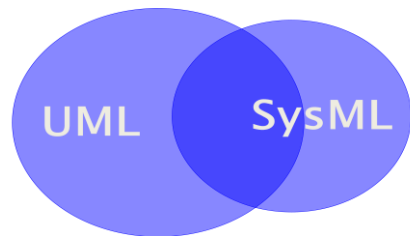


UML in Context

The Unified Modelling Language™ - UML is the Object Management Group's (OMG) most-used specification, and is used widely to model not only application structure, behaviour, and architecture, but also business processes and data structure. The OMG has been an international, open membership, not-for-profit computer standards consortium since 1989. Its board of directors includes representation from almost all organizations that shape enterprise and internet computing today.

The UML language can be extended and used on any scale of project, from small software applications, right up to large and complex full system projects in avionics, maritime and other major applications. To help cover this vast range of possible applications, the OMG has now also produced a modelling language known as the System Modelling Language (or SysML), which extends the UML and provides added notation suitable for the systems engineer. Because the UML forms the foundation of SysML, the learning of UML will give the systems engineer most of the modelling notation he or she will ever require and will also help them on their way to learning and understanding SysML should they wished to.

Most engineering projects involve a large amount of software, so it makes sense to combine the use of both the UML and SysML on the same project. Some projects will use both languages, whereas some will be modelled using just the UML or sometimes, though more unusually, just the SysML. The two languages share many diagrams in common with each other, although sometimes minor modifications for their specific applications.



UML and SysML in Context

Designed and
Managed by the OMG

Overview

This course applies the UML to a large, real life avionics case study. The modeling will involve object orientation which will be introduced in easily assimilated modules throughout the course.

The understanding of the subject matter will be developed through a large amount of practical, hands on work which will be used to consolidate every topic covered.

We will be looking at different processes that can be used with the UML on systems engineering projects, but will focus on the V-model.

Objectives

By the end of the course attendees will:

- be familiar with the use of the UML language for describing the artefacts of complex systems
- be familiar with one of the major UML CASE Tools
- be comfortable with the concepts of system engineering
- have used a structured process as a framework within which to apply UML
- be able to understand all of the major UML diagrams to apply them to non trivial examples
- be able to handle requirements and link them to the functionality or equipment that satisfies these requirements
- be able to add connectivity throughout the model, especially with a view to tying together the structural and behavioural aspects of the model

Prerequisites

This course has no prerequisites. All subjects are covered from scratch.

Detailed Outline – Day 1

Process Driven Model-based Systems Engineering

- What is Systems Engineering?
- System Engineering Process
- Modelling Structure
- Modelling Behaviour

An Introduction to UML

- A Brief Look at UML
- An Overview of the UML diagrams
- Stereotyping in UML
- UML Tool Support

Organising the Model with Packages

- Organising the Model Structure
- Organising a Package Hierarchy

Capturing the Requirements

- Functional Requirements
- Other Requirements
- The Requirements Model
- Linking the Model to a Requirements Database
- Linking requirements Artefacts to other UML Artefacts

Capturing System Behaviour in Use Cases

- Use Cases as structured requirements
- Granularity of Use Cases
- Uncovering Use Cases from System Requirements

Capturing Detailed Behaviour – The Activity Diagram

- The Activity Diagram
- Activities, Sub-activities and Actions
- Linking the Activities
- Control Flow
- Object Flow
- Alternative Object Representation
- Initial, Final and Flow Final
- Forks and Joins
- Forks, Joins and Control Flow
- Decision Points and Merges
- Signals
- Objects and Signals
- Swimlanes
- Activity Partitions
- Interruptible Activity Region
- Pins
- Expansion Region
- Parameter Set

Detailed Outline – Day 2

The Structural View – The Deployment Diagram

- The application of the Deployment Diagram
- The notation of the Deployment Diagram

The Structural View – The Class Diagram

- The Class Diagram for Systems Engineering
- Drilling Down from systems design to the software design
- The Class Diagram for Software Engineering
- Classes and Blocks – stereotyping a Class
- Class Properties and Attributes
- Capturing Class/Block Behaviour
- Associations
- Multiplicities
- Using Generalisation to Create Hierarchies
- Using Generalisation for Software Inheritance
- Finding Classes

The Structural View – The Composite Structure Diagram

- Instantiating the Class into an Object
- Capturing Object Behaviour
- Modelling Interfaces with Ports and Flows

Detailed Outline – Day 3

Expanding the Use Cases

- Ranking Use Cases
- Specifying Use Cases
- Use Case Descriptions
- Non functional requirements
- Style Guidelines
- Use Case Storyboards
- Preconditions
- Postconditions
- Main Flow
- Extension Flow
- Graphical Form
-

Modelling Event Based Behaviour - The State Machine

- Capturing Business Rules
- Events and States
- Basic Notation
- Superstates and Substates
- Conditional Transitions
- Actions
- Finding Use Cases from the State Model

Modelling Message Based Behaviour - The Sequence Diagram

Modelling Cross Diagram Relationships

- The Purpose
- The Valid Relationships
- Requirements to Use Case Relationships
- Use Case to Test Case Relationships
- Relating the Behavioural Model to the Structural Model

A Resume of all UML 2.0 Diagrams

UML for Systems Engineering

