Learn how to use Object-Oriented techniques to analyze real-world requirements and to design solutions that are ready to code. Students learn how to identify and design objects, classes, and their relationships to each other, which includes links, associations, and inheritance. A strong emphasis is placed on diagram notation for use cases, class and object representation, links and associations, and object messages. This course utilizes UML 2.0 notation.

**Audience:** Analysts, designers, and programmers responsible for applying OO techniques in their software engineering projects.

**Prerequisites:** Familiarity with structured techniques such as functional decomposition is helpful.

**Number of Days:** 5 days

1. **Course Introduction**
   - Course Objectives
   - Course Overview
   - Using the Workbook
   - Suggested References

2. **Introduction to Analysis and Design**
   - Why is Programming Hard?
   - The Tasks of Software Development
   - Modules
   - Models
   - Modeling
   - Perspective
   - Objects
   - Change
   - New Paradigms

3. **Objects**
   - Encapsulation
   - Abstraction
   - Objects
   - Classes
   - Responsibilities
   - Attributes
   - Composite Classes
   - Operations and Methods
   - Visibility
   - Inheritance
   - Protected and Package Visibility
   - Scope
   - Class Scope

4. **Advanced Objects**
   - Constructors & Destructors
   - Instance Creation
   - Abstract Classes
   - Polymorphism
   - Multiple Inheritance
   - Solving Multiple Inheritance Problems
   - Interfaces
   - Interfaces with Ball and Socket Notation
   - Templates

5. **Classes and Their Relationships**
   - Class Models
   - Associations
   - Multiplicity
   - Qualified Associations
   - Roles
   - Association Classes
   - Composition and Aggregation
   - Dependencies
   - Using Class Models

6. **Sequence Diagrams**
   - Sequence Diagrams
   - Interaction Frames
   - Decisions
   - Loops
   - Creating and Destroying Objects
   - Activation
   - Synchronous & Asynchronous
   - The Objects Drive the Interactions
   - Evaluating Sequence Diagrams
   - Using Sequence Diagrams

7. **Communication Diagrams**
   - Communication Diagrams
Communication and Class Diagrams
Evaluating Communication Diagrams
Using Communication Diagrams

8. **State Machine Diagrams**
- What is State?
- State Notation
- Transitions and Guards
- Registers and Actions
- More Actions
- Internal Transitions
- Superstates and Substates
- Concurrent States
- Using State Machines
- Implementation

9. **Activity Diagrams**
- Activity Notation
- Decisions and Merges
- Forks and Joins
- Drilling Down
- Iteration
- Partitions
- Signals
- Parameters and Pins
- Expansion Regions
- Using Activity Diagrams

10. **Package, Component, and Deployment Diagrams**
- Modeling Groups of Elements – Package Diagrams
- Visibility and Importing
- Structural Diagrams
- Components and Interfaces
- Deployment Diagram
- Composite Structure Diagrams
- Timing Diagrams
- Interaction Overview Diagrams

11. **Use Cases**
- Use Cases
- Use Case Diagram Components
- Actor Generalization
- Include
- Extend
- Specialize
- Other Systems
- Narrative
- Template for Use Case Narrative
- Using Use Cases

12. **Process**
- Process
- Risk Management
- Test
- Reviews
- Refactoring
- History
- The Unified Process
- Agile Processes

13. **The Project**
- Inception
- Elaboration
- Elaboration II
- Construction Iterations
- Construction Iterations - The Other Stuff

14. **Domain Analysis**
- Top View – The Domain Perspective
- Data Dictionary
- Finding the Objects
- Responsibilities, Collaborators, and Attributes
- CRC Cards
- Class Models
- Use Case Models
- Other Models
- Judging the Domain Model

15. **Requirements and Specification**
- The Goals
- Understand the Problem
- Specify a Solution
- Prototyping
- The Complex User
- Other Models
- Judging the Requirements Model

16. **Design of Objects**
- Design
- Factoring
- Design of Software Objects
- Features
- Methods
- Cohesion of Objects
- Coupling between Objects
- Coupling and Visibility
- Inheritance

17. **System Design**
- Design
- A Few Rules

© ITCourseware, LLC
Object Creation
Class Models
Interaction Diagrams
Printing the Catalog
Printing the Catalog II
Printing the Catalog III
Object Links
Associations

18. Refactoring
Refactoring
Clues and Cues
How to Refactor
A Few Refactoring Patterns

Appendix A – UML Syntax
Appendix B – Design by Contract
Contracts
Enforcing Contracts
Inheritance and Contracts

Appendix C – University Summary
Appendix D – Implementations
C++
Java
C#