incusdata (pty) ltd. 1995/04149/07
Coding · Matters
PO Box 148
Wierda Park, 0149
tel : +27 12 666 2020
info@incusdata.com
www.incusdata.com

Price: R12,200.00 excl. VAT
Duration: 3 days
Delivery: Virtual classroom or on-site training

Object-Oriented Analysis

Description

Object Orientation (OO) is a methodology that is used for the whole software development life cycle: from analysis of users' needs, to design and then coding and testing. The Unified Modelling Language (UML) is a standard set of diagrams for modelling your system using an OO approach.

This course is ideal for analysts who work closely with OO developers. It will teach you how to analyse and document your system using OO principles and UML diagrams. It will help you to understand the problem and the solution better. You will also be able to communicate more effectively with your developers.

This course is run in parallel with the first 3 days of the Object-Oriented Analysis & Design course. If you have programming experience, you will benefit more from attending the full 5 days.

Objectives

After you have completed the Object-Oriented Analysis using UML course, you will be able to:

- Understand object-oriented concepts and principles, and the OO project lifecycle.
- Analyse a system in terms of the objects involved
- Read and write use cases.
- Take part in a design session that uses Class, Responsibilities, and Collaboration (CRC) cards.
- Identify high-level classes and the relationships between them.

Intended Audience

You should attend the Object-Oriented Analysis using UML course if:

- You are a business analyst, project manager or system architect, and you work with developers who use object orientation.
- You need to learn how to read and write use cases.
- You need to understand the object-oriented process.

Prerequisites

You do not need to have any knowledge of object orientation or programming before attending the Object-Oriented Analysis using UML course.

You should, however, have some experience working with systems and analysing user requirements.

Course Contents

Introduction

- The evolution of the object-oriented paradigm.
- OOP compared to other software development paradigms.
- Advantages and disadvantages of OOP.

Object-Oriented Concepts and Terminology

- Classes and objects.
- Attributes and behaviours.
- Data abstraction and encapsulation.
- Polymorphism.
- Inheritance and code reuse.
- Associations and relationships between classes.

Unified Modelling Language (UML)

- History and evolution of the UML.
- Overview of UML diagrams: use case, activity, class, object, sequence, communication, state, component, package, timing, composite structure, interaction overview, deployment diagrams.
- Common extension mechanisms.
- UML modelling tools.

Object-Oriented Methodologies

- Traditional Software Development Lifecycle (SDLC).
- Iterative and incremental development (IID).
- The need for an OOAD process.
- The Rational Unified Approach (RUP).
- The Iconix method.
- Agile Modelling.
- Extreme Programming.

Object-Oriented Analysis

- Behaviour analysis and use cases.
- Use case text vs use case diagrams.
- Activity diagrams.
- Domain modelling.
- Class identification and domain classes.
- CRC cards and CRC sessions.
- Which UML diagrams to use during analysis.

** *The lecturer reserves the right to modify the contents of the course to suit the needs of the delegates.*