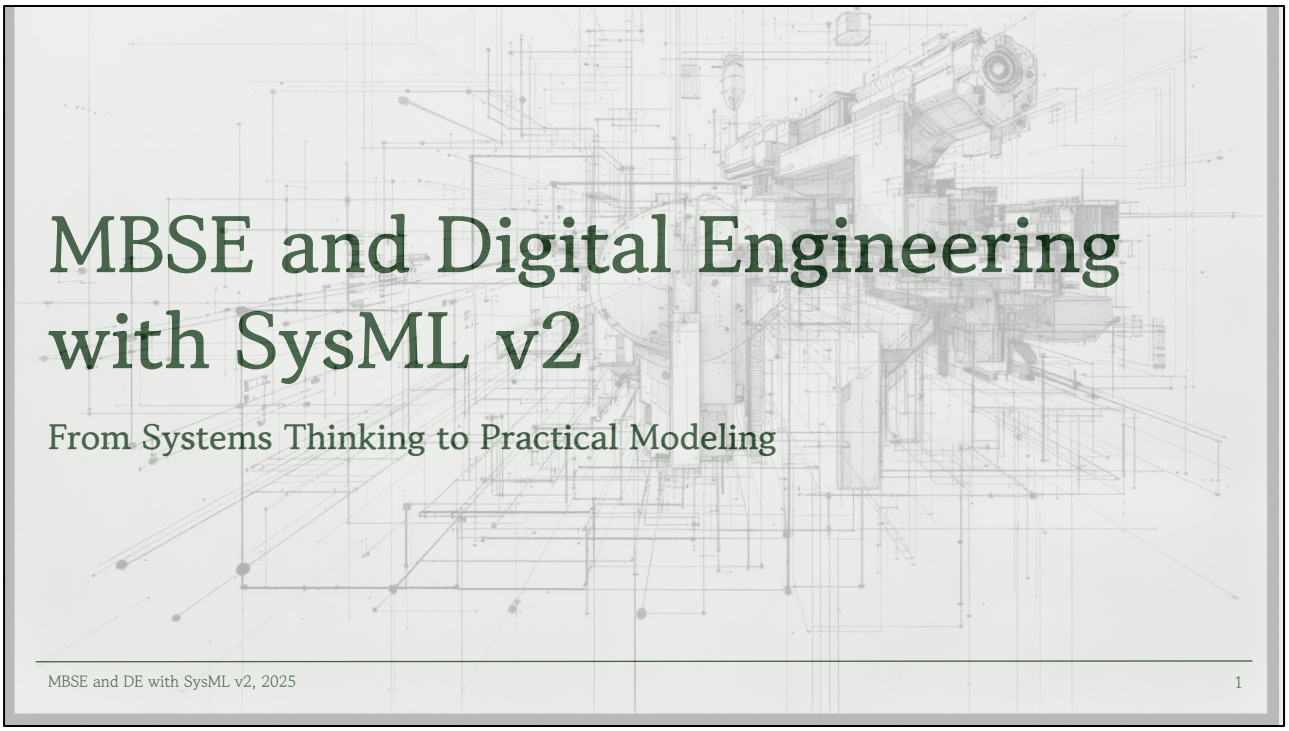


Welcome everyone to our course on **MBSE and Digital Engineering with SysML v2**.
Throughout this journey, we'll transform from basic systems thinking into practical, hands-on modeling.
You'll not only understand what Systems Engineering and MBSE are, but you'll **build a real system model** step-by-step, applying the new **SysML v2** language.
This course is designed to be very practical — combining theory, modeling exercises, and a long-term project to connect everything you learn.

About Me

20+ years of experience in MBSE, software engineering, software architecture, and project management with a proven track record for designing, developing, managing, and implementing various projects.
A Java and UML/SysML/BPMN/CATIA Magic tools expert, and a seasoned trainer with excellent communication, management, and technology skills.

Publications

- Automated Component-Selection of Design Synthesis for Physical Architecture with Model-Based System Engineering using Evolutionary" Trade-Off, The 28th Annual INCOSE International Symposium
- Encoding Technique of Genetic Algorithms for Block Definition Diagram using OMG SysML™ Notations, The 29th Annual INCOSE International Symposium
- Hatley-Pirbhai Control Flow Diagram with SysML for Early Validation, The 30th Annual INCOSE International Symposium
- A Pragmatic MBSE Approach of Nissan Powertrain Team to Minimizing Document-Based SEA Pragmatic MBSE Approach of Nissan Powertrain Team to Minimizing Document-Based SE, The 32nd Annual INCOSE International Symposium 2022
- Model-Based FMEA & FTA with Case-Based Reasoning, The 33rd Annual INCOSE International Symposium 2022
- Model-Based Product Line Engineering with Genetic Algorithms for Automated Component Selection, Complex Systems Design & Management, Apr 8, 2021

COURSE OUTLINE

Start

Week 01: Introduction to Digital Engineering, Systems Engineering, and MBSE Foundations

- Digital Engineering Vision
- Systems Engineering Fundamentals
- Traditional vs Model-Based Systems Engineering
- Motivation for MBSE
- Strategic Use of MBSE
- MBSE and Digital Engineering Integration
- Foundations for SysML v2
- Course Project Introduction

Objective: Introduce the vision of Digital Engineering and Model-Based Systems Engineering (MBSE), establish systems thinking principles, and explain the motivation for using SysML v2 as a modern modeling language — all while setting the stage for the course-long **drone project.**

Digital Engineering Vision
Introduce the modern approach to engineering systems digitally across their lifecycle using digital threads, digital twins, and authoritative models.
Systems Engineering Fundamentals
Establish basic Systems Engineering principles, lifecycle stages, and why systems thinking is essential.
Traditional vs Model-Based Systems Engineering
Explain the shift from document-based SE to model-driven SE and what problems MBSE solves.
Motivation for MBSE
Highlight inefficiencies and risks in traditional SE that MBSE addresses, showing benefits like traceability, analysis, and reduced errors.
Strategic Use of MBSE
When and where MBSE is most effective in real-world projects (not all systems need full MBSE).
MBSE and Digital Engineering Integration
Show how MBSE is a core enabler of Digital Engineering — especially connecting design, simulation, testing, and operation.
Foundations for SysML v2
Introduce why SysML v2 was created (problems in SysML v1) and what goals it aims to achieve (precision, integration, extendibility).

Course Project Introduction

Explain the course-long project: modeling the Autonomous Delivery Drone System step-by-step in SysML v2.

Week 02: Understanding the System Lifecycle and the MBSE Workflow

- Deep Dive: Systems Engineering Lifecycle
- How MBSE Supports Each Lifecycle Phase
- MBSE Workflow Overview
  - MBSE Practical Workflow Example (Applied to the Drone System)
- MBSE Tooling Overview
  - MBSE Tool Selection Decision (SysON selected instead of MBSE IDE)
- Hands-on Preparation
  - SysON Setup Instructions (Docker + Offline Package)
  - First SysON Hands-on Exercise (Sample Drone Model Tour)

Objective: Deepen understanding of the Systems Engineering Lifecycle and show how MBSE supports each phase through structured modeling workflows. Introduce the MBSE toolchain and prepare students to begin hands-on modeling in SysON.

Deep Dive: Systems Engineering Lifecycle
Explore in detail the five main phases (Concept, Development, Production, Utilization/Support, Retirement) and the deliverables at each phase.
How MBSE Supports Each Lifecycle Phase
Show how MBSE provides models and support at every phase (capturing needs, designing architecture, managing production, supporting operations, decommissioning).
MBSE Workflow Overview
Introduce the step-by-step MBSE process: from Needs modeling ➔ Requirements modeling ➔ Architecture ➔ Behavior ➔ V&V Planning ➔ Traceability.
MBSE Tooling Overview
Introduce available MBSE tools (SysON, MBSE IDE, CATIA Magic), why tools are needed, and select SysON for the course.
Hands-on Preparation
Install SysON, explore the tool, open and navigate a simple Drone System model to prepare for real modeling exercises starting Week 3.

Week 03: Introduction to SysML V2

- Key Differences in Philosophy vs SysML v1
- Core Modeling Concepts: Elements, Structure, Behavior, Requirements
- Modeling Domains and Metamodel Foundation
- SysML v2 Element Taxonomy Overview
- Tooling Environment Setup and Basic Navigation (SysON)
- Simple Practice: Create First Basic Model (Hands-On)

Objective: Establish the foundational concepts of SysML v2, including its unified modeling philosophy, element taxonomy, and domain structure. Guide students through their first hands-on modeling session in SysON to connect theory with practice.

Key Differences in Philosophy vs SysML v1
Explain the new conceptual mindset of SysML v2 compared to SysML v1 (focusing on modeling principles, not syntax details).
Core Modeling Concepts: Elements, Structure, Behavior, Requirements
Introduce foundational modeling ideas: everything in SysML v2 is an element; understand structure, behavior, and requirement domains.
Modeling Domains and Metamodel Foundation
Explain that SysML v2 unifies structure, behavior, requirements, and parametrics under one metamodel for consistency and traceability.
SysML v2 Element Taxonomy Overview
Introduce the organized hierarchy of modeling elements (e.g., Definitions, Usages, Actions, States) and explain how everything fits systematically.
Tooling Environment Setup and Basic Navigation
Practice using SysON: create a new project, navigate model explorer, switch between text and diagrams, validate models.
Simple Practice: Create First Basic Model
In-class hands-on exercise: create a Block (e.g., 'Drone'), add a Part (e.g., 'Sensor Suite'), create a simple Need and link it to a Requirement. This anchors the week's theory with a small real model.

Week 04: Modeling Stakeholder Needs and Concerns

- Identifying Stakeholders
- Organizing Stakeholders by Role and Influence
- Capturing Stakeholder Needs (using Need, Concern, Feature)
- Stakeholder Need Quality Tips
- Defining Use Cases from Needs
- Simple Practice: Create Stakeholder-Need-Concern Diagram
- Start Modeling Initial Needs for Course Project

Objective: Begin the Conceptual Layer by identifying stakeholders, capturing their needs and concerns, and translating them into early Use Cases. Students create their first stakeholder-centered model in SysON and begin the course project with real conceptual content.

Identifying Stakeholders
Systematically identify all individuals, groups, or organizations that affect or are affected by the system (Drone System example: Customer, Operator, Regulator, Maintenance Crew, etc.).
Organizing Stakeholders by Role and Influence
Teach students not just to list stakeholders, but also **categorize** them (e.g., Primary vs Secondary, High Influence vs Low Influence) — helps when prioritizing needs later.
Capturing Stakeholder Needs (using Need, Concern, Feature)
Model stakeholder needs using SysML v2 'Need', 'Concern', and optionally 'Feature' elements — linking needs to concerns clearly.
Stakeholder Need Quality Tips
Teach how to write good Needs: clear, single-focused, measurable if possible (without becoming a Requirement yet). Prevent vague Needs like "System shall be good."
Defining Use Cases from Needs
Translate needs into system interactions (Use Cases) — simple descriptions of "what the system must do" from the user's point of view.
Simple Practice: Create Stakeholder-Need-Concern Diagrams
Short hands-on exercise: Create a Stakeholder element, link to a Need, link Need to a Concern — using SysON. Just like a mini "first real model" for the Drone System.
Start Modeling Initial Needs for Course Project

Students start building their Autonomous Delivery Drone System model by creating real Needs, Concerns, and linking first Use Cases.

Week 05: Structure Modeling (Part I - Part Definitions and Parts)

- Modeling Structure with Part Definition, Part Usage, and Connections
- Defining System Hierarchy
- Composition and Decomposition
- Part Definition vs Part Usage Clarification
- Modeling Internal Connections and Connectors
- Simple Practice: Build a Basic System Architecture Diagram
- Start Creating Structure for the System Project

Objective: Model the System Context by defining the system boundary, external actors, and high-level conceptual structure using Part Definitions, Usages, and Connectors — without modeling internal architecture yet. Students apply these concepts by creating the System Context for their project in SysON.

Modeling Structure with Part Definition, Part Usage, and Connections
Use Part Definitions and Usages to represent the **system**, **external actors**, and their interactions in the System Context — without detailing internal structure yet.

Defining System Hierarchy
Model abstract conceptual components or subsystems using Part Usages inside the top-level system — focus on black-box **capabilities**, not internal implementation.

Composition and Decomposition
Break down complex systems into abstract, high-level parts or capabilities — modeled as black-box Part Usages in a system definition.

Part Definition vs Part Usage Clarification
Clarify that Part Definitions define reusable types, while Part Usages are instances used to represent structure or context in the system.

Modeling Internal Connections and Connectors
Use Connectors to model **conceptual interactions** between the system and external entities — without defining Ports or internal flows yet.

Simple Practice: Build a Basic System Context Diagram
Hands-on: model a neutral system (e.g., Vehicle System) with external actors and connectors in SysON — no internal structure yet.

Start Creating Structure for the System Project

Students build the **System Context** for their Drone System — define the system, add external entities, and connect them using conceptual links.

Week 06: Behavioral Modeling (Part I - Scenarios and Interaction)

- Use Case, Interaction, and Sequence Modeling
- Defining Scenario, Event, and Interaction
- Time and Causality in SysML v2
- Simple Practice: Create a Basic Interaction Model
- Model Behavioral Interactions for the Student Project

Objective: Model how the system is used and what it must respond to — from the outside, via scenarios, actors, and message flows.

Use Case, Interaction, and Sequence Modeling
Model how actors use the system by defining high-level goals (Use Cases) and interaction flows as message sequences.
Defining Scenario, Event, and Interaction
Describe usage Scenarios as sequences of Events and Interactions between actors and the system — without modeling logic yet.
Time and Causality in SysML v2
Introduce the concept of **event order and causality** in interaction models — showing what happens, when, and in what sequence.
Simple Practice: Create a Basic Interaction Model
Hands-on: use SysON to model a simple usage sequence (e.g., Drone Operator starts mission, Drone responds) — identify Events and Interactions.
Model Behavioral Interactions for the Student Project
Students model usage scenarios for their Drone System as Use Cases and Sequence Models — keep focus on actor ↔ system communication.

Week 07: Modeling Requirements

- How to Define and Refine Requirements in SysML v2
- Functional vs Non-Functional Requirements
- Linking Needs to Requirements
- Requirement Quality Tips
- Requirement Grouping and Organization
- Simple Practice: Create Requirement Model and Traceability Links
- Apply to Student Project

Objective: Transform stakeholder-driven conceptual insights into clear, testable, structured system requirements using SysML v2.

How to Define and Refine Requirements in SysML v2
Teach how to create formal system Requirements from Needs — using SysML v2 requirement elements with clear structure and rationale.

Functional vs Non-Functional Requirements
Explain the two types of requirements: Functional = what the system does, Non-Functional = how well it must do it (performance, safety, etc.)

Linking Needs to Requirements
Model traceability from Needs to Requirements using SysML v2 links — showing how stakeholder concerns translate into system constraints.

Requirement Quality Tips (Good vs Bad Requirements)
Teach how to write **clear, testable, and unambiguous** Requirements. Avoid vague words like "fast" or "user-friendly" without quantification.

Requirement Grouping and Organization
Show how to organize requirements into categories (e.g., Safety, Performance, Interface) for better readability and traceability.

Simple Practice: Create Requirement Model and Traceability Links
Hands-on: use SysON to define Requirements and link them back to Needs and Use Cases. View traceability graphs in the model.

Apply to Student Project
Students refine their Drone System Needs into formal, categorized Requirements inside SysON — starting the Logical Layer with structured traceability.

Week 08: Structure Modeling (Path II - Interfaces and Ports)

- Interface Definitions
- Port Types and Connections
- Definition vs Usage of Interfaces Clarification
- Modeling Plug-and-Play Architectures
- Modeling Interface Contracts
- Simple Practice: Build Interface and Connect Parts in SysON
- Apply Interface Modeling to Student Project

Objective: Define and use formal Interfaces and Port types to support modular, traceable system interaction modeling.

Interface Definitions
Define reusable interface def elements to specify types of interaction (data, power, mechanical, etc.) between system parts.
Port Types and Connections
Teach the kinds of Ports (port def, port) and how they enable interface realization between Part Usages.
Definition vs Usage of Interfaces Clarification
Clarify that Interfaces are defined separately and then **used via Ports** — reinforcing the Definition/Usage modeling pattern.
Modeling Plug-and-Play Architectures
Show how modularity and flexibility arise from well-structured Interfaces — enabling replaceable or upgradable components.
Modeling Interface Contracts
Use Interface Definitions to specify expectations such as **data structure**, **protocol**, or **behavioral rules** for interaction — supporting validation and integration.
Simple Practice: Build Interface and Connect Parts in SysON
Hands-on: define an Interface, assign Ports to Parts, connect them using Ports, and verify the connection graph in SysON.
Apply Interface Modeling to Student Project
Students define Interfaces (e.g., TelemetryLink, PowerSupply) and connect components in their Drone System using Ports and Interface Definitions.

Week 09: Behavioral Modeling (Part II - State Machines and Activities)

- Modeling Behavior with Action, State, Transition, Activity
- State Machine Modeling Best Practices
- Activity Modeling: Control Flow vs Object Flow
- Combining Behavior and Structure
- Simple Practice: Build a State Machine and an Activity Model
- Apply to System Functions

Objective: Model how the system behaves over time using states, transitions, and structured activities — linked to the system structure.

Modeling Behavior with Action, State, Transition, Activity
Introduce the core behavioral modeling tools: state, transition, action, activity. Used to describe system response and task flows.
State Machine Modeling Best Practices
Teach how to build clear and readable state machines — with defined triggers, transitions, and guard conditions.
Activity Modeling: Control Flow vs Object Flow
Distinguish between control flows (task ordering) and object flows (data passed between actions).
Combining Behavior and Structure
Show how to allocate state machines or activity behavior to specific system parts (e.g., controller, sensor).
Simple Practice: Build a State Machine and an Activity Model
Hands-on: model a basic operational Flight State Machine and a simple Delivery Activity sequence in SysON.
Apply to System Functions
Students implement behavior for key Drone functions using State and Activity models — mapped to their existing structure.

Week 10: Parametric and Constraint Modeling

- Mathematical Relationships and Constraints
- Using Constraint, Parameter, Value
- Parametric Diagram Overview
- Constraint Propagation Concept
- Trade Studies and Analysis Support
- Simple Practice: Build a Basic Constraint Model
- Apply Constraints to Performance Needs in Student Projects

Objective: Define and apply engineering constraints to support analysis, trade studies, and performance modeling in SysML v2.

Mathematical Relationships and Constraints
Model engineering formulas and rules directly in the system using SysML v2 constraint def, constraint, and related elements.

Using Constraint, Parameter, Value
Introduce the building blocks: parameter (variable inputs), value (system data), constraint (rules that relate them)

Parametric Diagram Overview
Use parametric diagrams to show how constraints are structurally bound to system parts — not floating math.

Constraint Propagation Concept
Teach how SysML v2 enables automatic calculation when values change — ideal for performance impact and sensitivity analysis.

Trade Studies and Analysis Support
Explain how constraints support **evaluating design alternatives** — e.g., different battery sizes or payload weights.

Simple Practice: Build a Basic Constraint Model
Hands-on: define one constraint set (e.g., flight time vs payload) and connect it to system parts in SysON.

Apply Constraints to Performance Needs in Student Projects
Students model performance-related constraints (e.g., endurance, battery capacity) in their Drone System and trace them to functional or non-functional requirements.

Week 11: Allocation and Mapping

- Behavior-to-Structure Mapping
- Requirement Allocation and Traceability
- SysML v2 Relationships: refine, satisfy, verify
- Traceability View and Navigation in SysON
- Simple Practice: Create a Traceability Chain
- Apply Allocation to Current System Architecture

Objective: Bridge behavior, structure, and requirements by teaching allocation and traceability in SysML v2. Students learn to build complete traceability chains and apply formal relationships to connect stakeholder intent to architecture and system logic.

Behavior-to-Structure Mapping
Teach how to allocate system behaviors (like activities or interactions) to structural elements (Part Usages) — e.g., "Navigation logic is executed by the Flight Controller."
Requirement Allocation and Traceability
Show how Requirements are allocated to system elements and behaviors — and how to trace from Need ➔ Requirement ➔ Architecture ➔ V&V.
SysML v2 Relationships: refine, satisfy, verify
Introduce the formal relationships used in SysML v2 to connect Needs to Requirements, Requirements to Parts, and Requirements to Tests.
Traceability View and Navigation in SysON
Practice using SysON's model navigation to trace links between Needs, Requirements, Functions, and Structures.
Simple Practice: Create a Traceability Chain
Hands-on: build a full trace from a single Stakeholder Need to its Requirement, allocated Behavior, and linked Part Usage.
Apply Allocation to Current System Architecture
Students apply mappings inside their Autonomous Drone System model (e.g., allocate "Obstacle Avoidance" behavior to "Sensor Suite").

Week 12: Verification and Validation Modeling

- Verification Elements in SysML v2
- Modeling V&V Plans
- Linking Tests to Requirements and Functions
- Types of Verification Methods
- Simple Practice: Build a Test Case and Link it
- Apply V&V Modeling to the Project

Objective: Introduce model-based verification by teaching how to define Test Cases, Verification Criteria, and verify relationships in SysML v2. Students build traceable V&V models that link requirements to testable system behaviors and structure.

Verification Elements in SysML v2
Introduce testcase, satisfy, verify, and criteria elements — used to represent how each requirement will be tested or validated.
Modeling V&V Plans
Teach how to define a test plan inside the model — connecting Test Cases to Requirements and System Elements.
Linking Tests to Requirements and Functions
Show how tests are linked using verify relationships and how to connect them to the correct system behaviors and parts.
Types of Verification Methods
Explain common methods: Analysis, Inspection, Demonstration, and Test — and how to represent them inside a model.
Simple Practice: Build a Test Case and Link it
Hands-on: students define a test case and link it to one requirement, one behavior, and one structural part using verify relationships.
Apply V&V Modeling to the Project
Students create a small V&V model for their Drone System (e.g., "Verify Max Payload", "Demonstrate Auto-Return Behavior").

Week 13: Variability and Configurability

- Modeling Product Lines and Variants
- Introduction to Variation, Choice, Variant Elements
- Capturing Configurable Structures and Behaviors
- Simple Practice: Create a Variant Structure Model
- Apply Variant Thinking to the Project

Objective: Introduce product line modeling and system variability using variation and choice elements in SysML v2. Students learn to build configurable system models that support multiple structural and behavioral configurations for different drone missions.

Modeling Product Lines and Variants
Introduce how to model a family of related systems (e.g., multiple drone types) using variation points and variant selections.
Introduction to Variation, Choice, Variant Elements
Teach the core SysML v2 elements used for variability: variation, choice, variant, and how they work together to represent configurable models.
Capturing Configurable Structures and Behaviors
Show how to apply variability not only in structure (different payloads or sensors) but also in behavior (different mission flows).
Simple Practice: Create a Variant Structure Model
Hands-on: define a variation point (e.g., camera type), create variant options, and select a configuration using choice.
Apply Variant Thinking to the Project
Students update their Drone System model to support multiple configurations — e.g., a Surveillance Drone vs. Delivery Drone.

Week 14: Advanced Topics and Best Practices

- Model Organization and Reuse
- Namespaces, Packages, and Libraries
- Modular Modeling and Views
- Introduction to Profiles (Optional)
- Simple Practice: Organize and Import Elements
- Apply Model Organization to Project

Objective: Teach how to organize and modularize large system models using packages, namespaces, and libraries in SysML v2. Students apply reuse and structuring techniques to improve clarity, scalability, and stakeholder-specific views in their project models.

Model Organization and Reuse
Teach how to structure large system models using packages, libraries, and reuse strategies — to avoid duplication and improve maintainability.

Namespaces, Packages, and Libraries
Introduce SysML v2 concepts for modular organization: package, import, namespace, and how to use them for clear model structuring.

Modular Modeling and Views
Explain how to organize a system into modular components and present specific views (e.g., structure view, behavior view) for different stakeholders.

Introduction to Profiles (Optional)
Briefly introduce the concept of profiles and stereotypes for domain-specific extensions — only as an optional preview for advanced learners.

Simple Practice: Organize and Import Elements
Hands-on: students use import statements and packages in SysON to split their model cleanly into subsystems.

Apply Model Organization to Project
Students repackage and refactor their Drone System model for clarity and reuse (e.g., separate requirements, behavior, and structure into packages).

Week 15: Final Project Presentations and Course Wrap-Up

- Student Presentations of Final System Models
- Model Walkthrough and Evaluation
- Reflection on MBSE and SysML v2 Learning Journey
- Discussion: SysML v2 in the Real World
- Next Steps: Certification, Research, Career

Objective: Conclude the course with student presentations of their complete SysML v2 models, peer/instructor evaluation, and reflection on the MBSE learning journey. Highlight real-world SysML v2 applications and guide students toward next steps in certification, research, and careers.

Student Presentations of Final System Models

Each student (or team) presents their completed SysML v2 model of the Autonomous Drone System — including Needs, Requirements, Structure, Behavior, Constraints, Allocation, and V&V elements.

Model Walkthrough and Evaluation

Instructor and peers provide feedback based on model completeness, clarity, correctness, traceability, and applied best practices.

Reflection on MBSE and SysML v2 Learning Journey

Students reflect on how their understanding evolved from Week 1 to Week 15, and discuss real challenges, insights, and breakthroughs.

Discussion: SysML v2 in the Real World

Overview of how SysML v2 is being adopted in aerospace, defense, automotive, and medical domains — including toolchains, APIs, and digital thread integration.

Next Steps: Certification, Research, Career

Guide on continuing the journey — SysML/MBSE certifications (e.g., OMG, INCOSE), further research areas, and MBSE roles in industry.

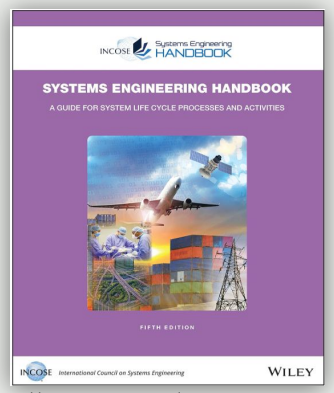
Recommend Reading

SE Book Of Knowledge

https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)

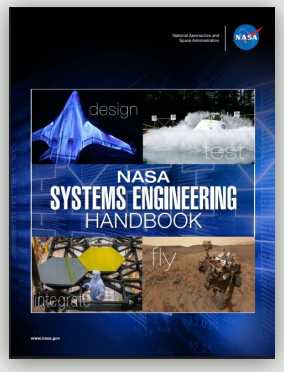
SysML 2.0 Document:
https://github.com/Systems-Modeling/SysML-v2-Release/tree/master/doc

INCOSE Handbook



https://www.amazon.com/INCOSE-Systems-Engineering-Handbook/dp/1119814294

NASA SE Handbook



https://www.nasa.gov/wp-content/uploads/2018/09/nasa_systems_engineering_handbook_0.pdf

QUESTION!