttle commands via a single neural network; tradeoffs in interpretability, safety certification, and data efficiency; Sensor suite for Level 4 autonomy: the redundant multi-modal configuration (camera ring, long-range radar, 360° LiDAR, HD map localization) and the rationale for sensor diversity as a fault-tolerance strategy; Real-time computing platforms: the NVIDIA DRIVE, Qualcomm Snapdragon Ride, and Mobileye EyeQ SoC families as purpose-built inference and sensor processing hardware; The simulation-to-real gap in autonomous driving: photorealistic sensor simulation (CARLA, LGSVL, Waymo's SimAgents) as the primary scalable data source for training and safety validation.

Unit 2

3D Perception: LiDAR and Multi-Modal Fusion: Point cloud representation and the challenges of 3D deep learning: sparsity, irregular structure, and the absence of a canonical grid; PointNet and PointNet++ as the foundational architectures for permutation-invariant processing of unordered point sets; Voxel-based 3D object detection: VoxelNet and VoxelNeXt as sparse convolution pipelines that discretize the point cloud into a regular voxel grid for efficient 3D CNN processing; Bird's Eye View (BEV) representation as the unified spatial canvas: projecting LiDAR, radar, and camera features into a common top-down feature map for detection and tracking; BEVFusion and BEVFormer as multi-modal fusion architectures that aggregate camera and LiDAR features in BEV space using deformable attention; 3D multi-object tracking: the tracking-by-detection paradigm, Kalman Filter state estimation for object kinematics, and the Hungarian algorithm for data association as the mathematical backbone of the perception stack.

Unit 3

HD Maps, Localization, and Scene Understanding: High-Definition (HD) maps as the prior knowledge layer of autonomous driving: lane geometry, road topology, traffic rules, and semantic annotations at centimeter-level accuracy; Map formats and standards: OpenDRIVE for road geometry, Lanelet2 for semantic lane-level maps, and NDS (Navigation Data Standard) as the industry standard for production map distribution; LiDAR-based localization: Normal Distributions Transform (NDT) and ICP (Iterative Closest Point) as scan-matching algorithms aligning live point clouds to a pre-built HD map; Simultaneous Localization and Mapping (SLAM) for autonomous vehicles: LIO-SAM and LOAM as LiDAR-inertial odometry systems building the map and localizing within it simultaneously; Occupancy grid maps as a probabilistic representation of free, occupied, and unknown space: Bayesian log-odds update rule and the computational tradeoff between resolution and memory; Semantic scene understanding: panoptic segmentation combining instance detection of dynamic objects with semantic labeling

of static infrastructure in a single unified output.

Unit 4

Prediction, Motion Planning, and Decision Making: Agent behavior prediction as the bridge between perception and planning: goal-directed prediction (lane-level intent classification) vs. trajectory prediction (continuous future path regression); Transformer-based prediction architectures: MTR and MotionDiffuser modeling the multimodal, uncertain future distribution over agent trajectories as a generative problem; Motion planning problem formulation: finding a collision-free, comfort-optimizing, rule-compliant trajectory through a dynamic environment; Sampling-based planners: RRT* and its kinodynamic extensions for continuous configuration spaces; optimization-based planners: the path-speed decoupled approach (EM Planner, Apollo) using polynomial curves and QP solvers; Frenet frame planning: transforming the planning problem from Cartesian to a road-aligned coordinate system simplifying lateral-longitudinal decoupling; Hierarchical decision making: behavior layer (FSM or MCTS-based maneuver selection) commanding the trajectory planner as the two-level control architecture separating strategic from tactical decisions.

Unit 5

Safety, Validation, and the Road to Full Autonomy: The safety argument for autonomous vehicles: why traditional testing (miles driven) is statistically insufficient and the case for formal safety cases; Responsibility Sensitive Safety (RSS) as a formal mathematical model defining safe longitudinal and lateral distance thresholds and the proper response to dangerous situations; Runtime monitoring and anomaly detection: out-of-distribution input detection, epistemic uncertainty estimation (MC Dropout, deep ensembles), and the safe fallback (minimal risk condition) trigger logic; Scenario-based testing: the ASAM OpenSCENARIO format for parameterized test scenario specification and the combinatorial explosion problem of corner case coverage; Federated learning for autonomous driving: training perception and prediction models across vehicle fleets without centralizing raw sensor data; Regulatory landscape and AV deployment models: UNECE WP.29 regulations (R157 ALKS), geofenced ODD (Operational Design Domain) restrictions, and the disengagement report as public accountability mechanisms for AV operators.