+44 7989 401397
info@olsensoft.com

OOAD using UML (5 days)

Course overview

UML is the standard notation for representing object-oriented systems. This course provides comprehensive coverage of UML, describing each of the standard diagram types in detail. The course begins with actors, context diagrams, and use cases. The course then goes on to investigate class diagrams, sequence diagrams, and object interaction diagrams in detail. The course also explores stateful systems, timing, and packaging and deployment considerations.

The course provides detailed coverage of how to transition from analysis into design, and also explores some commonly used object oriented design patterns.

What you'll learn

- Defining actors and use cases
- Defining a class diagram
- Understanding class diagram notation
- Defining scenarios and sequence diagrams
- Modelling state transitions
- Using design patterns

Prerequisites

- Some familiarity with object oriented concepts would be an advantage

Course details

- Object-Oriented Concepts: Objects and messages; Classes; Encapsulation; Inheritance; Polymorphism
- Object-Oriented Methodologies: Software development life cycles; Evolution of methodologies; The UML; Overview of UML diagrams
- Actors and Use Cases: Overview of actors and use cases; Use case diagrams; Identifying and describing actors; Identifying and describing use cases
- Building an Initial Class Diagram: Overview of class diagrams; Gathering requirements; Text analysis; Identifying candidate classes; Adding classes to the class diagram
- Class Diagram Notation: Attributes and operations; Derived attributes; Polymorphic operations; Class associations; Associative roles; Multiplicity
- Object Diagram Notation: Overview of object diagrams; Link attributes; Object interactions; Qualified associations
- Scenarios and Sequence Diagrams: Overview of scenarios and sequence diagrams; Identifying and describing scenarios; Sequence diagram notation; Specifying object creation and destruction

- Refining the Class Diagram: Visibility modifiers; Aggregation; Reflexive aggregation; Operation propagation; Inheritance; Abstract classes and abstract methods; Multiple inheritance
- State Modelling: Modelling object lifetimes; Events and states; Components of a state diagram; State diagrams; Actions and activities; State operations; Dynamic modelling process
- Refining Analysis to Design: Extending and refining the class diagram; Association directionality; Association multiplicity; Ternary associations; Resolving link attributes; Refining inheritance; Designing classes
- Packaging and Deployment: Packaging considerations; Dependency management; Using interfaces to minimize decoupling; Overview of packaging diagrams; Deployment options and considerations
- Overview of Design Patterns: Overview of design patterns; Benefits of design patterns; Taxonomy of patterns; A pattern catalogue; Design pattern notation; Limitations of design patterns
- Design Patterns Explored: Creational patterns; Structural patterns; Behavioural patterns