

User and System Requirements for Successful Software Development - 4 Days

Course 218 Overview

- You Will Learn How To**
- Develop requirements for software-intensive systems using proven methodologies
 - Build a use case-based requirements model
 - Write user stories and brief, casual, fully developed use cases
 - Enhance and refine use cases using an iterative approach
 - Model user interface using mock-ups and develop a data model
 - Validate requirements, manage the changes and keep traceability
- Course Benefits** Requirements gathering is the cornerstone of any software development project. In this course, you gain the knowledge and skills needed to capture software requirements using clearly defined processes. You learn to specify user and system requirements, match the process to the size of your project, and apply quality and consistency tests to the requirements model.
- Who Should Attend** Those developing, leading, designing, testing or managing a requirements initiative for a software system. UML experience is not required. Those responsible for identifying user requirements in a nonsoftware development environment should take Course 315, "Developing User Requirements".
- RealityPlus™** Extensive PC-based activities throughout the course immerse you in a realistic user requirements environment, providing practical experience in constructing a software requirements model. Activities include:
- Capturing stakeholder input from video scenarios that put you at the meeting table
 - Modelling requirements with UML diagrams using a leading CASE tool
 - Capturing, structuring and refining use cases in a realistic simulated environment
 - Developing screen mock-ups with an interface simulation
 - Producing a UML requirements data model
 - Validating requirements using IEEE standard checklists
 - Performing inspections on real-world use case documents

User and System Requirements for Successful Software Development - 4 Days

Course 218 Outline

The Importance of Software Requirements

The software development life cycle

- Defining and differentiating between requirement types
- Locating requirement sources
- Development approaches

Presenting software requirements

- Structuring the requirements document
- Requirements components: text, diagrams, data

Structuring Your Project

Tuning your methodology to your project size

- Matching the process to small, medium and complex systems
- Differentiating agile from standard techniques

Analysing stakeholder input

- Identifying and prioritising stakeholders
- Eliciting initial requirements from input documents
- Iterating requirements collaboratively

Applying the requirements process

- Elicitation
- Analysis
- Specification
- Validation
- IEEE
- SWEBOK
- The Unified Process

Capturing and Refining Use Cases

Writing user stories

- Scripting user stories and brief versions of use cases
- Iteration and progressive elaboration of use cases

Creating structured use cases

- Use cases as behavioural requirements
- Identifying stakeholders and actors
- Naming and scoping use cases
- Writing scenarios: main and alternatives
- Adding preconditions and guarantees

Iterating use cases

- Refining use cases with stakeholders
- Factoring common steps
- Discovering extension scenarios
- Verifying use case completeness

Organising use cases

- Diagramming scenarios with UML

- Choosing between free text and formal use case notation

Generating Interface Requirements

Integrating interface requirements

- Supporting use cases with user interface mock-ups
- Comparing types of interface

Producing interface models

- Storyboarding and prototyping
- Modelling interfaces with UML state diagrams and navigation maps

Data Requirements

Analysing data requirements

- Exploring the use cases and the interface
- Determining data business rules

Creating a requirements data model

- Representing data models with UML class diagrams
- Entities
- Attributes
- Associations
- Adding associations' multiplicity
- Maintaining the glossary

Nonfunctional Requirements

Gathering nonfunctional requirements

- Obtaining volumetrics
- Classifying nonfunctional requirements using FURPS

Documenting nonfunctional requirements

- System reliability: Availability, Accuracy and Failures
- Addressing the "-ilities"

Validating Requirements and

Producing Test Scenarios

Performing requirements validation

- Achieving well-formed requirements through validation
- Reviewing requirements with walkthroughs
- Verifying requirements with inspections

Generating use case tests from requirements

- Ensuring testability of requirements
- Extrapolating test scripts and test scenarios from requirements
- Relating requirements to system and UA testing

Managing Changing Requirements

- Developing a process for managing requirements
- Negotiating changes using a Change Control Board (CCB)
- Confirming requirements through a traceability matrix