

## Core Object-Oriented Analysis & Design using UML 2.0 (TT1100)

**Length:** 3 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.

### Topics

#### INTRODUCTION TO MODELING AND OOAD

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

#### 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

#### RELATIONSHIPS

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

## Core Object-Oriented Analysis & Design using UML 2.0 (TT1100)

### USE CASES

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

### CONCEPTUAL MODELING

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

### DOMAIN BEHAVIOR MODELING

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

### DISCOVERING POTENTIAL OBJECTS USING CRC CARDS

- Discovering/Harvesting Objects
- Brainstorming for Classes
- CRC cards & CRC Steps

### STATIC DESIGN CONCEPTS

- Visibility of Attributes and Operations
- Multiplicity of Objects
- Interfaces and Components
- Identifying "Good" Classes
- Multiplicity of Associations
- Role and Role Names
- Association Class
- Extensibility Mechanisms:
- Abstract Classes
- Types and Substitutability

- Polymorphism

### DYNAMIC DESIGN CONCEPTS

- Sequence Diagrams
- Business Rules
- Verifying Completeness
- Advanced Sequencing
- Concurrent Sequences

### DOMAIN DESIGN

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

### DETAILED DESIGN

- Detailed Design Steps
- Detailed Design Activities
- GRASP patterns/principles revisited
- Controller
- Creator
- Information Expert
- Law of Demeter
- Low Coupling/High Cohesion
- Polymorphism
- Pure Fabrication
- Designing Components and Interfaces

### SUMMARY & CONCLUSION

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