

## UML 2: A Comprehensive Hands-On Introduction - 3 dagar

*kurser 216*

### **You Will Learn How To**

- Model software and nonsoftware systems using UML 2
- Capture and document user requirements using use cases
- Generate and interpret UML models using the complete diagramming notation
- Use CASE tools to build and manipulate fully featured UML models
- Ensure consistency and accuracy throughout all diagrams
- Represent design patterns in UML

### **Course Benefits**

The Unified Modeling Language (UML) is the industry-standard notation for producing the models of a system. In this course, you learn to generate and interpret UML models as applied to a wide range of activities using the significant extensions and enhancements of UML 2. These skills are put into practice using a market-leading CASE tool.

### **Who Should Attend**

Designers, programmers, project managers and all other personnel involved in systems development. UML practitioners who wish to update their skills to UML 2 will also benefit. Knowledge of object-oriented techniques is helpful but not required.

### **Hands-On Training**

You gain hands-on UML experience using CASE tools. Exercises include:

- Modelling system requirements and business processes with use cases
- Forward- and reverse-engineering between UML models and code
- Representing system structure using class and object diagrams
- Modelling behaviour with interaction, state machine and activity diagrams
- Generating HTML and textual documentation
- Producing interrelated diagrams in a large system model

## UML 2: A Comprehensive Hands-On Introduction - 3 dagar

*kurser 216*

### Introduction to UML

#### Speaking a common language

- The importance of modelling
- Enabling concise communication
- The evolution of UML

#### Elements of UML

- Building blocks: things, relationships and diagrams
- Architectural views: use case, design, implementation, process and deployment
- Levels of detail: visualisation, specification and construction

#### Object-oriented concepts

- Objects and classes
- Links and relationships
- Inheritance and polymorphism

#### Modelling the Structure of a System

##### Specifying classes

- Modelling user-defined types as classes
- Representing information as attributes
- Representing functionality as operations

##### Identifying relationships between classes

- Dependencies
- Associations
- Aggregation and composition
- Generalisation

##### Object and class diagrams: the core of UML

- Showing classes and their relationships
- Depicting snapshots using object diagrams
- Defining information models with class diagrams

#### Modelling the Behaviour of a System

##### Use case diagrams: describing user requirements

- Representing systems boundaries
- Actors and use cases
- Notations for refinement

##### Sequence and communication diagrams: depicting typical event scenarios

- Events and signals
- Showing time-ordered behaviour
- Expanding use cases into the developers' view
- Converting between sequence and communication diagrams

##### Expressing real-time aspects

- Synchronous/asynchronous messages

- Representing timing constraints and transmission delays
- Implementing timing diagrams

#### Representing State Machines

##### State machine diagrams: capturing state-dependent behaviour

- States, transitions and events
- Concurrent substates
- History and synch states

##### Activity diagrams: specifying behavioural logic

- Modelling workflows
- Partitioning activities using swimlanes
- Concurrency and synchronisation of parallel activities

#### Architectural Modelling

##### Packages and interfaces

- Distinguishing between classes/interfaces
- Exposing class and package interfaces
- Subscribing to interfaces

##### Component and deployment diagrams

- Describing dependencies
- Deploying components across threads, processes and processors
- Describing internal structure using composite structure diagrams

##### Design patterns

- Patterns, mechanisms and frameworks
- Representing design patterns
- Referencing design patterns

#### Applying UML

##### Model-Driven Architecture (MDA)

- The Meta-Object Facility (MOF)
- Common Warehouse Meta-model (CWM)

##### Life cycle stages

- Using UML within the Unified Process
- Modelling business processes
- Capturing requirements
- Systems analysis
- Software design