Object Oriented Analysis en Design

Duur: 5 dagen **Prijs: € 2999**

Open Rooster

Doelgroep Cursus Object Oriented Analysis en Design

De cursus Object Oriented Analysis en Design is bedoeld voor developers en architecten die object georiënteerde analyse en design technieken en UML willen leren.

Voorkennis Cursus Object Oriented Analysis en Design

Om aan de cursus Object Oriented Analysis en Design deel te kunnen nemen is kennis van de basis principes van object oriëntatie en ervaring in object-georiënteerde software ontwikkeling wenselijk.

Uitvoering Training Object Oriented Analysis en Design

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. De cursustijden zijn van 9.30 tot 16.30.

Certificering Object Oriented Analysis and Design

De deelnemers krijgen na het goed doorlopen van de cursus een officieel certificaat Object Oriented Analysis and Design.

Inhoud Cursus Object Oriented Analysis en Design

In de cursus Object Oriented Analysis and Design leren de deelnemers de object georiënteerde 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.

Iteratieve en Incrementele Ontwikkeling

Na een introductie en overzicht van de belangrijkste object georiënteerde concepten en principes, wordt de moderne methodiek van iteratieve en incrementele systeem ontwikkeling besproken.

Requirements Gathering en Uses Cases

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.

Domain Modeling

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.

Interaction Modeling

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.

Packages en Subsystems

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.

Architectural Design

De cursus gaat ook in op architectural design waarbij component en deployment diagrammen worden gebruikt.

Design Patterns

Tot slot wordt het belang van design patterns voor het implementeren van standaard oplossingen benadrukt.

Modules Cursus Object Oriented Analysis en Design

Module 1 : Software Process	Module 2 : Requirements Analysis	Module 3 : Use Case Modeling
Software Development Process Software Development Phases Good Software Characteristics Iterative and Incremental Development Requirements Capturing Requirements Analysis Process System Design Test Driven Development Waterfall Model Evolutionary Development Unified Process	Understanding Requirements Vision Documents Requirement Analysis Activities Requirement Types Functional Requirements Non-Functional Requirements Requirements Determination Requirements Classification Conflicting Requirements Requirements Risks The glossary	Use Cases and Actors Identifying Actors System Context Diagram Identifying Use Cases Use Case Diagram Use Case Modeling Steps High Level Use Cases Alternative Paths Scenarios Generalizations include and extends
Module 4 : UML Overview	**Module 5 : Domain Modeling**	**Module 6 : Use Case Realization**
What is UML? UML Diagrams Use Case View Logical View Component View Deployment View Notes and Adornments Stereotypes Tagged Values Constraints System Sequence Diagrams	Why Domain Modeling? Conceptual Classes Noun Identification Physical and Conceptual Objects Types of Classes Domain Analysis Classes Finding Associations Multiplicity and Associations Generalization and Specialization Aggregation and Composition Finding Attributes	Realizing Requirements System Behavior Input Events and Operations System Sequence Diagrams Derivation from Use Case Postconditions Class Responsibilities Class Collaborations Interaction Diagrams Sequence Diagrams Grasps Design Patterns
Module 7 : Statecharts	**Module 8 : Design Model**	**Module 9 : Architectural Design**
State Machine Concepts State Machine Diagram Event Driven Behavior State Machines and Objects Object Behavior Objects and Threads Passive and Active Objects Entry and Exit Actions Internal Transitions State Activities Guards History States	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	System Partitioning Large Scale Element Collaboration Layers and Packages Simple Logical Architecture Consider Coordination Layer Web Application Architecture Consider MVC Architecture Package Dependencies 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 FaÇade pattern Proxy Pattern		

SpiralTrain BV
Standerdmolen 10, 2e verdieping
3995 AA Houten

info@spiraltrain.nl
www.spiraltrain.nl
Tel.: +31 (0) 30 – 737 0661

Locaties
Houten, Amsterdam, Rotterdam, Eindhoven, Zwolle, Online