

Object-Oriented Analysis & Design using UML 2.0 (TT1130)

Length: 5 days

Description

This course focuses on the advantages of the OO paradigm and domain modeling in reducing the representational gap between a target domain and the software application itself. Minimizing this gap leads to more effective solutions that are both flexible and robust. The modeling notation taught and used in conjunction with the course is the industry standard UML (Unified Modeling Language) 2.0. UML provides a programming language independent framework for the analysis, design, programming and testing of software applications. Using a combination of UML and various techniques for analysis and design, the course relates Object Oriented concepts to modeling complex problems. Models built using these techniques have a very high success rate when turned into working code.

Audience

This course is aimed at developers who specify, design and develop software and applications using traditional/formal/structured methods and want to learn how to apply good practices to OO design and analysis.

Prerequisite

Attendees should have a working knowledge of developing software applications. Designing and analysis experience is also extremely beneficial. This is not a coding class.

Topics

Session: Introduction to Modeling and OOAD

- Building Models
- Notation
- Domains
- The Process of OO Analysis and Design
- Overview of UML 2.0

Session: Classes and Objects

- Objects Provide a Service
- Abstractions
- Responsibilities and Operations
- Overview of GRASP principles
- Messages and Public Interfaces
- Instances
- Classes

- Instantiation
- Encapsulation
- UML Class and Instance Diagramming

Session: Relationships

- Static Relationships
- Dependencies
- Associations
- Navigability
- Whole/Part Associations
- Composition
- Generalization/Specialization Relationships
- Inheritance of Methods and Method Overriding
- Abstract Classes

Object-Oriented Analysis & Design using UML 2.0 (TT1130)

- Dynamic Relationships
- Sequence Diagrams
- Communication Diagrams

Session: Use Cases

- Discovering the Use Cases
- Actors
- Use Case
- Caveats!
- Extending Use Cases
- Generalizations

Session: Use Case Scenarios

- Scenarios
- Primary and Secondary Scenarios
- Essential and Real Scenarios
- Documenting Use Cases and Scenarios
- Use Case Benefits

Session: Conceptual Modeling

- Conceptual Modeling
- Concepts
- Identifying Concepts
- Mapmaking Principles
- Attributes versus Concepts
- Specification or Description
- Associations
- Common Association List

Session: Domain Behavior Modeling

- Domain Behavior Modeling
- Importance of Understanding Dynamic Behavior
- System Sequence Diagrams
- Contracts

Session: Discovering Potential Objects using CRC Cards

- Discovering/Harvesting Objects
- Brainstorming for Classes

- CRC cards & CRC Steps

Session: Static Design Concepts

- Visibility of Attributes and Operations
- Multiplicity of Objects
- Interfaces and Components
- Design Complex Systems from Components
- Identifying "Good" Classes
- Multiplicity of Associations
- Ternary Relationships
- Role and Role Names
- Association Qualification
- Association Class
- Whole/Part Associations
- Extensibility Mechanisms:
- Abstract Classes
- Types and Substitutability
- Polymorphism
- Packages
- Using Packages
- Component Diagrams

Session: Dynamic Design Concepts

- Sequence Diagrams
- Business Rules
- Verifying Completeness
- Advanced Sequencing
- Concurrent Sequences
- Activity Diagrams: Swimlanes

Session: Domain Design

- Iterative Development
- Domain Design
- Detailed Design
- Forming the Architectural vision
- Low Coupling Examined

Session: Detailed Design

- Detailed Design Steps

Object-Oriented Analysis & Design using UML 2.0 (TT1130)

- Detailed Design Activities
- GRASP patterns/principles revisited
- Controller
- Creator
- Information Expert
- Law of Demeter
- Low Coupling/High Cohesion
- Polymorphism
- Pure Fabrication
- Good/Bad packaging principles
- Coupling (allowed and/or required communication paths), layering, and dependencies
- Patterns In Design
- Mapping to Databases
- Mapping to User Interfaces
- About Frameworks
- Designing Components and Interfaces
- Entry and Exit Actions
- Activity
- Statecharts Model a Single Object
- Analysis State Diagrams
- Activity Diagrams: Swimlanes

Session: Summary & Conclusion

- Usage of OO Technology
- Methodologies and Notation
- Management Issues
- Reuse

Session: Remaining UML 2.0 Diagrams

- Use Case Diagrams
- Interaction Diagrams
- Communication Diagrams
- Sequence vs. Communication Diagrams
- State Machine Diagrams
- Statechart Diagram
- Activity Diagram
- Implementation Diagrams

Session: States and Activities

- State Diagrams: Object Lifecycles
- Definitions
- States