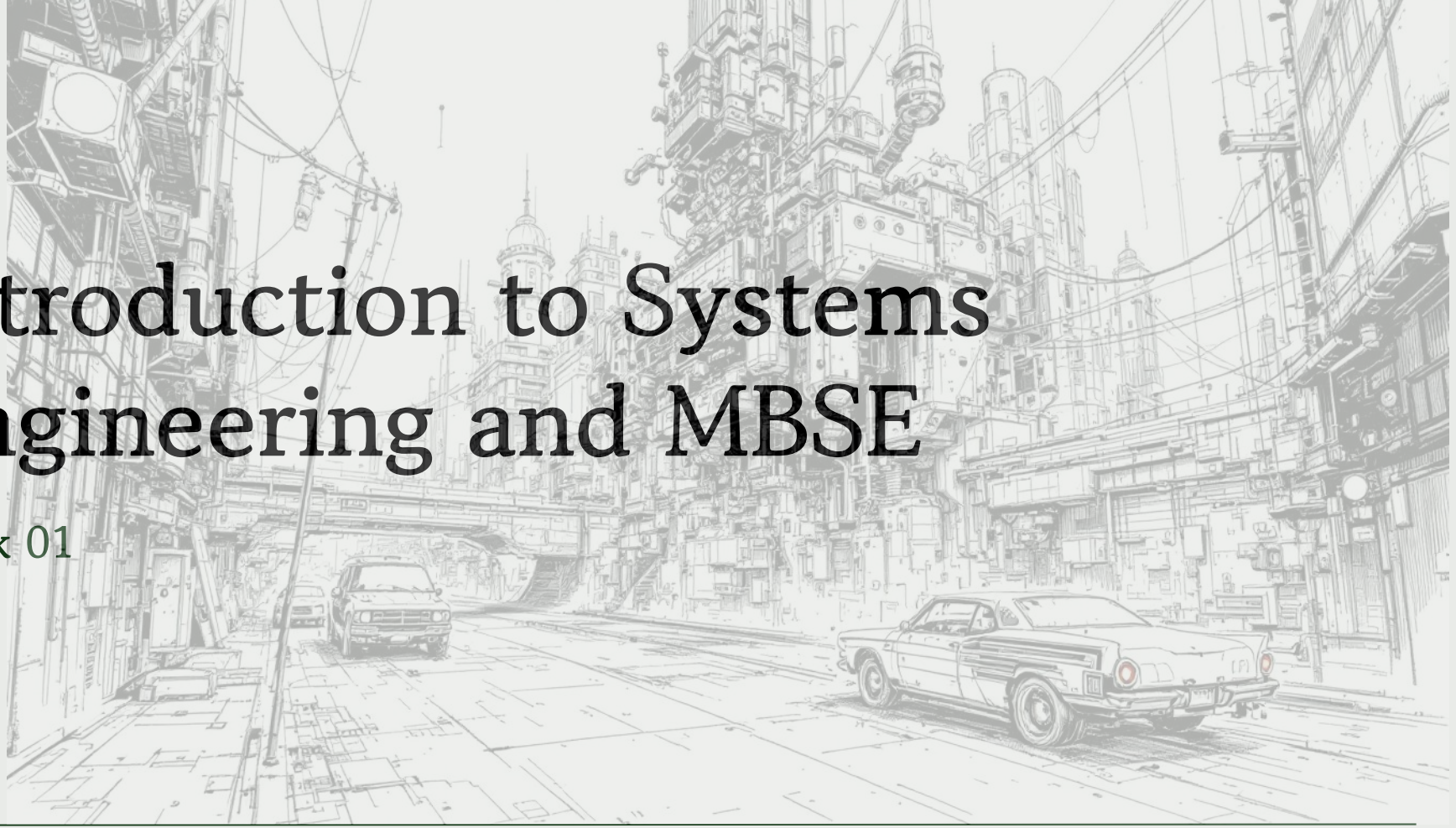


Introduction to Systems Engineering and MBSE

Week 01

Introduction to Digital Engineering

Digital Engineering is the future of systems development — using models and digital data across the entire system lifecycle to improve quality, speed, and collaboration.

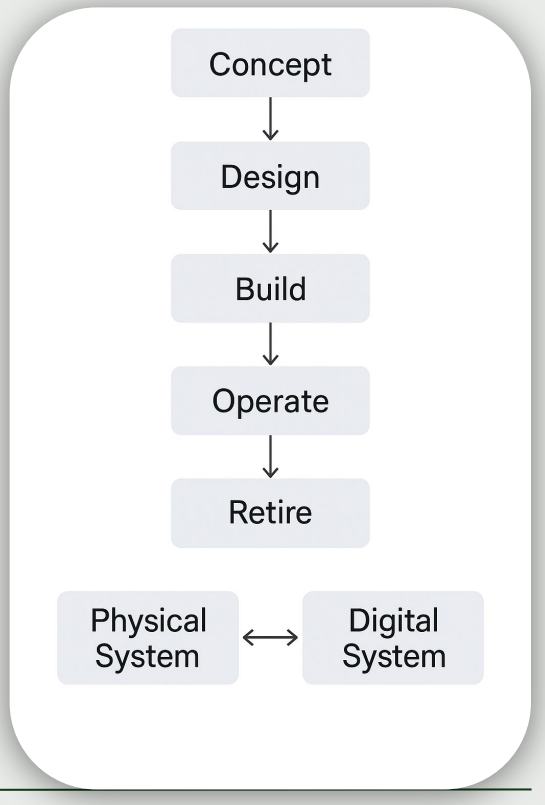
- Moves from document-based to **data-driven and model-driven** processes.
- Ensures **traceability** and **continuity** from concept to disposal.
- Enables **Digital Thread** and **Digital Twin** capabilities.
- Reduces cost, risk, and time-to-market for complex systems.

Digital Thread and Digital Twin

Digital Thread and Digital Twin are key concepts that enable the realization of Digital Engineering across the system lifecycle.

Digital Thread :- A connected, traceable flow of information about a system across its entire lifecycle — from requirements to design to operation and beyond.

Digital Twin :- A dynamic, real-time digital representation of a physical system that mirrors its behavior, performance, and evolution over time.



What is Systems Engineering?

Systems Engineering is an interdisciplinary approach focused on enabling the realization, deployment, and operation of successful systems.

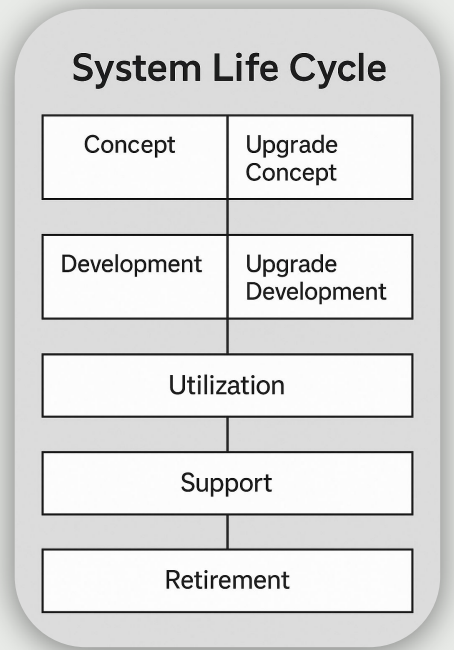
- Focuses on whole system success across its entire lifecycle.
- Integrates people, processes, and technology.
- Balances cost, schedule, performance, and risk.
- Starts from stakeholder needs and ends with validated solutions.

“THE RIGHT SYSTEM IS BUILT, IT IS BUILT RIGHT”

Systems Engineering Lifecycle

The Systems Engineering Lifecycle includes multiple overlapping stages that govern the realization, use, and retirement of a system.

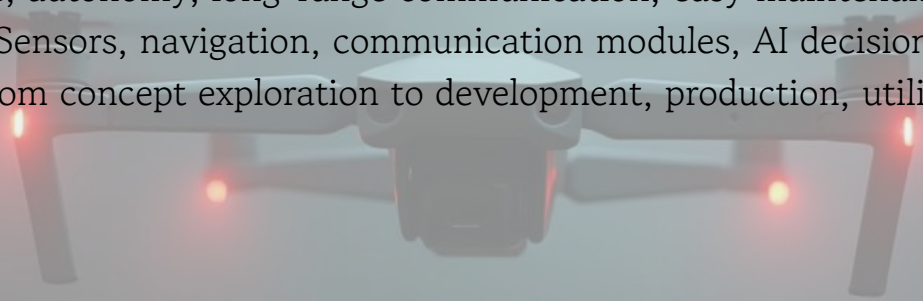
- **Concept**: Define stakeholder needs and explore possible solutions.
- **Development**: Develop and refine system designs and architectures.
- **Production**: Manufacture, assemble, and deliver the system.
- **Utilization**: Operate the system to deliver intended services.
- **Support**: Sustain and maintain system performance and availability.
- **Retirement**: Safely remove the system from service at end of life.



Systems Engineering in the Real World

Example: Designing an Autonomous Drone System

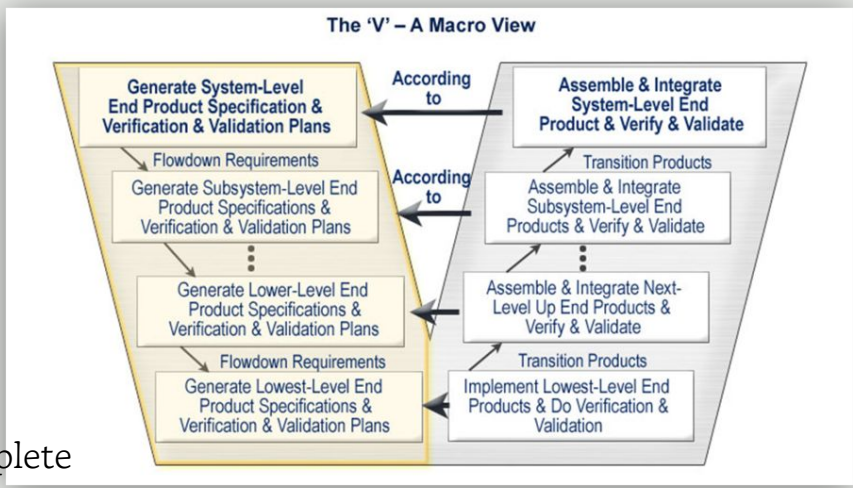
- **Stakeholders**: Military users, pilots, maintenance crews, command centers.
- **Needs**: Surveillance, autonomy, long-range communication, easy maintenance.
- **System Elements**: Sensors, navigation, communication modules, AI decision logic, power systems.
- **Lifecycle Focus**: From concept exploration to development, production, utilization, support, and retirement.



Systems Engineering and the V-Model

The V-Model represents the Systems Engineering process, showing how system definition and system realization activities are connected through verification and validation.

- **Left side**: Define stakeholder needs, requirements, architecture, and detailed design.
- **Bottom**: Build, integrate, and implement system elements.
- **Right side**: Verify components and validate the complete system.
- **Verification** ensures "Did we build the system right?"
- **Validation** ensures "Did we build the right system?

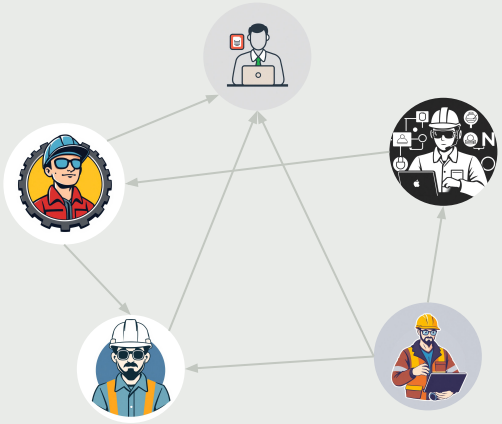


The Vee Activity Diagram (Prosnik 2010). Released by the Defense Acquisition University (DAU)/U.S. Department of Defense (DoD).

Traditional Systems Engineering

Traditional Systems Engineering relies heavily on document-based processes to capture and manage system information.

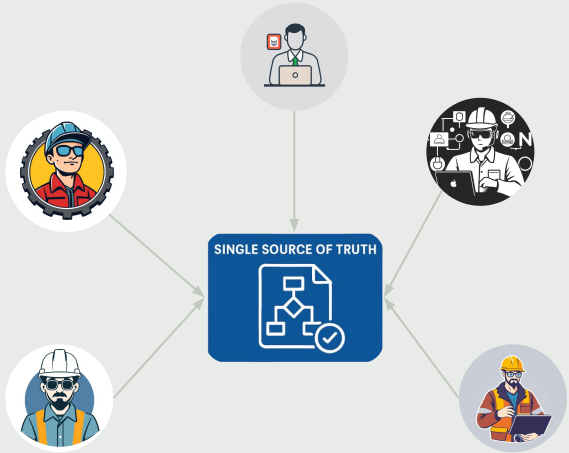
- System information stored in **separate documents** (requirements, design specs, test plans).
- **Manual updates** cause inconsistencies and delays.
- Difficult to maintain **traceability** between lifecycle stages.
- **Collaboration barriers** across teams and suppliers.
- Risk of **error propagation** and **late problem discovery**.



Model-Based Systems Engineering (MBSE)

Model-Based Systems Engineering (MBSE) replaces document-centric approaches with integrated, digital models of the system across its lifecycle.

- System information captured in **centralized, dynamic models**.
- Models represent **requirements, structure, behavior, constraints, and verification** together.
- **Traceability** and **consistency** are built-in.
- **Collaboration** improves through a shared system model.
- Enables **early simulation**, **analysis**, and **automation**.



Traditional vs MBSE

Comparing Traditional and Model-Based Systems Engineering

Aspect	Traditional SE	Model-Based SE (MBSE)
Information Storage	Separate documents	Centralized digital models
Updates	Manual and error-prone	Automatic and consistent
Traceability	Difficult and fragmented	Built-in, automatic
Collaboration	Siloed, disconnected teams	Shared system model
Verification	Late, after building	Early, throughout lifecycle
Complexity Handling	Limited	Scalable to highly complex systems

Benefits and Motivation for MBSE

MBSE delivers major benefits by improving system quality, reducing risks, and enabling faster, smarter development.

- **Improved Quality**: Early detection of errors through model analysis and simulation.
- **Reduced Risk**: Consistent traceability and validation throughout lifecycle.
- **Faster Development**: Automation reduces manual rework and accelerates design cycles.
- **Enhanced Collaboration**: Shared, integrated models improve team communication.
- **Better Complexity Management**: Scales for large, multidisciplinary systems.
- **Supports Digital Engineering**: Anchors the Digital Thread and Digital Twin concepts.

Case Study – NASA's Artemis Program

NASA uses MBSE to manage the complexity of the Artemis Program, aiming to return humans to the Moon and enable future Mars exploration.

- **Challenge**: Coordinating thousands of components, international teams, and contractors.
- **Solution**: Integrated system models for spacecraft, launch vehicles, ground systems, and mission planning.
- **MBSE Benefits**:
  - Early detection of design inconsistencies.
  - Improved collaboration across diverse teams.
  - Accelerated decision-making.
  - Reduced cost and program risk.
- **Result**: Increased confidence in mission success with sustainable system evolution.

https://www.space.com/artemis-program.html
https://ntrs.nasa.gov/api/citations/20240005284/downloads/NASAs_Use_of_MBSE_and_SysML_Modeling.pdf

What If MBSE Was Not Used?

Without MBSE, large, complex projects face overwhelming risks, delays, and failures.

- **Fragmented information**: Inconsistent documents and missing links.
- **Late discovery of errors**: Problems found only during integration and testing.
- **Poor collaboration**: Misaligned teams and duplicated efforts.
- **Increased project delays**: Rework and missed dependencies.
- **Escalating costs**: Late fixes are 10–100× more expensive.
- **Higher risk of mission failure**: Incomplete verification and validation.

When MBSE is a Good Fit

MBSE is ideal for complex, high-risk, collaborative, and evolving system developments.

- **Complex Systems**: Many interacting parts and disciplines (e.g., spacecraft, autonomous vehicles, smart grids).
- **High-Risk Projects**: Mission-critical, safety-critical, or high-cost systems (e.g., defense, aerospace, healthcare).
- **Multi-Team Collaboration**: Distributed, international, or cross-domain projects.
- **Long Lifecycle Systems**: Systems operated over decades with multiple upgrades.
- **Systems Requiring Traceability**: Regulatory compliance or formal certification needed.

When MBSE Might Not Be Necessary

For small, simple, short-lifecycle projects, MBSE may not deliver enough return on investment.

- **Simple Systems**: Few components, low complexity, clear interactions.
- **Short-Lifespan Projects**: Temporary or experimental systems.
- **Small, Co-Located Teams**: Fast communication, low risk of misalignment.
- **Rapid Prototyping and Exploration**: Early-stage innovation needing speed over structure.
- **Low-Risk Tolerance**: Where failure cost is low and acceptable.

Role of MBSE in Digital Engineering

MBSE is a foundational enabler of Digital Engineering, providing the structured system models needed for data continuity, automation, and lifecycle integration.

- **Anchors the Digital Thread**: Maintains traceability across all lifecycle stages.
- **Supports the Digital Twin**: Provides accurate, validated models for simulation and monitoring.
- **Enables Automation**: Simulation, verification, and validation of system behaviors early and continuously.
- **Enhances Collaboration**: Cross-discipline teams work from a common, evolving system model.
- **Facilitates Agile, Scalable Engineering**: Adapts to changes rapidly without losing control.

The Three Pillars of MBSE

01

MODELING LANGUAGE

Defines what can be modeled (e.g., SysML v2).

02

TOOLS

Provide environments for creating, managing, analyzing models (e.g., SysML V2 Pilot, CATIA Magic, SysOn).

03

METHODOLOGY

Guides how modeling is applied systematically to support lifecycle processes (e.g., V-Model, Agile MBSE).

What is SysML v2 in MBSE?

SysML v2 is the next-generation modeling language designed to fully support Model-Based Systems Engineering (MBSE) with improved precision, expressiveness, and automation.

- Developed by **Object Management Group (OMG)**.
- Captures **structure**, **behavior**, **requirements**, **constraints**, and **verification** in a unified model.
- Enables **machine-readable, interoperable, analyzable models**.
- Designed for **modern complex systems** across industries.
- Foundation for **Digital Engineering** practices.

https://www.omg.org/spec/SysML/2.0/Beta2/About-SysML
https://github.com/Systems-Modeling/SysML-v2-Release
https://github.com/Systems-Modeling/SysML-v2-Pilot-Implementation

Why SysML v2 Was Developed

SysML v2 was created to overcome the limitations of SysML v1 and meet the needs of modern, complex, and digital systems engineering.

- **Clarify Semantics**: Reduce ambiguity and inconsistency in models.
- **Enable Automation**: Support machine-readable, executable models.
- **Improve Expressiveness**: Better model structure, behavior, constraints, and parametrics.
- **Enhance Usability**: More intuitive, flexible modeling structures for engineers.
- **Support Interoperability**: Easier exchange and integration across tools and domains.

Introduction to the Course-Long Project

Throughout this course, you will develop a complete system model using SysML v2 — applying MBSE principles step-by-step.

- **Project Theme**: Design an **Autonomous Drone System** for surveillance and data collection.
- **Incremental Building**:
  - Capture **stakeholder needs** and **requirements**.
  - Model **structure**, **behavior**, **constraints**, and **verification**.
  - Apply MBSE practices across the full **system lifecycle**.
- **Final Goal**: Deliver a **complete SysML v2 system model** that could support simulation, verification, and traceability. (with future tool supported)

Summary of Week 1

This week, we built the foundation for understanding Digital Engineering, Systems Engineering, and MBSE.

- 💯 Understood the **vision of Digital Engineering** (Digital Thread + Digital Twin).
- 💯 Learned the **purpose and process of Systems Engineering** across the lifecycle.
- 💯 Identified **limitations of traditional approaches**.
- 💯 Explored how **MBSE transforms systems engineering**.
- 💯 Recognized **where MBSE is critical** and **where it may not be necessary**.
- 💯 Positioned **MBSE as a key enabler of Digital Engineering**.
- 💯 Introduced **SysML v2** as the modern modeling language for MBSE.
- 💯 Launched the **Course-Long Project**: Autonomous Drone System.

QUESTION!