SOFTWARE SYSTEMS ENGINEERING
ENEX 386

Lecture : 3 | **Year : III**
Tutorial : 2 | **Part : II**
Practical : 1

Course Objectives:
The objective of this course is to introduce the fundamental concepts of software engineering in the context of engineering systems. It focuses on the development of structured software solutions involving system modeling, architectural design, implementation, and testing. The course emphasizes the role of software as part of larger systems, considering integration, constraints, reliability, and real-world applications.

1 Introduction (4 hours)

1.1 Software and its role in engineering systems
1.2 Nature and characteristics of software
1.3 Software application domains
1.4 Legacy software
1.5 Software crisis and myths
1.6 Fundamental principles of software engineering

2 Software Development Process (6 hours)

2.1 Overview of the software development lifecycle
2.2 Process models
  2.2.1 Waterfall model and its extensions
  2.2.2 Incremental process model
  2.2.3 Evolutionary process models (Prototyping, spiral)
2.3 Introduction to agile concepts
2.4 Selection of an appropriate development approach

3 Software Requirements Engineering (6 hours)

3.1 Requirement engineering process
3.2 Functional and non-functional requirements
3.3 Software requirement specification
3.4 Use case-based requirements analysis
3.5 System scope, constraints, and assumptions

4 System Modeling for Engineering Applications (9 hours)

4.1 System modeling and its importance

4.2 Use-case descriptions and diagrams
4.3 Activity diagrams
4.4 Sequence diagrams for interaction modeling
4.5 State-based modeling
4.6 Basics of class diagram

5 System Design and Architecture **(4 hours)**

5.1 Concept of system design
5.2 High-level system architecture
5.3 Modular design, cohesion, and coupling
5.4 Basics of hardware-software interaction architecture

6 Implementation Practices **(3 hours)**

6.1 Coding standards and good practices
6.2 Code readability and documentation
6.3 Version control (Git basics)

7 System Testing **(5 hours)**

7.1 Importance of testing
7.2 Types of testing
7.3 Verification and validation in system testing
7.4 Black-box and white-box testing approach
7.5 Basics of hardware-software integration testing
7.6 Writing basic test cases

8 Software Quality and Maintenance **(4 hours)**

8.1 Quality concepts
8.2 Software maintenance and its types
8.3 Maintenance effort and lifecycle considerations
8.4 Common software issues and improvements

9 Software in Engineering Systems **(4 hours)**

9.1 Software in embedded and communication systems
9.2 Introduction to IoT-based applications
9.3 System constraints and reliability considerations
9.4 Case studies of engineering systems

Tutorial **(30 hours)**

1. Problem understanding, scope definition, and feasibility analysis
2. Selection and justification of the development process
3. Identification of functional and non-functional requirements
4. Preparation of initial SRS
5. Refinement of SRS with constraints and assumptions

6. Identification of actors and use cases
7. Use case diagram and descriptions
8. Activity diagram for system workflow
9. Sequence diagram for interaction modeling
10. Introduction to basic class diagram
11. High-level architectural design
12. Modular design and module interaction
13. Coding standards and code structure discussion
14. Design of basic test cases
15. Final documentation and integration of system artifacts

Practical **(15 hours)**

1. Preparation of the system requirements document using a standard SRS template
2. Construction of use case and activity diagrams using CASE tools
3. Construction of a sequence diagram and a basic class diagram using CASE tools
4. Development of system architecture and module structure using CASE tools
5. Implementation of a functional module following coding standards
6. Execution of test cases, application of version control (Git)

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapters	Hours	Marks distribution*
1	4	5
2	6	9
3	6	8
4	9	13
5	4	5
6, 7	8	10
8, 9	8	10
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Pressman, R. S., Maxim, B. R. (2019). Software engineering: A practitioner's approach. McGraw-Hill Education.
2. Kossiakoff, A., Biemer, S. M., Seymour, S. J., Sweet, W. N. (2011). Systems engineering: Principles and practice. Wiley.
3. Sommerville, I. (2026). Software engineering. Pearson.
4. Mall, R. (2009). Fundamentals of software engineering. PHI Learning.
5. Pfleeger, S. L., Atlee, J. M. (2009). Software engineering: Theory and practice. Pearson

ANALYSIS OF ALGORITHMS
ENCT 388

Lecture : 3 **Year : III**
Tutorial : 2 **Part : II**
Practical : 1

Course Objectives:
The objective of this course is to provide students with a strong foundation in the analysis of algorithm efficiency and computational complexity. It aims to develop understanding of key algorithm design paradigms, including divide-and-conquer, greedy methods, dynamic programming, and backtracking, and covers fundamental concepts of NP-completeness and approximation algorithms.

1 Introduction to Algorithm Analysis (5 hours)

1.1 Algorithm and its properties, RAM model, time and space complexity, detailed analysis of algorithms, Concept of Aggregate Analysis
1.2 Asymptotic notations (Big-O, Big-Ω and Big-Θ), their geometrical interpretations and examples
1.3 Concept of best case, average case and worst case performance of an algorithm
1.4 Modeling algorithms by recurrence relation
1.5 Solving recurrence relation for evaluating computational complexity
  1.5.1 Recursion tree method
  1.5.2 Substitution method
  1.5.3 Using masters theorem

2 Iterative and Numeric Algorithms (8 hours)

2.1 Algorithm for GCD and Fibonacci number
2.2 Sequential search
2.3 Review of bubble sort, selection sort, and insertion sort algorithms
2.4 Number theoretic notations
2.5 Euclid’s and Extended Euclid’s algorithms
2.6 Solving modular linear equations using Chinese remainder theorem
2.7 Fermat’s theorem
2.8 Miller-Rabin randomized primility test and algorithm

3 Divide and Conquer Algorithms (8 hours)

3.1 Binary search, min max finding algorithm
3.2 Analysis of sorting algorithms
  3.2.1 Merge sort

- 3.2.2 Heap sort
- 3.2.3 Quick sort
- 3.2.4 Randomized quick sort

3.3 Order statistics

- 3.3.1 Selection in expected linear time
- 3.3.2 Selection in worst case linear time

4 Greedy Algorithms (7 hours)

4.1 Basic concepts
4.2 Fractional knapsack problem
4.3 Job sequencing with deadlines
4.4 Analysis of minimum spanning trees related algorithms
4.5 Analysis of single source shortest path algorithm

5 Dynamic Programming (8 hours)

5.1 Basic concepts
5.2 All pair shortest path algorithm
5.3 Travelling salesperson problem
5.4 String editing
5.5 0/1 knapsack problem using dynamic programming
5.6 Matrix chain multiplication
5.7 Flow shop scheduling

6 Backtracking Techniques (4 hours)

6.1 Basic concepts
6.2 The N-Queen problem
6.3 Sum of subsets
6.4 Graph coloring
6.5 Hamiltonian cycles
6.6 0/1 knapsack problem using backtracking approach

7 NP-Hard and NP-Complete Problems (5 hours)

7.1 Basic concepts
7.2 Cook's theorem
7.3 NP-Hard graph problems
7.4 NP-Hard scheduling problems
7.5 NP-Hard code generation problems
7.6 Simplified NP-hard problems
7.7 Approximation algorithms: ε- approximation, polynomial time approximation scheme, probabilistically good algorithms
7.8 Vertex cover problem, subset sum problem

Tutorial **(30 hours)**

1. Algorithm analysis
2. Iterative and numeric algorithms
3. Divide and conquer algorithms
4. Greedy algorithms
5. Dynamic programming
6. Backtracking techniques
7. NP-Hard and NP-complete problems

Practical **(15 hours)**

1. Implementation and complexity analysis of iterative, numeric and recursive algorithms
2. Implementation and complexity analysis of greedy algorithms
3. Implementation and complexity analysis of algorithms involving divide and conquer strategy
4. Implementation and complexity analysis of algorithms based on dynamic programming
5. Implementation and complexity analysis of algorithms using backtracking concept

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	5	6
2	8	10
3	8	11
4	7	9
5	8	12
6	4	5
7	5	7
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Horowitz, E., Sahni, S., Rajasekaran, S. (2007). Fundamentals of computer algorithms. Universities Press.
2. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (2022). Introduction to algorithms. MIT Press.
3. Kleinberg, J., Tardos, É. (2006). Algorithm design. Pearson.
4. Skiena, S. S. (2020). The algorithm design manual. Springer.
5. Dasgupta, S., Papadimitriou, C. H., Vazirani, U. V. (2006). Algorithms. McGraw-Hill Education.

AUDIO PROCESSING
ENCT 389

Lecture	**: 3**	**Year : III**
Tutorial	**: 2**	**Part : II**
Practical	**: 1**	

Course Objectives:
The objective of this course is to develop competency in analyzing and processing audio signals in time and frequency domains. It covers fundamentals of acoustics, digital audio representation, spectral analysis and audio effects, along with practical skills in feature extraction and Music Information Retrieval (MIR). Hands-on use of Digital Audio Workstations (DAWs), MIDI, and plugins enables application of audio signal processing techniques in multimedia, speech processing, music technology and emerging digital audio applications.

1 Fundamental of Sound and Audio Signal (7 hours)

1.1 Physical acoustic: Wave propagation, frequency, amplitude, phase
1.2 Psycho acoustic: Pitch, loudness, timbre, dynamics and intensity
1.3 Fundamental of digital audio
1.4 Audio file format: WAV, AIFF, FLAC, ALAC, MP3
1.5 Practical audio quality: Sample rates (44.1khz, 48khz, 96khz) and bit depths

2 Time Domain Audio Processing (8 hours)

2.1 Discrete time audio signal, basic signal operations, system in time domain
2.2 Convolution and impulse response
2.3 Analog and digital filters in audio systems
2.4 Temporal feature extraction

3 Spectral Representation (7 hours)

3.1 DFT and FFT: Derivation and computational complexity
3.2 Windowing function in audio
3.3 Short time Fourier transform: Frame size, hope size and overlap
3.4 Spectrograms: Reading and interpreting time-frequency representation,
3.5 Mel spectrogram: Reading and interpreting; Mel frequency representation
3.6 Constant Q transform and its relevance to music analysis

4 Frequency Domain and Non-Linear Effects (10 hours)

4.1 Sinusoidal model
4.2 Deterministic plus residual model
4.3 Deterministic plus stochastic model

4.4 Basis of non-linear effects
4.5 Dynamic range control: Envelop follow, compressor and limiting, expansion and gating, ADSR, modulation effect
4.6 Time segment process: Time scratching and pitch shifting

5 Music Information Retrieval (6 hours)

5.1 MIR overview: Types of music data, challenges, application
5.2 Spectral centroid and roll off
  5.2.1 Basics of spectral centroid and spectral roll
  5.2.2 Comparison between centroid and roll off
  5.2.3 Application (Timbre analysis, music/ speech classification)
5.3 Mel frequency cepstral coefficient (MFCC)
  5.3.1 Motivation for ceptral features
  5.3.2 Steps in MFCC
  5.3.3 Interpretation of coefficient
  5.3.4 Application (Speech recognition, music classification)
5.4 Chroma
  5.4.1 Concept of pitch classes (12- tone system) mapping
  5.4.2 Chromagram representation,
5.5 Onset and tempo and their applications

6 Audio system and DAW Integration (4 hours)

6.1 Digital audio workstation (DAW)
  6.1.1 Role in audio production
  6.1.2 History and evolution of DAW
6.2 DAW available in market, DAW interface, timeline and arrangement view
6.3 MIDI: Basics, MIDI message structure, VSTi basics,
6.4 Common plugin: Equalizer, compressor, reverb, delay

7 Recent Trend (3 hours)

7.1 Machine learning in audio
7.2 Audio foundation models for bioacoustic and healthcare
7.3 Spatial audio
7.4 Audio DSP ICs
7.5 Generative AI in music

Tutorial (30 hours)

1. Discuss lossless versus lossy compression in the context of audio
2. Derive the convolution sum for a linear time-invariant (LTI) system and explain its physical significance in audio signal processing.
3. Discuss different types of analog filter. Write an advantage of FIR and IIR filter in audio processing
4. Case study discussion
5. Explain the various steps involved in MFCC

6. State the Nyquist theorem and find the Nyquist rate for a signal with maximum frequency 8 kHz.
7. Explain the concept of a chromagram in music information retrieval
8. Explain binaural rendering using head-related transfer functions (HRTFs) and identify one application domain for dedicated audio DSP hardware
9. Music feature analyzer: Input (Audio file), output (Spectrogram, MFCC, tempo)
10. Mini DAW project: Record and edit audio using DAW, apply multiple effects, mix a short track
11. Music classifier: Genre basis classifier

Practical **(15 hours)**

1. Generate sinusoids; Downsample to observe aliasing
2. Measure file size, SNR, and spectral differences between WAV, MP3 (128k/320k), and FLAC
3. Temporal features extraction
4. Spectrogram and STFT
5. Mel spectrogram and CQT
6. MIR features: MFCC
7. DAW: Implement basic features
8. DAW integration and MIDI, VSTi implementation

Final Exam

The questions will cover all the chapters in the syllabus. The evaluation scheme will be as indicated in the table below:

Chapter	Hours	Marks distribution*
1	7	9
2	8	11
3	7	9
4	10	13
5	6	8
6	4	6
7	3	4
Total	**45**	**60**

* There may be minor deviation in marks distribution.

References

1. Christensen, M. G. (2019). Introduction to audio processing. Springer.
2. Smith, J. O. (2011). Spectral audio signal processing. W3K Publishing.
   Zölzer, U. (Ed.). (2011). DAFX: Digital audio effects. Wiley.
3. Müller, M. (2015). Fundamentals of music processing: Audio, analysis, algorithms, applications. Springer.
4. Lyons, R. G. (2011). Understanding digital signal processing. Pearson Education.