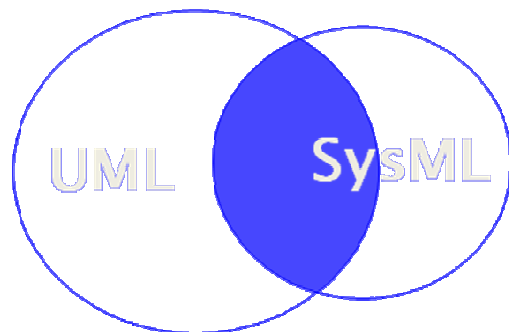


SysML in Context

The Unified Modelling Language™ - UML is the Object Management Group's (OMG) most-used specification. It is used widely to model application structure, behaviour, and architecture, as well as business processes and data structure. The OMG has been an international, open membership, not-for-profit computer standards consortium since 1989. The board of directors includes representation from almost all organizations that shape enterprise and internet computing today. The UML however falls short in providing the means to capture complex engineering systems. For this, the OMG has produced the System Modelling Language or SysML for short.

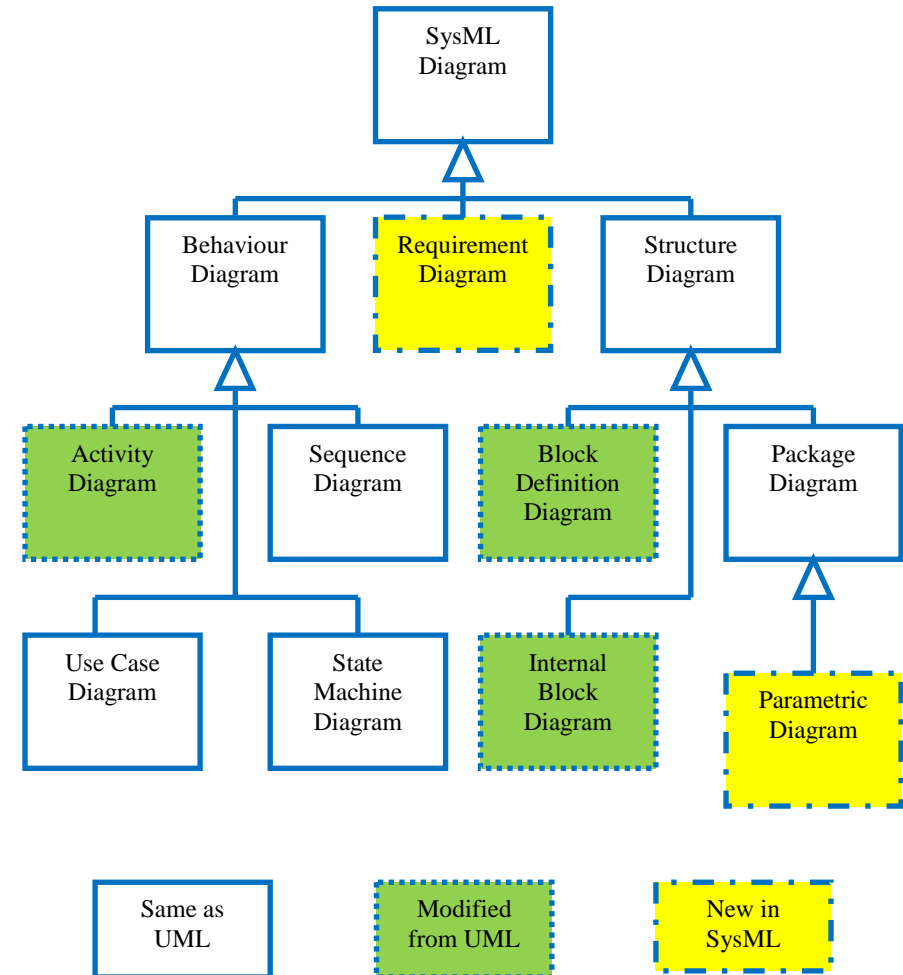
Most engineering projects also involve a large amount of software, so it makes sense to combine the use of both the UML and SysML on the same project. Some roles will use both languages, whereas others will model using just the UML or just the SysML.

SysML shares many diagrams with UML, although some of these diagrams have minor modifications for the purposes of systems engineering. SysML also adds some systems engineering orientated diagrams of its own.



SysML in Context

Designed and
Managed by the OMG



The SysML Diagrams

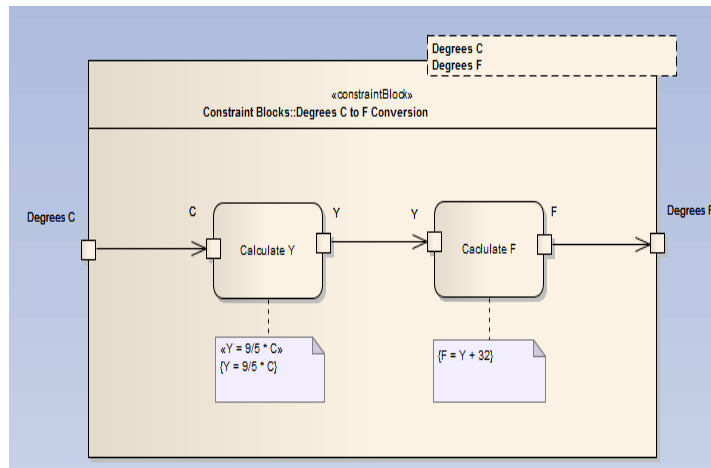
Overview

Duration: 4 days

This course applies the SysML notation to a large, real life case study. The process of tackling a single non-trivial example throughout the course ensures the attendees have an opportunity to discover when and where specific SysML diagrams are useful and to put this into the context of an overall systems engineering process. All organisations have their own processes, so the trainer will also focus (where appropriate) on how the SysML approach can be dovetailed into any existing processes.

An 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.

The course focuses on a single case study, but attendees are welcome and encouraged to replace this with a real life case study based on their own working domain or project experience.



Parametric Diagram

For capture and re-use of low level system constraints

Objectives

By the end of the course, attendees will:

- Be familiar with the use of the SysML language for describing the artefacts of complex engineering systems
- Be familiar with one of the major SysML CASE Tools
- Be comfortable with the concepts of systems engineering
- Have used a structured process as a framework within which to apply SysML
- Be able to understand all of the major SysML diagrams and apply them to non trivial examples
- Be able to provide an overall structure to the SysML Model
- 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 SysML model

Prerequisites

This course has no prerequisites.

Experience with systems engineering is not necessary, although it will help attendees fully appreciate the value and power of the SysML language.



Detailed Outline

Process Driven Model-based Systems Engineering

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

An Introduction to SysML

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

Organising the Model with Packages

- Organising the Model Structure
- Organising a Package Hierarchy
- The Package Diagram
- Capturing Package Relationships
- Packages as Namespaces
- Views and Viewpoints

Capturing the Requirements

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

Capturing Behaviour

- 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

Use Cases

- Use Cases as structured requirements
- Granularity of Use Cases
- Uncovering Use Cases from Business Processes
- Primary and Secondary Actors

Detailed Outline Continued

The Structural View

- The Block Definition Diagram
- Block Properties
- Modelling Interfaces with Ports and Flows
- Capturing Block Behaviour
- Using Generalisation to Create Hierarchies
- The Internal Block Diagram

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 Constraints with Parametrics

- System Constraints
- Using Constraint Expressions
- Re-useable Constraints
- Building Complex Constraint Blocks
- The Constraint Block Diagram
- Building the Parametric Diagram
- Constraining the Value Properties of a Block
- Capturing Values in Block Configurations
- Time-based Analysis
- Constraining Time-Dependent Properties
- Using Constraint Blocks to Constrain Item Flow

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

Customising SysML

A Resume of all SysML 1.1 Diagrams