

Systems Modeling Language

Level: intermediate

Length: 21 – 35 hours, depending on the practical part

Course objective: learn to use SysML in software development

What You Will Learn

- Learn the role of SysML, how it is used
- Introduces the diagram types in order to express a software construct or behavior

Who Can Attend: programmers who want to apply SysML during software development

Prerequisites: because SysML is an UML profile, it helps a prior knowledge of UML as well as the knowledge at least at a medium level of a particular object oriented programming language like C++, Java, C#, Python, etc.


Required Facilities: VGA projector, white board, workstation, development tools for writing programs in an object oriented language (Java, C++, C#, etc.)

Related Courses: UML, Design Patterns, Object Oriented Programming, object oriented programming languages (C++, Java, C#)

Description

The course offers a theoretical and practical approach of SysML towards its usage in constructing software constructs. The examples, case studies, hands on assignments offer a good understanding of SysML diagrams. There are performed the following activities:

- Presentation of the main graphical elements, diagram types, their semantics and how they are used
- Understanding, „reading” the SysML diagrams which were built by others
- Building of diagrams in order to express structures (static aspects) or behaviors (dynamic aspects)
- Implementation of SysML diagrams („translation”) in one object oriented language (C++, Java, C#) in order to emphasize variants and particularities related to that language
- Using of SysML for modeling during software development, at requirements specifications, object oriented analysis (OOA) & object oriented design (OOD)



The course is not based on any particular modeling tool or any editing tool of the SysML diagrams.

Contents:

1. Introduction to Model-Based Systems Engineering
2. Introduction to SysML
3. Class diagrams
4. Block definition diagrams
5. Internal block diagrams
6. Use case diagrams
7. Activity diagrams
8. Sequence diagrams
9. State machine diagrams
10. Package diagrams
11. Requirements diagrams
12. Allocations: cross-cutting relationship