



EFFECTIVE USE CASE DEVELOPMENT (4 DAYS)

DESCRIPTION

This class presents an up-to-date, practical guide to use case writing. The class expands on the classic treatment of use cases to provide software developers with a "nuts-and-bolts" tutorial for writing. The course thoroughly covers introductory, intermediate, and advanced concepts in use case development. During the class the instructor will use examples of both good and bad use cases to reinforce the student's learning.

OBJECTIVES

At the completion of this course, the student will be able to:

- Understand the key elements of use cases
- Understand stakeholders, design scope and scenarios
- Develop a use case style guide with action steps and suggested formats
- Use an extensive list of time-saving use case writing tips
- Develop a helpful presentation of use case templates
- Develop a proven methodology for taking advantage of use cases

TOPICS

- Introduction
- The use case as a contract for behavior
- Scope
- Stakeholders
- Three named goal levels
- Preconditions, triggers, and guarantees
- Scenarios and steps
- Extensions
- Linking use cases
- Formats to choose from
- On being done
- Scaling up to many use cases
- CRUD and parameterized use cases
- Business process modeling
- The missing requirements
- Use cases in the overall process
- Mistakes fixed

AUDIENCE

This course is designed for analysts, software engineers, application experts, and technical project managers.

PREREQUISITES

Students should have a general understanding of object-oriented analysis and design concepts. Students that have attended an object-oriented analysis and design course have fulfilled this requirement. Basic computer skills and a familiarity with Windows-based applications are also a must.

COURSE OUTLINE

- I. Introduction
 - A. What is a use case?
 - B. Requirements and use cases
 - C. Use Cases as project-linking structure
 - D. When use cases add value
 - E. Manage your energy

- II. The Use Case as a Contract for Behavior
 - A. Interactions between actors with goals
 - B. Contract between stakeholders with interests
 - C. The graphical model

- III. Scope
 - A. Functional scope
 - B. Design scope
 - C. The outermost use cases
 - D. Using the scope-defining work products

- IV. Stakeholders
 - A. The primary actor
 - B. Supporting actors
 - C. The system under discussion
 - D. Internal actors and white-box use cases

- V. Three Named Goal Levels
 - A. User goals (blue, sea-level)
 - B. Summary level (white, cloud/ kite)
 - C. Subfunctions (indigo/black, underwater/clam)
 - D. Using graphical icons to highlight goal levels
 - E. Finding the right goal level
 - F. A longer writing sample: "handle a claim" at several levels

- VI. Preconditions, Triggers, and Guarantees
 - A. Preconditions
 - B. Minimal guarantees
 - C. Success guarantee
 - D. Triggers

- VII. Scenarios and Steps
 - A. The main success scenario
 - B. Action steps

- VIII. Extensions
 - A. Extension basics
 - B. The extension conditions
 - C. Extension handling

- IX. Linking Use Cases
 - A. Sub use cases
 - B. Extension use cases

- X. Formats to Choose From
 - A. Forces affecting use case writing styles
 - B. Standards for five project types
 - C. Conclusion

- XI. On Being Done

- XII. Scaling Up to Many Use Cases
 - A. Say less about each one (low-precision representation)
 - B. Create clusters of use cases

- XIII. CRUD and Parameterized Use Cases
 - A. CRUD use cases
 - B. Parameterized use cases

- XIV. Business Process Modeling
 - A. Modeling versus designing
 - B. Linking business and system use cases

- XV. The Missing Requirements
 - A. Precision in data requirements
 - B. Cross-linking from use cases to other requirements

- XVI. Use Cases in the Overall Process
 - A. Use cases in project organization
 - B. Use cases to task or feature lists

- C. Use cases to design
- D. Use cases to UI design
- E. Use cases to test cases
- F. The actual writing

XVII. Mistakes Fixed

- A. No system
- B. No primary actor
- C. Too many user interface details
- D. Very low goal levels
- E. Purpose and content not aligned
- F. Advanced example of too much UI