5 days

0061

This course is available with several options. Please contact us to discuss

UML for Software Development using Enterprise Architect

The Unified Modelling Language has become one of the industry standard notations for the analysis and design of IT systems. This course uses the UML to illustrate OO analysis and design principles. Participants receive a solid grounding in OO technology and are then taken through the software development cycle, addressing essential elements such as Testing and Database ER design. There are extensive exercises and opportunities for discussion. This is an intensive course showing how to design robust OO systems. The end-to-end process of Requirements capture, Use Case creation, SDLC selection, OO analysis and design, through to Testing, Acceptance and Delivery is emphasized ,using Enterprise Architect, enabling participants to deliver high-quality surprise-free systems. All exercises come with worked solutions and delegates receive a copy of "UML Distilled" by Martin Fowler

Course Objectives

- Explain Object Oriented design and development techniques and terminology
- Show how OO Analysis and Design relates to traditional techniques
- Describe how the use of the UML for modelling fits with OO technology for software development
- Show how the use of Abstraction, Inheritance and Polymorphism can simplify programs.
- Describe Requirements capture and their management with Use Cases
- Present Enterprise Architect for use throughout the lifecycle
- Explain why Dynamic modelling leads to more complete solutions
- Show how to design successful normalised and optimise de-normalised relational databases, relating to our classes
- Show how UML is used to model business processes and avoid inconsistent design.
- Show how complex business rules can be captured in Decision Tables and how these are introduced into Use Cases
- Describe the phases and workflows of the unified process
- Show how UML is used to manage a smooth transition from Analysis to Design and Implementation
- Present Package and Architecture modelling features
- Present Patterns as a way of simplifying programs and maximising reuse.
- Show how awareness of state transitions can help produce bullet-proof code
- Show how Testing is implemented throughout a UML-based project

Audience

- Management wanting to understand the project issues of OO and SDLCs
- Technicians wanting to understand analysis and design in a UML world
- Senior staff needing to examine the potential of UML for their organization
- Anyone concerned with Requirements Gathering and Program Design

Prerequisites

- General knowledge of software development process.
- Keen to provide more professional IT services

Timetable

Register at 09:00 for 09:30 start on Day1, 09:00 the rest of the course.
Finish at 17:00 on all days.

Presentation Style

Lectures, demonstrations, group discussions and hands-on exercises

5 days *006*	# UML for Software Development using Enterprise Architect

The course covers:

- **Introduction to Software Development Processes**

- Process Alternatives
- The Rational Unified Process
- Faults in our Processes
- Iteration as a General Technique
- Agile processes in general and XP, SCRUM and Spiral in particular

- **Object Orientation in Business**

- What is OO?
- Benefits of Object Technology
- Abstraction
- Objects and Classes
- Objects and State
- Polymorphism and inheritance

- **Advanced OO Analysis**

- Analysis versus Design
- The analysis phase
- Domain modelling
- CRC Techniques
- Concrete and Abstract design
- Inheritance Pitfalls
- Interface versus implementation

- **UML Overview**

- Origins and purpose of UML
- A walkthrough of the UML diagrams
- Using UML diagrams
- What's new in UML 2.0 and why
- The UML Meta Language
- OCL

- **Enterprise Architect Introduction .. used throughout remainder of course**

- EA structure
- Requirements and Use Cases
- The Class diagrams
- Automatic code generation
- Forward/reverse engineering
- Architecture modelling

- **Decision Tables**

- Capturing complex requirements
- Completeness Checking
- Business Terminology
- Code Generation from Tables
- Decision Tables in Use Cases

- **Use Cases**

- What are Use Cases
- Constructing Use Case diagrams
- Requirements and Use Cases
- Documenting Use Cases
- Alternates and Exceptions
- Scenarios
- Test Case creation

- **Static Modelling**

- Classes and interfaces
- Class diagrams
- Object diagrams
- Class relationships
- Robustness analysis and Class diagrams
- Association
- Aggregation and composition
- Association classes
- Inheritance
- Abstract classes
- Interfaces
- Stereotypes
- Constraints

- **Dynamic Modelling**

- Introduction to UML dynamic modelling
- Activity diagrams
- Sequence diagrams
- Collaboration diagrams
- State charts and State tables
- Generating code from tables

- **Objects and Databases**

- Object Persistence
- Mapping Classes to Relational Tables
- Proxies
- Factory Classes
- OO Databases
- Relational Databases
- Object-Relational
- JDO, Hibernate and related mapping tools

- **Component and Deployment Models**

- Using Packages to organise your system
- Using components to model physical organisation
- Modelling hardware architecture

- **What are Patterns?**

- Designing with Patterns
- Where to find Patterns
- Frameworks
- Some Core Patterns
- Delegation
- Factory
- Singleton
- Decorator
- Pros and Cons of Patterns

- **Testing**
- From Use Cases to Acceptance tests
- Class-oriented testing
- Test Frameworks
- Test-driven development
- Test metrics

- **Summary and Conclusion**

- A strategy for Beginning
- Where to go for Information
- Avoiding mistakes
- Commonsense approach to introducing the UML