**Class Description**

**Class Overview**

ExecuTrain’s Object Oriented Analysis and Design (OOAD) with UML training course teaches students how to use object-oriented techniques from requirements gathering to implementation. Learn how to analyze and design classes and their relationships to each other in order to build a model of the business requirements. All the UML diagrams are covered in the course to identify the most suitable diagram suite for your organization. An emphasis is placed on Use Cases to capture and manage requirements and form the basis for test scenarios. Class relationships, through the use of Design Patterns, are leverage to enforce OO concepts that will produce a reusable structural design for your system that adapts well to change. Activity and Sequence diagrams are explored in some detail to model the dynamic aspects of your system. This is a “hands on” course, with students applying their new OOAD and UML skills obtained in the course, to model prospective application solutions.

Although not trying to be tool centric, the course can include the utilization of a UML drawing Tool i.e. Sparx EA or Eclipse Omondo amongst others. In addition the course can also encompass the Agile approach to system development with some possible methodologies listed in our course appendix.

**Class Objectives**

- Understanding requirements, analysis and design workflow
- Identifying classes
- Use diagram notation for use cases, class and object representation, links and associations, inheritance and object messages
- Gain a working knowledge of UML 2.0
- Understand the advantages of Design patterns

What you will learn after completing this course, the student should be able to:

- Approach new complex software development with confidence
- Capture requirements through the utilization of Use Cases
- Utilize the UML diagrams that “best fit” your organization
- Implement the software based on the design
- Establish a software development methodology for in-house development
Table of Contents:

- OO Building Blocks
  - Classes and Objects
  - Operations and Methods
  - Instantiation of Objects
  - Inheritance
  - Overloading
  - Overriding
  - Interfaces
  - Abstract Classes

- Encapsulation
  - Data Hiding
  - Type Hiding
  - Polymorphism

- Associations
  - Dependency and Delegation
  - Aggregation and Composition
  - Coupling
  - Cohesion
  - Redundancy

- UML Overview
  - Structure Type Diagrams
    1. Class Diagram
    2. Object Diagram
    3. Component Diagram
    4. Composite Structure Diagram
    5. Package Diagram
    6. Deployment Diagram
  - Behavior Type Diagrams
    1. Use Case Diagram
    2. Activity Diagram
    3. State Diagram
  - Interaction Type Diagrams
  - Sequence Diagram
- Communication Diagram
- Timing Diagram
- Interaction Overview Diagram

- Use Cases and Use Case Diagrams
  - Capturing and Managing Requirements
  - Abstract Use Cases
  - Actors
  - Anatomy of a Fully Dressed Use Case
  - What are Use Case Diagrams?
  - The include, extends and generalization relationships

- Class Diagrams
  - What does a Class Diagram model?
  - Class Diagram Parts and notation

- Sequence Diagrams
  - What does a Sequence Diagram model?
  - Sequence Diagram Tour
  - Interaction Frames – loop and alt

- Model View Controller (MVC)
  - MVC1 and MVC2
  - Front Controller Pattern
  - Observer Pattern

- Design Patterns
  - Factory Pattern
  - Abstract Factory Pattern
  - Factory Method Pattern
  - Strategy Pattern
  - Command Pattern
  - Template Pattern

- Additional Diagrams – Practical examples
  - Package Diagram
  - Component Diagram
  - Deployment Diagram

Appendices
- Agile Project Development
- SCRUM
• Extreme Programming (XP)
• Dynamic System Development Method (DSDM)
• Open Unified Process (Open UP)
• Using the Sparx Enterprise Architect (EA) Tool for UML Development
• Using the Omondo Tool for UML Development

Please note that this content is meant to be a guideline.
Class material is subject to change and may be presented in a slightly different format than listed.