

# ADE300 : Object Oriented Analysis and Design

**Code :** ADE300

**Duur :** 5 dagen

**Categorie :** Analysis and Design

## Doelgroep :

Deze cursus is bedoeld voor developers en architecten die object georienteerde analyse en design technieken en UML om systemen te ontwerpen willen leren.

## Voorkennis :

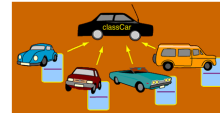
Om aan deze cursus deel te kunnen nemen is kennis van de basis principes van object oriëntatie en ervaring in object-georienteerde software ontwikkeling wenselijk.

## Uitvoering :

De stof wordt behandeld aan de hand van presentatie slides. Tijdens de cursus worden oefeningen gedaan met twee case studies die van requirements tot ontwerp worden uitgewerkt. Een modern UML tool wordt gebruikt om UML diagrammen te tekenen.



## Object Oriented Analysis and Design



## Inhoud :

In deze cursus leren de deelnemers de object georienteerde manier van denken en de technieken voor het analyseren, ontwerpen en modelleren van een software systeem als een verzameling samenwerkende objecten. De UML taal loopt als een rode draad door de cursus heen. Na een introductie en overzicht van de belangrijkste object georienteerde concepten en principes, wordt de moderne methodiek van iteratieve en incrementele systeem ontwikkeling besproken. Vervolgens wordt aandacht besteed aan hoe de requirements van een systeem kunnen worden geanalyseerd en hoe de typische vormen van systeem gebruik kunnen worden beschreven met use cases. Na een overzicht van UML wordt besproken hoe een domain model kan worden opgesteld, hoe de verschillende objecten kunnen worden onderscheiden met hun respectievelijke attributen en relaties en welke informatie ze uitwisselen. Er wordt aandacht besteed aan hoe verantwoordelijkheden kunnen worden toegewezen aan objecten en hoe deze door interactie modellering vertaald en gevisualiseerd kunnen worden met sequence en collaboration diagrammen en state charts. De verschillende design patterns die in dit proces kunnen worden gebruikt worden ook besproken. En ook de vertaling van het domain analysis model naar een design class model is onderdeel van de cursus inclusief het ontwerp van een logische architectuur met packages en subsystems en de mapping naar code. De cursus gaat ook in op architectural design waarbij component en deployment diagrammen worden gebruikt. Tot slot wordt het belang van design patterns voor het implementeren van standaard oplossingen benadrukt.

### Module 1 : Software Process

Software Development Process  
Software Development Phases  
Good Software Characteristics  
Building Good Software  
Capability Maturity Model  
Iterative Development  
Incremental Development  
Requirements Capturing  
Requirements Analysis Process  
System Design  
Test Driven Development  
Waterfall Model  
Evolutionary Development  
Software Errors  
Unified Process  
UP Phases  
OO Analysis and Design

### Module 4 : UML Overview

What is UML?  
UML History  
UML Design Goals  
UML Diagrams  
UML Views  
Use Case View  
Logical View  
Component View  
Deployment View  
Notes and Adornments  
Stereotypes  
Tagged Values  
Constraints  
System Sequence Diagrams

### Module 7 : Statecharts

State Machine Concepts  
State Machine Diagram  
Diagram Elements  
State Machine Usage  
Event Driven Behavior  
State Machines and Objects  
Object Behavior  
Objects and Threads  
Passive Objects  
Active Objects  
Entry and Exit Actions  
Order of Actions  
Internal Transitions  
State Activities  
Guards  
History States

### Module 2 : Requirements Analysis

Understanding Requirements  
Vision Documents  
Requirement Analysis Activities  
Requirement Types  
Functional Requirements  
Non-Functional Requirements  
Requirements Determination  
Requirements Classification  
Requirements Specification  
Conflicting Requirements  
Requirements Risks  
The glossary

### Module 5 : Domain Modeling

Why Domain Modeling?  
Domain Model  
Conceptual Classes  
Noun Identification  
Physical and Conceptual Objects  
Filter Rules  
Similar Conceptual Classes  
Types of Classes  
Domain Analysis Classes  
UML Notation and Classes  
Lowering Representational Gap  
Finding Associations  
Roles in Associations  
Multiplicity and Associations  
Generalization and Specialization  
Aggregation and Composition  
Finding Attributes  
Association Classes

### Module 8 : Design Model

Transition to Design  
From Requirements to Design  
Class Design Diagrams  
The Design Model  
Design Aspects  
Design Model Characteristics  
Mapping to Code  
Packages  
Package Design  
Packaging Guidelines  
Data Access Class Packages  
Subsystems

### Module 3 : Use Case Modeling

Use Cases  
Actors  
Use Case Modeling  
Identifying Actors  
System Context Diagram  
Identifying Use Cases  
Use Case Diagram  
Use Case Modeling Steps  
High Level Use Cases  
Expanded Use Case  
Use Case Template  
Branching with If  
Alternative Paths  
Scenarios  
Structuring Use Case Model  
Generalizations  
include and extends

### Module 6 : Use Case Realization

Realizing Requirements  
System Behavior  
Input Events and Operations  
Sequence of Events  
Event Trace Diagram  
System Sequence Diagrams  
Scenario with Events  
Derivation from Use Case  
Naming System Operations  
Recording System Operations  
Contract Sections  
Postconditions  
Responsibility Driven Design  
Class Responsibilities  
Class Collaborations  
Interaction Diagrams  
Sequence Diagrams  
Grasps Design Patterns

### Module 9 : Architectural Design

System Partitioning  
Large Scale Element Collaboration  
Consider Layering  
Layers and Packages  
Simple Logical Architecture  
Consider Coordination Layer  
Web Application Architecture  
Consider MVC Architecture  
Package Dependencies  
Ordering Work  
Load Test Early  
Clustering  
Vertical Scaling  
Horizontal Scaling  
Physical Architecture

## **Module 10 : Applying Design Patterns**

What are Patterns?

Creational Patterns

Behavioral Patterns

Structural Patterns

Architectural Patterns

Singleton

Abstract Factory

Factory Method

Reducing Dependencies

Observer Pattern

Adapter Pattern

Faade pattern

Proxy Pattern