Object-Oriented Analysis and Design

Course Overview:
The object-oriented approach to analysis and design represents one of the most significant advances in software engineering, and has become the industry standard method for developing modular, scalable, reliable and maintainable systems. The practical application of object-oriented analysis and design requires understanding a new view of software organization, as well as a firm grasp of key principles and practices. Through an effective combination of lecture and hands-on exercises built around a real-world case study, this course provides your architects and developers with the practical knowledge and skills to apply an object-oriented approach to your projects. This course uses UML as a design and modeling language, and presents examples in several object-oriented programming languages including Java and C++.

Who Should Attend:
This course is intended for architects and designers who will be developing applications using an object-oriented approach. A background in software development is assumed.

Course Content:
- What Is Object-Oriented?
  - Comparisons with Functional Design
  - Comparisons with Data-Driven Design
  - A Brief History
- Object-Oriented Concepts
  - Managing Complexity in Software
  - Classes and Objects
  - Object State, Identity, and Encapsulation
  - Inheritance and Polymorphism
  - Responsibilities and Collaborations
  - Components
  - Patterns
- Use Case-Driven Analysis
  - Identifying Roles and Actors
  - Building Use Case Models
  - Detailing Use Cases with Scenarios
- **Hands-On Analysis Case Study Workshop**
- Responsibility-Driven Design
  - Finding Objects
  - Identifying Responsibilities
  - Designing Collaborations
- **Hands-On Design Case Study Workshop**
- Principles of Object-Oriented Design
  - Object Cohesion and Coupling
  - Design by Contract
  - Guidelines for Using Inheritance
  - The Open-Closed Principle
  - The Liskov Substitution Principle
  - The Dependency Inversion Principle
- Object-Oriented Design Patterns
  - Creational Patterns – Factory, Singleton
  - Structural Patterns – Adapter, Façade, Decorator, Composite
  - Behavioral Patterns – Command, Iterator, Observer, State, Strategy, Visitor
- Design Anti-Patterns
  - Duplication
  - Breaking Encapsulation
  - Messaging Issues
- Large-Scale Design
  - Package Design Principles
  - Designing for Reuse
  - Frameworks
- **Hands-On Design Using Patterns Workshop**