Architectures in Simulink & Stateflow

How to Manage Large Software Models - 2 days

Models are the core artifacts in software development. Over time – within a single development project or even across multiple evolution steps – models grow as they capture more and more functionality. As a result, models become hard to maintain, barely understandable, and the risk of errors increases due to unexpected behavior. The technical debt of the model demands proper countermeasures. This training class addresses deficits caused by large Simulink & Stateflow models and shows ways to overcome their risks.

Target Audience

This training class is targeted at modelers, developers, testers, quality managers, project managers, and team leaders, whose focus is model-based development of embedded software based on MATLAB/Simulink for serial projects.

Highlights

- Basic concepts of software architectures
- Assessing architectural design principles in models
- Refactoring Simulink models
- Layered application architectures
- Representing architectures in models

Pawel Malysz, FCA US

"The seminars provide insight and ideas on how to approach handling large software projects in a systematic way with useful suggestions and quantitative metrics."

Languages

Available in English and German

Formats

On-Site
at one of our locations

Online
wherever you are

Icon Inhouse Training

Image not found or type unknown

For Your Company
online or in-house

More Details on Formats and Locations

Costs & Conditions

For costs and conditions see the PDF.

- Image not found or type unknown MES terms and conditions - Training classes (41.8 KiB)

Send Request

Our Trainers

Image not found or type unknown Image not found or type unknown Image not found or type unknown Image not found or type unknown Image not found or type unknown Image not found or type unknown

Agenda

Day 1

Overview: Model-based development and quality assurance with Simulink

- Foundations of model-based development
- Overview of development and quality assurance activities
- Characteristics of ISO 26262-compliant development

Analysis and evaluation of model structure

- Introduction to complexity metrics
- Calculating model complexity
- Countermeasures to overly complex models
- Assessing coherence in models
- Software architecture and model structure of the sample application

Software architecture

- Basics of software architectures

- Expected properties of an ISO 26262-compliant software architecture
- Principles of software unit design

Implementing software architectures in models

- Software architecture in models
- Principles for layered models
- Interface handling in models

Integrating models and distributed modeling

- Advantages of model referencing and libraries
- Defining distributed parameter files

Hands-on: Improving model structures

Day 2

Refactoring Simulink models and their structures

- Modeling styles facilitating refactoring
- Basic refactoring operations for Simulink
- Complex refactoring operations

Hands-on: Model refactoring

Refactoring Stateflow charts

- Challenges of Stateflow semantics
- A safe modeling style for Stateflow
- Sample refactoring rules

Hands-on: Refactoring participant models

Regression testing of models

- Test goals on different testing levels
- Safeguarding functional properties of model and code
- Regression testing and back-to-back testing, MiL – SiL – PiL
- Automatic test evaluation with test assessments

Process concerns regarding refactoring

- Roles and responsibilities of software architect, software developer, and test engineer
- Distinction between architecture design (top-down approach) and architecture improvement (bottom-up approach) of emerging architectures
- Refactoring in agile settings
- Refactoring legacy models