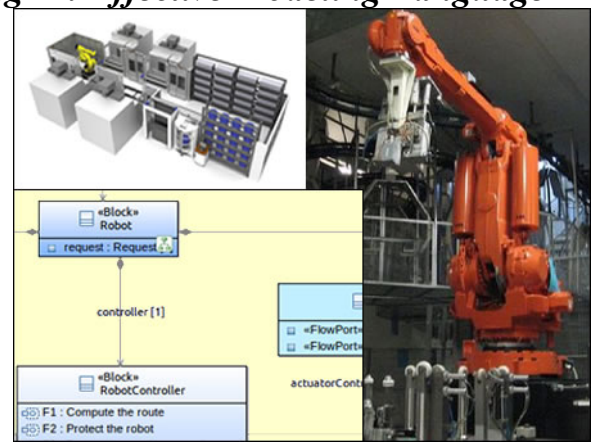
Honourcode, Inc.
1169 McCoury Lane, Spring Hill, TN 37174 – (615) 614-1109

July 2014 v2.0

System Architecting with SysML

A 3-Day Course

Model-Based Systems Engineering Using An Effective Modeling Language



The discipline of systems engineering (SE) is transforming, with much of the design information now captured in graphical models. System Modeling Language (SysML) is the primary tool used to create and retain this design information. Design information in SysML includes operational (stakeholder) definition, technical requirements, architectural analysis/structure, parametric definition, and test information, which together represent nearly the entirety of SE artifacts. An underlying database holds the SysML information so that data from one diagram appears synchronized on other diagrams. The benefits to the system architect are extensive.

This course shows how to architect and maintain a system definition using SysML. The course is filled with graphic examples from SysML models, but it is unlike other SysML courses in that the spotlight is on the system architecting. Students do not work on a computer during class, so that they can focus on the concepts rather than on use of a specific software tool. The course flows through familiar SE processes while teaching how the SysML models and structures support and enhance each task. We cover every SE activity and every SysML diagram, from Use Case and Activity diagrams to define operations; through State Machine, Sequence and Parametric diagrams to define system requirements; to Block Definition, Internal Block, and Requirements diagrams to define architectural structure. By the completion of this course, you will be able to apply SysML effectively in your own work.

In addition to our complete course materials, students also receive a copy of the seminal textbook *A Practical Guide to SysML* by Friedenthal, Moore, and Steiner.

***You should attend this course if you are*:**

- Designing or redesigning large systems and need better technical control.
- Transitioning from software engineering to systems engineering.
- Improving your systems engineering skill set.
- Working as part of a model-based systems engineering effort.

The course is aimed at

- Systems engineers,
- Design engineers,
- Technical team leaders,
- System support leaders, and
- Others who participate in defining and developing complex systems.

Design a drone aircraft

The Architecting Challenge takes the model-based design of a remotely piloted aircraft from concept through system design using SysML. Student groups work through the challenge in four parts to envision a drone aircraft useful for major event monitoring and control. In Part A, students define use cases and activities. In Part B, they define states, sequences, and parameters. In Part C, groups create alternative architectural concepts using block diagrams. Finally in Part D, they fill in the model with requirements allocation and model structure.

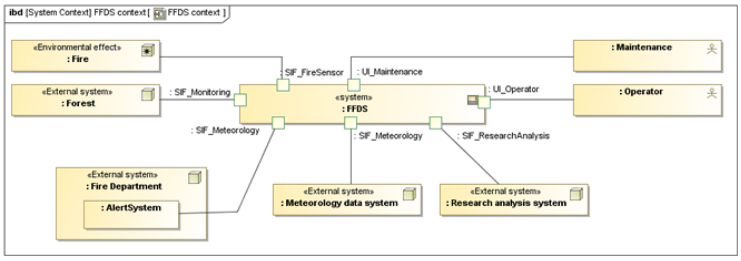
Topics Covered in the Course

Systems Architecting and Engineering (2:00) – How systems architecting and systems engineering fit together; how model-based systems engineering (MBSE) has developed and what benefits it offers

- A systems engineering model based on ISO-15288 and the INCOSE Handbook.
- What is an architecture?
- What is architecting?
- Six principles of MBSE
- Survey of current SysML tools

Basic SysML Concepts (1:00) – Where SysML came from; its purpose within the SE paradigms; the basic constructs of SysML.

- SysML underlying concepts; the information database; correct vs. complete
- The SysML language
- SysML and UML
- The nine SysML diagram types



- Common diagram structures: frames, headers, keywords, node symbols, path symbols, icons, notes

Operational Definition and Analysis (3:00) – Understanding stakeholder views of the problem and the system; stakeholder requirements; using SysML to analyze and document the operational architecture.

- The concept of a use case (scenario)
- System boundaries and external actors
- Use Case diagrams to define functionality
- Activity diagrams to elaborate the behavior of a use case
- Object flows to activities; fork, merge; streaming parameters, flow rates
- Control flows to handle logic
- Activity hierarchies

System Requirements Modeling (2:00) – Modeling technical requirements; the relationships between operations and requirements; how to document requirements and their relationships using SysML.

- Requirements, their forms and uses
- Requirements diagram to show relationships among requirements
- Types of requirement relationships and how to show them in SysML
- Requirements rationale in SysML
- Constraints as a part of requirements; the constraint block
- Parametric diagrams to define constraints
- Constraint blocks to modify flows
- Representing trade-offs
- Modeling requirements verification

System Logical Architecting and Analysis (3:00) – Requirements analysis using logical constructs; understanding the requirements better as a step toward physical system design; the logical architecture.

- Logical vs. physical architecture
- Functional design vs. object-oriented design; how SysML supports either
- The concept of a state; state transitions, triggers, guards, and effects
- State hierarchies and operation calls
- State Machine diagrams to analyze and document the event-based behaviors
- Sequence diagrams to analyze and document the message-passing behaviors
- Lifelines and interactions in sequences

System Physical Architecting (3:00) – System physical design; how to use SysML to show the physical architecture; the end-state of architecting.

- The block as a representation of systems, components, or flow items
- Block relationship types: association, composite, reference, generalization
- Block Definition diagrams to depict structural block relationships
- Internal Block diagrams to depict dynamic block relationships
- Quantifiable characteristics in a block
- Modeling interfaces using ports and flows
- Modeling block behavior
- Modeling classifications and variants
- Requirements diagrams to show hierarchical requirements allocations
- Requirements allocations in the block diagrams

Additional SysML Constructs (1:45) – Some remaining features of SysML for better architecting; organizing the model; allocating relationships

- Package diagrams to organize the model; types of organization; namespaces; imports and dependencies
- Requirements containment hierarchies
- Allocation between model constructs
- Alternate constructs in SysML
- Customizing SysML for projects or enterprises; SysML profiles; stereotypes

Architecting Challenge **Exercise** (5:00) – Student group work in four segments to practice the major aspects of architecting with SysML; creating the SysML model diagrams to define a system.

- Introduction to the remotely-piloted aircraft system
- Part A: Operational definition with use cases and activities
- Part B: Logical architecting with state machines, sequences, and parameters
- Part C: Physical architecting and alternatives with block diagrams
- Part D: Requirements allocation and package diagrams.

Summary (0:15) - Review of the important points of the course. Interactive discussion of participant experiences that add to the material.

Dr. Eric Honour, **CSEP, INCOSE Fellow**, and former INCOSE President, has been in international leadership of the engineering of systems for two decades, part of a 40+ year career of complex systems development and operation. His energetic and informative presentation style actively involves class participants. He was the founding Chair of the INCOSE Technical Board in 1994, and served as Director of the Systems Engineering Center of Excellence (SECOE). He is on the editorial board for *Systems Engineering*. He has been a successful entrepreneur, systems engineer, engineering manager, and program manager at Harris , E-Systems, and Link, and was a Navy pilot. He has led or contributed to the development of 17 major systems, including Air Combat Maneuvering Instrumentation, Battle Group Passive Horizon Extension System, and National Crime Information Center. BSSE (Systems Engineering) from US Naval Academy, MSEE from Naval Postgraduate School, PhD from University of South Australia based on his ground-breaking work to quantify the value of systems engineering.

Dr. Scott Workinger has led innovative technology development efforts in complex, risk-laden environments for 30 years in the fields of manufacturing (automotive, glass, optical fiber), engineering and construction (nuclear, pulp & paper), and information technology (expert systems, operations analysis, CAD, collaboration technology). He currently teaches courses on program management and engineering and consults on strategic management and technology issues. Scott has a B.S in Engineering Physics from Lehigh University, an M.S. in Systems Engineering from the University of Arizona, and a Ph.D. in Civil and Environment Engineering from Stanford University.