Object-Oriented Application Analysis and design for Java™ Technology

Cursusduur: 5 Dagen   Cursuscode: OO-226

Beschrijving:

The Object-Oriented Analysis and Design Using UML course effectively combines instruction on the software development processes, object-oriented technologies, and the Unified Modeling Language (UML). This instructor-led course uses lecture, group discussion, and facilitator-led activities (such as analyzing stakeholder interviews) to present one practical, complete, object-oriented analysis and design (OOAD) roadmap from requirements gathering to system deployment.

Students are provided a pragmatic approach to object-oriented (OO) software development using a widely adapoted methodology (the Unified Process), the latest UML specification (version 1.4), and OO technologies, such as the Java(TM) programming language. This course progresses through: a primer on OO technology and software development methodologies, requirements gathering and analysis (including interviewing stakeholders), system architecture and design, implementation, testing, and deployment. The classroom lectures expose students to other proven OOAD practices, such as class-responsibility- colaboration (CRC) analysis (used to discover the Domain entities) and Robustness analysis (used to move from analysis to design).

Doelgroep:

Students who can benefit from this course are system architects, software engineers, systems analysts, and designers responsible for the conception and creation of object-oriented software applications.

Doelstelling:

- Upon completion of this course, students should be able to:

- Describe the object-oriented software development process, including object-oriented methodologies and workflows.

- Gather system requirements through interviews with stakeholders.

- Analyze system requirements to determine the use cases and domain model of the problem domain (the Requirements model).

- Create a system architecture (the Architecture model) supporting the nonfunctional requirements (NFRs) and development constraints.

- Create a system design (the Solution model) supporting the functional requirements (FRs).

Vereiste kennis en vaardigheden:

To succeed fully in this course, students should be able to:

- Understand object-oriented concepts and methodology
- Demonstrate a general understanding of programming, preferably using the Java programming language
- Understand the fundamentals of the systems development process

Vervolgcursussen:

Related courses before

- SL-275: Java Programming Language (SL-275)

Related courses after

- SL-285: Java Programming Language Workshop (SL-285)
- SL-425: Architecting and Designing J2EE Applications (SL-425)
- SL-500: J2EE Patterns (SL-500)

Cursusinhoud:

- Introducing the Software Development Process
- Describe the Object-Oriented Software Development (OOSD) process
- Describe how modeling supports the OOSD process
- Explain the purpose, activities, and artifacts of the following OOSD workflows: Requirements Gathering, Requirements Analysis, Architecture, Design, Implementation, Testing, and Deployment
- Examining Object-Oriented Technology
- Describe how OO principles affect the software development process
- Describe the fundamental OO principles
- Choosing an Object-Oriented Methodology
- Explain the best practices for OOSD methodologies
- Describe the features of several common methodologies
- Choose a methodology that best suits your project
- Determining the Project Vision
- Interview business owners to determine functional requirements of the software system
- Analyze interview results to identify NFRs, risks, and constraints
- Create a project Vision document from the results of the interviews and risk analysis
- Gathering the System Requirements
- Plan for the process of gathering requirements
- Plan for the stakeholder interviews to validate and refine the FRs and NFRs from the Vision document
- Document the system in the System Requirements Specification (SRS) from all requirements sources
- Creating the Initial Use Case Diagram
- Identify and describe the essential elements in a UML Use Case diagram
- Develop a Use Case diagram for a software system based on the SRS
- Record Use Case scenarios for architecturally significant Use Cases
- Refining the Use Case Diagram
- Document a Use Case and its scenarios in a Use Case form
- Recognize and document Use Case and Actor inheritance
- Recognize and document Use Case dependencies

- Identify the essential elements in an Activity diagram
- Validate a Use Case with an Activity diagram
- Determining the Key Abstractions
- Identify a set of candidate key abstractions
- Identify the key abstractions using CRC analysis
- Constructing the Problem Domain Model
- Identify the essential elements in a UML Class diagram
- Construct a Domain model using a Class diagram
- Identify the essential elements in a UML Object diagram
- Validate the Domain model with one or more Object diagrams
- Creating the Analysis Model Using Robustness Analysis
- Explain the purpose and elements of the Design model
- Identify the essential elements of a UML Collaboration diagram
- Create a Design model for a use case using Robustness analysis
- Identify the essential elements of a UML Sequence diagram
- Generate a Sequence diagram view of the Design model
- Introducing Fundamental Architecture Concepts
- Justify the need for the architect role
- Distinguish between architecture and design
- Describe the SunTone Architecture Methodology
- Exploring the Architecture Workflow
- Describe the Architecture workflow
- Describe the diagrams of the key architecture views
- Select the Architecture type
- Create the Architecture workflow artifacts
- Creating the Architectural Model for the Client and Presentation
- Explore user interfaces

- Document a graphical user interface (GUI) application in the Client tier of the Architecture model
- Document a web user interface (Web UI) application in the Presentation tier of the Architecture model
- Creating the Architectural Model for the Business Tier
- Explore distributed object-oriented computing
- Document the Business tier in the Architecture model
- Creating the Architectural Model for the Resource and Integration Tiers
- Document the persistence mechanism in the Resource tier of the Architecture model
- Document the persistence integration mechanism in the Integration tier of the Architecture model
- Creating the Solution Model
- Create a Solution model for a GUI application
- Create a Solution model for a Web UI application
- Refining the Domain Model
- Refine the attributes of the Domain model
- Refine the relationships of the Domain model
- Refine the methods of the Domain model
- Declare the constructors of the Domain model
- Applying Design Patterns to the Solution Model
- Define the essential elements of a software pattern
- Describe the Composite pattern
- Describe the Strategy pattern
- Describe the Observer pattern
- Describe the Abstract Factory pattern
- Modeling Complex Object State Using Statechart Diagrams
- Model object state
- Describe techniques for programming complex object state

Nadere informatie:

Neem voor nadere informatie of boekingen contact op met onze Customer Service Desk 030 - 60 89 444

info@globalknowledge.nl

www.globalknowledge.nl

Iepenhoeve 5, 3438 MR Nieuwegein