ICS 3105
OBJECT ORIENTED SOFTWARE ENGINEERING

COURSE OUTLINE

Course Aim

- To equip learners with concepts, skills and knowledge in Object Oriented software development.

Course Objectives

- To learn and understand various O-O concepts along with their applicability contexts.
- Given a problem, identify domain objects, their properties, and relationships among them.
- How to identify and model/represent domain constraints on the objects and (or) on their relationships.

Course Objectives

- Develop design solutions for problems on various O-O concepts.
- To learn various modeling techniques to model different perspectives of object-oriented software design (UML).

Course Objectives

- To learn software development life cycle for Object-Oriented solutions for Real-World Problems.
- To learn O-O design solutions for the recurring problems

Learning Outcomes

- By the end of this course, the learner should be able to:
  - Discuss how the object-oriented approach may contribute to, or diminish, the risks of software engineering.
  - Specify the structure (statics) and behaviour (dynamics) of a software problem (analysis) and its associated solution (design).

Learning Outcomes

- Apply the UML design notation to analyse and design software.
- Describe and apply design patterns.
- Explain how patterns are constructed from meta patterns.

Learning Outcomes

- Differentiate between structured design and object oriented design.
- Develop and adhere to a process for managing an object-oriented project and creating and evolving analysis/design models.

Course Description

- Object oriented analysis and design. Design and construction of modular, reusable, extensible and portable software. OO Modeling Techniques (the Unified Modeling Language). Design Patterns. Software development process models e.g. “Rational Unified Process”, “Extreme Programming”.

Chapter 1: Software and Software Engineering

- The nature of software
- What is software engineering?
- Software engineering as a branch of the engineering profession
- Stakeholders in software engineering
- Software quality
- Software engineering projects
- Activities common to software projects

Chapter 2 Software Process

- Software process models
- Process activities
- Rational Unified Process.
- Agile software Development.
- Extreme Programming.

Chapter 3 Introduction to Object Orientation

- What is object orientation?
- Classes and objects
- Instance variables
- Methods, operations and polymorphism
- Organizing classes into inheritance hierarchies

Chapter 4 Unified Modeling Language (UML)

- What is UML?
- System models: Context models, Interaction models, Structural models, Behavioral models, Model-driven engineering

Chapter 5 Object Oriented Analysis and Design

- Design and construction of modular, reusable, extensible and portable software.
- Use Case Diagram, Class diagrams, State transition diagrams, Object diagrams, Interaction diagrams, Activity diagrams, Package diagram, Component diagram, Deployment diagram, Object Constraint Language (OCL)

Chapter 6: Design Patterns

- The Abstraction–Occurrence pattern, The General Hierarchy pattern , The Player–Role pattern, The Singleton pattern, The Observer pattern, The Delegation pattern, The Adapter pattern, The Façade pattern, The Immutable pattern, The Read-Only Interface pattern, The Proxy pattern, The Factory pattern, Difficulties and risks when using design patterns

References

- Bernd Bruegge, Allen Dutoit: “Object-Oriented Software Engineering: Using UML, Patterns, and Java”, Addison Wesley.
- Blaha and Rumbaugh “Object-oriented modeling & Design with UML

References

- Eric Gamma, et al. “Design Patterns” Pearson Education.
- Martin Fowler, “UML Distilled”, Pearson Education