UML for Java Programmers

Course Summary

Description

This course is designed as a broad overview of the UML, with specific emphasis on reading the UML notation and accurately translating design into code. This course is designed for software Implementers, not Designers and Architects. Therefore, the course is focused on aspects of developing code in strict accordance with the design. The majority of the course is composed of lectures and exercises in Rational Rose designed to enhance an Implementers understanding of the UML input artifacts from design to construction.

Participants will be exposed to all major UML design artifacts, and strategies for code development from each design construct for Java. Participants will learn how to do code generation, reverse engineering, and roundtrip engineering in Rational Rose for Java. RUP methods will be referenced for iterative development, stressing the importance of design and code synchronization.

Objectives

At the end of this course, students will be able to:

- Describe the UML design artifacts– what they are, when they are produced, by whom, and for what purpose.
- Describe the purpose of use cases and how they relate to code.
- Interpret Activity and Use Case Diagrams.
- Translate the static elements of design into code: Class Diagrams and Specifications
- Translate the relational elements of design into code: generalization, association, aggregation
- Differentiate between the data model and the object model
- Translate the dynamic elements of design into code: Sequence Diagrams, State Diagrams, Collaboration Diagrams
- Translate the detailed elements of Class Specifications into code: methods, attributes, relationships (types, export control, stereotypes)
- Synchronize code and design using forward, reverse, and roundtrip engineering

Topics

- Introduction to UML
  Use Case View: Interpreting Use Case Documents
- Use Case view: Interpreting Use Case Flows
- Logical View: UML Static Models
- Code Generation
- Reverse Engineering and Roundtrip Engineering with Rational Rose
- Object Models and Data Models
- Logical View: UML Dynamic Models
- Implementation View
- Process View
- Deployment View

Audience

This course is intended for Implementers only.

Prerequisites

There are no prerequisites for this course.

Duration

Two days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

ProTech
protechtraining.com

"Charting the Course ...

... to Your Success!"

UML for Java Programmers

Course Outline

I. Introduction to UML
- A. Purpose and Background
- B. 4+1 model architecture review
- C. Logical
- D. Implementation
- E. Process
- F. Deployment
- G. Use Case
- H. RUP Artifact Set
- I. RUP Design Package
- J. UML References

II. Use Case View: Interpreting Use Case Documents
- A. Use Case diagrams
- B. Use Case documents

III. Use Case View: Interpreting Use Case Flows
- A. Activity diagrams
- B. Use Case Realizations
- C. Relationships between the parts of the use case view

IV. Logical View: UML static models
- A. What is a static model?
- B. Class Specifications/Diagrams
- C. Stereotypes
- D. Relationships
- E. Association
- F. Aggregate
- G. Composite
- H. Generalization
- I. Roles
- J. Attributes
- K. Operations
- L. Method Specification
- M. Packages
- N. Dependencies
- O. Control Units

V. Code Generation
- A. Using Rational Rose to generate Java
- B. Parts of the model, which generate code
- C. Forward engineering practices
- D. Setting model properties for code generation
- E. Creating code for languages, which are not object-oriented

VI. Reverse Engineering and Roundtrip Engineering with Rational Rose
- A. Reverse engineering practices
- B. Roundtrip engineering practices

VII. Object Models and Data Models
- A. Relationship between the models
- B. Generating a database schema from a data model in Rational Rose
- C. Reverse engineering DDL's in Rational Rose

VIII. Logical View: UML dynamic models
- A. What is a dynamic model?
- B. Object models
- C. Sequence diagrams
- D. Collaboration diagrams
- E. State diagrams

IX. Implementation View
- A. Component diagrams
- B. Expressing the implementation architecture in component diagrams

X. Process View:
- A. Component diagrams for process architecture
- B. Inter-process communication on sequence diagrams

XI. Deployment View
- A. Deployment diagrams
- B. Expressing the physical architecture in deployment diagrams
- C. How deployment diagrams influence the code

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically