

Object Oriented Analysis & Design with UML

Course AD352 – 4 Days

Object oriented analysis and design are key skills necessary to develop robust, maintainable systems. This workshop teaches practical analysis and design skills including processes, techniques and artifacts for specifying systems that satisfy business requirements, adapt well to change and promote reuse. The skills are applicable to small and large projects alike. All of the modeling is done in UML. The participants will learn to develop models in a step-by-step fashion, cross-referencing them in each step, resulting in correct and consistent models.

This workshop assumes that the participants will have knowledge/experience of requirements gathering with Use Cases and understand the fundamentals of Object Technology. The workshop is geared towards Analysts that will be involved in Analysis & Design activities. The Analysts will learn the nuts & bolts of analysis including an introduction to analysis patterns illustrated by well-established domain patterns. The Analyst-Students will also gain practical knowledge of architecture and detailed component design.

Objectives

- Identify and classify objects in the business domain
- Learn key principles that produce quality analysis and design models
- Become proficient in UML
- Apply simple design patterns and realize use cases with sequence and class diagrams
- Understand the concepts and principles of component based architecture
- Partition complex models into sub-systems and components
- Explain Analysis, Design and Architecture patterns

Who Will Attend

Business / System Analysts, Managers, Project Leaders, Developers and anyone who requires a practical knowledge of analysis & design with UML

Instructional Method

The workshop combines lecture, exercises and group discussion. Exercises and case studies will provide students with the opportunity to apply what they learn, and get hands-on experience with the best practices presented in class. The best way to learn is by doing. The split between lecture and exercises/discussion is 50/50.

Prerequisites

Prerequisite for this course are Object Technology Primer and Requirements Analysis with Use Cases, or equivalent experience.

Workshop Content

Analysis & Design with UML – an Overview

- What is Object Oriented Analysis & Design
- Software engineering principles for creating good system models
 - Cohesion
 - Coupling
 - Abstraction
 - Cohesion
 - Information hiding
 - Reuse
- UML overview
- Overview of analysis & design activities and artifacts

Analysis: Getting Started

- Steps of the analysis process
- Analysis artifacts and their interdependencies
- Requirements models as inputs to the analysis activity
- Analyzing use cases

Analysis: Analysis Classes

- What are analysis classes?
- Defining class attributes
- Defining operations
- Object lifecycle: construction, state change and deconstruction
- Analysis class stereotypes
- Identifying candidate classes from use case models

Analysis: Modeling Associations

- Identifying associations and links
- Defining roles and multiplicity
- Associations vs. attributes
- Association classes
- Modeling dependencies

Analysis: Modeling Inheritance

- What is generalization/specialization relationship
- Polymorphism
- Multiple inheritance
- When to use generalization?

Analysis: Organizing Models with Analysis Packages

- What is a package?
- Identifying package dependencies
- Nested packages
- Package stereotypes
- Architectural analysis

Analysis: Dynamic Modeling - Modeling Use Case Realization

- What is use case realization
- Entity, control and interface classes
- Communication diagrams
- Sequence diagrams
- Patterns of object collaboration
- Analysis patterns for dynamic modeling

Analysis: Modeling Object Flow with Activity Diagrams

- What are activity diagrams?
- Action states and transitions
- Decisions
- Forks & joins
- Object flow and signals

Analysis: Modeling Object States with State Diagrams

- What is a state
- Conditions and triggers
- Complex state diagrams
- When to use state diagrams

Analysis: Wrap-up

- Domain modeling
- What are Analysis Patterns?
- Creating robust models with Analysis Patterns
- Keeping the models in sink
- Reviews and walkthroughs

Design: Getting Started

- Transitioning from analysis to design
- UML design artifacts
- Design activity steps
- Artifact traceability
- Design model components

Design: Design Classes

- What are design classes
- Specifying attribute details – visibility data typing, multiplicity, class scope
- Specifying operation details - visibility, parameter and return data typing, class-scope operations
- What makes a good design class?
- Templates and nested classes
- Refractor classes to enhance cohesion

Design: Refining Analysis Relationships

- Modeling aggregation
- Modeling composition
- Aggregation vs. inheritance
- Refining analysis relations
- Representing collections
- Reified relationships

Design: Interfaces and Subsystems

- What are interfaces?
- Component-based development
- Modeling interfaces
- Inheritance vs. interface realization
- Designing with interfaces

Design: Architecture Design with Interfaces and Subsystems

- Modeling subsystems
- Applying interfaces to subsystems
- The Facade Design Pattern
- Architecture design – layers and views
- Things to watch out for

Design: Use Case Realization Design

- Refining the analysis use case realization interaction diagrams
- Messages and operation signatures
- Concurrency and focus of control
- Useful design patterns for assigning behavior
- Modeling subsystem interaction
- Guidelines for loosely coupled and highly cohesive design

Design: Introduction to Patterns

- What are patterns?
- Patterns examples
- Analysis, architecture, design and coding patterns
- Applying patterns to design

Implementation & Deployment: an Overview

- Transitioning from design to implementation and deployment
- Implementation steps & artifacts
- Modeling components with UML component diagrams.
- The UML deployment diagram