

Course 411: Object-Oriented Analysis & Design using UML (4 days)

Course Description...

Regardless of the software development approach, from the classic waterfall to extreme programming (XP), all of the experts agree that quality software development requires both analysis and design. The Unified Modeling Language (UML) provides a common, standard notation for recording both analysis models and design artifacts. This course delves into the processes of both analysis and design using UML as the notation language.

Learning Objectives...

- Demonstrate the importance of modeling in the software development life cycle
- Become conversant with the UML notation and symbols
- Use the UML notation to build analysis models to create detailed systems specifications
- Employ the UML notation to create effective and efficient system designs

Who should attend...

Software developers, system architects, analysts, designers, and anyone interested in understanding the UML as applied to analysis and design of systems.

Prerequisites...

A good understanding of object-oriented technologies and a basic understanding of analysis and design.



Course Outline...

Unit 1: Analysis, Design, Modeling and the UML

Overview

- Analysis
 - Requirements analysis
 - Systems analysis
- Design
- Modeling
 - Analysis modeling
 - Design modeling
 - Answering questions

The Unified Modeling Language (UML)

Processes and Approaches to OOA&D

- Unified Process
- Rational Unified Process
- Agile approaches
- Other approaches

Unit 2: The Object-Oriented Paradigm

Classes and Objects

- Classes and objects
- Characteristic of objects
 - Structure
 - Inheritance
 - Encapsulation
 - Polymorphism
- Messages and communication
- Associations

Unit 3: Domain Modeling

Data Analysis and Modeling

- Identifying objects and classes
 - Categorizing objects
 - Class types
- Inheritance
- Class structures
 - Aggregations
 - Collections
- Identifying attributes
 - Attribute dependency

Drawing the Object Model

- Objects, attributes and associations
- Links and messages



Unit 4: Use Case and the Behavior Model

Use Case

- Diagrams
- Descriptions

Use Case Extensions

- Use Case relationships
 - Generalization
 - The <<extends>> relationship
 - The <<includes>> relationship
 - Extension points

Unit 5: The Analysis Phase

Requirements Analysis

- Functions, attributes and constraints
- Domain analysis
- Containment and composition

Behavioral Analysis

- Analyzing the Use Cases
 - Object stereotypes
 - Use Case realizations
- Polymorphism, propagation and delegation
- Translating concepts into classes

Defining the Processes

- Collaboration diagrams
- Events and responses
- State charts
- Analyzing with dynamic modeling

Unit 6: Design Phase

Software Classes

- Optimization
- Mapping system functions to objects
- Using collaboration diagrams

Adding the Methods

- Using sequence diagrams
- Assigning methods
 - Method dependency
- Primary methods
- Object/class hierarchy
 - Method encapsulation
- Using partitions to simplify the model
 - Concurrency
- The class diagram



- Design best practices
 - Cohesion
 - Complexity
 - Coupling
 - Congruence

Unit 7: Physical Design

Requirements Allocation

- Persistence

Implementation Diagrams

- Component diagram
- Deployment diagram

Design Considerations

- Reusability
- Scalability
- Testability

Unit 8: Patterns

The Pattern Concept

- Benefits of patterns

Familiar Patterns

- Useful analysis patterns
- Useful design patterns

Unit 9: An Analysis and Design Process

- Understand the domain
- Develop a requirements model
- Analyze the requirements model to produce the analysis model
- Create the design model
- Apply the design model to a physical application

Unit 10: The Bottom Line

Ideas to use

Where to go for more information