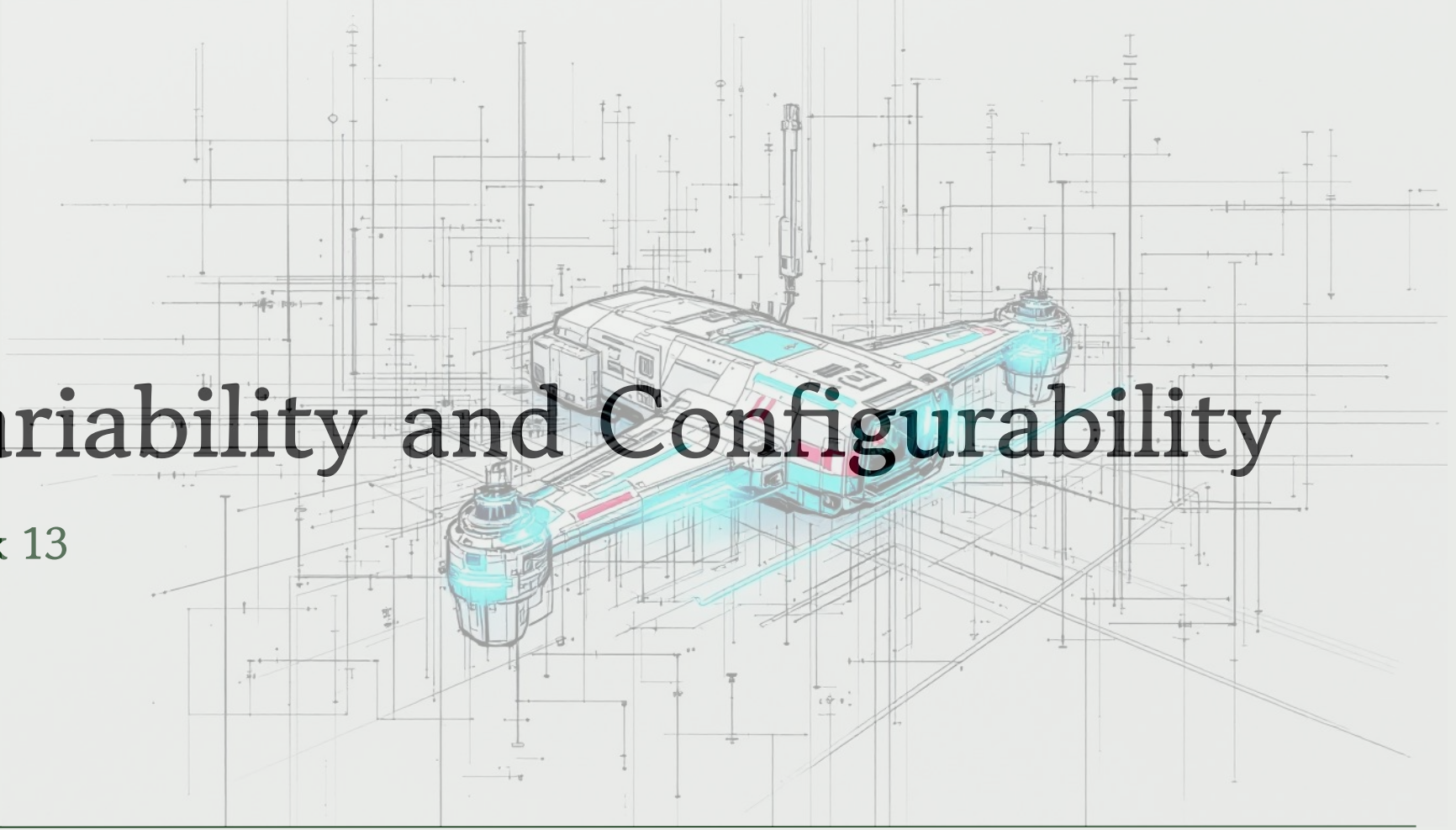
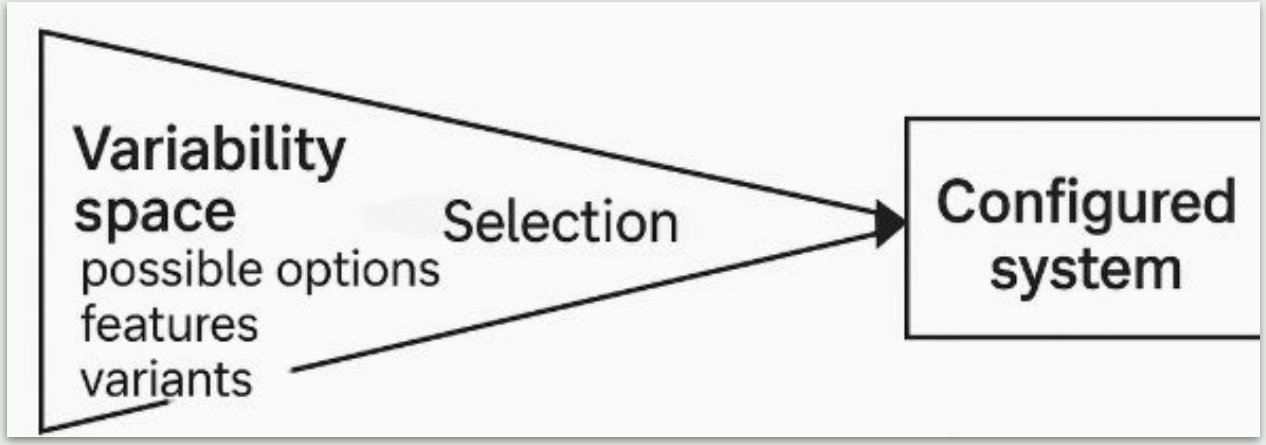


Variability and Configurability

Week 13

Variability and Configurability

This week, we explore how to model **variability** and **product configurations** within a single system model. You'll learn how to use variation and variant elements in SysML v2 to support flexible architectures and multiple drone missions.
By the end of this session, you'll be able to define variation points and build configurable structures and behaviors that adapt to different needs.

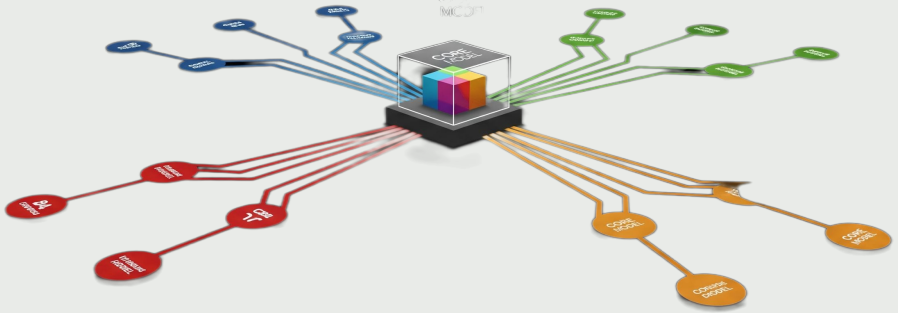


Why Model Variants in SysML?

Systems rarely exist in only one form. Most real-world architectures support multiple versions — differing in features, functions, or physical components. Modeling these **variants** directly in SysML helps you:

- **Avoid duplicating** models for each version
- Maintain a **single source of truth** with embedded flexibility
- Enable consistent **traceability** across configurations
- Plan for **reuse**, customization, and adaptation early in the design phase

Variability modeling turns static architectures into dynamic, configurable system families.



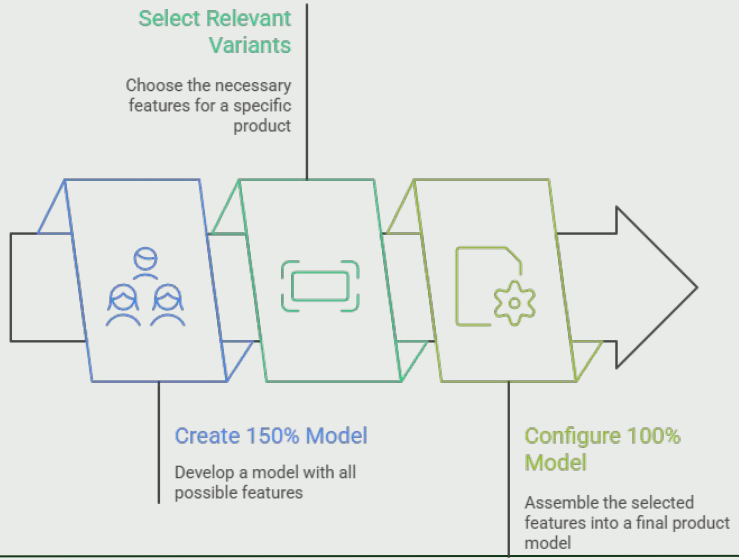
Product Lines vs. Single System Models

A **single system model** describes one specific configuration — useful for prototyping or focused analysis.
A **product line model** captures a whole family of configurations using variation and variant constructs — enabling reuse and flexibility.

Aspect	Single System Model	Product Line Model
Scope	One fixed system	Multiple configurations
Reuse	Low	High — shared architecture
Complexity	Grows linearly	Managed through variation logic
Adaptability	Hard to adapt	Easy to adapt using variants
Best Used For	Proof-of-concept, early demo	Long-term product architecture

The 150% Model in Variability

The 150% model is a system model that contains all possible features, functions, and configurations in one superset representation. It goes beyond a single product configuration by including every structural and behavioral option, even though no single product will use all of them at once. By applying variation points and selecting variants, individual 100% product models are derived from the 150% base. This approach ensures consistency, reuse, and adaptability across a product family.



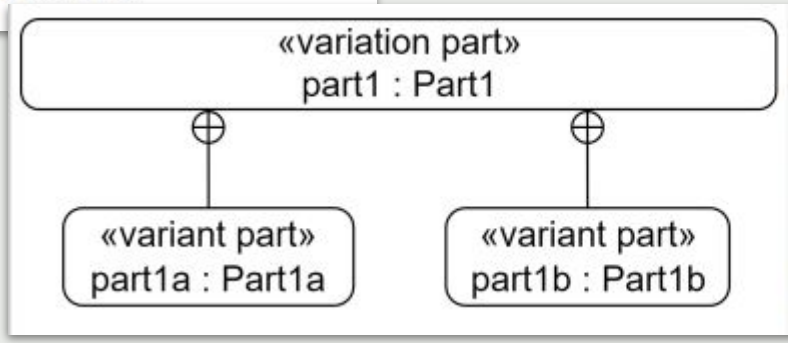
What Is variation in SysML v2?

In SysML v2, a variation identifies a **decision point** in the model where multiple alternatives may be applied. It enables a single model to describe multiple configurations using **structured options**.

- A variation acts as the anchor for one or more variation parts or variation items
- Each variation holds possible variants, which are selected based on design intent
- Used to represent optional features, alternate designs, or product line diversity

variation is the foundation for building flexible, configurable system models.


```
variation part def
PartDef1;
```




```
variation part part1 :
Part1 {
   variant part part1a :
Part1a;
   variant part part1b :
Part1b;
}
```

Understanding variation part and variation item

variation part and variation item define **where** variation occurs in the system:

- **variation part** → introduces variability in the system's structure (e.g., interchangeable sensors or payloads)
- **variation item** → introduces variability in data, flows, or behavioral elements (e.g., telemetry formats, mission types)

Each variation is defined at the model level, and its options are expressed through variant part or variant item.

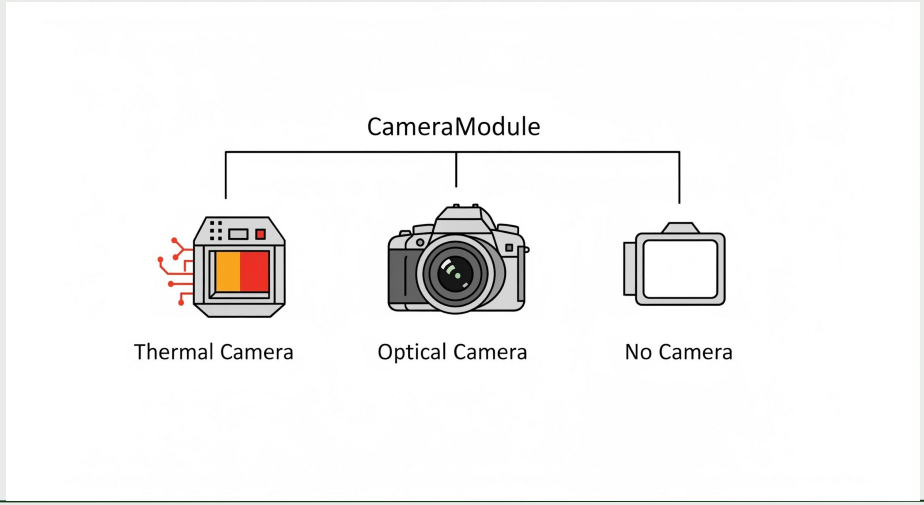
These elements help express **“what might change”** in your system.

Defining variant part and variant item

variant part and variant item specify the concrete alternatives that can populate a variation point:

- **variant part** → defines a structural alternative (e.g., ThermalCamera, OpticalCamera, NoCamera)
- **variant item** → defines a data, flow, or behavior alternative (e.g., ManualFlight, AutoPilot)

Each variant is declared using a variant part def or variant item def, and is linked to its associated variation through select. These define **"the available options"** for each configurable aspect of the system.



How Variant Selection Is Realized in SysML v2

SysML v2 does not use a select keyword — instead, variant selection is realized through **model usage** and **binding**.
You define a variation part or variation item, and in each configuration, you instantiate exactly one of its associated variant part or variant item.

- Variants are **declared** under the variation point
- One variant is **used** in each system configuration
- The selection is handled at the **usage level** — not through a keyword

This makes the model declarative and flexible — selections emerge through context, not imperative syntax.

Modeling Structural Alternatives

To model structural variability, use variation part to represent a **replaceable component**, such as a sensor or payload.

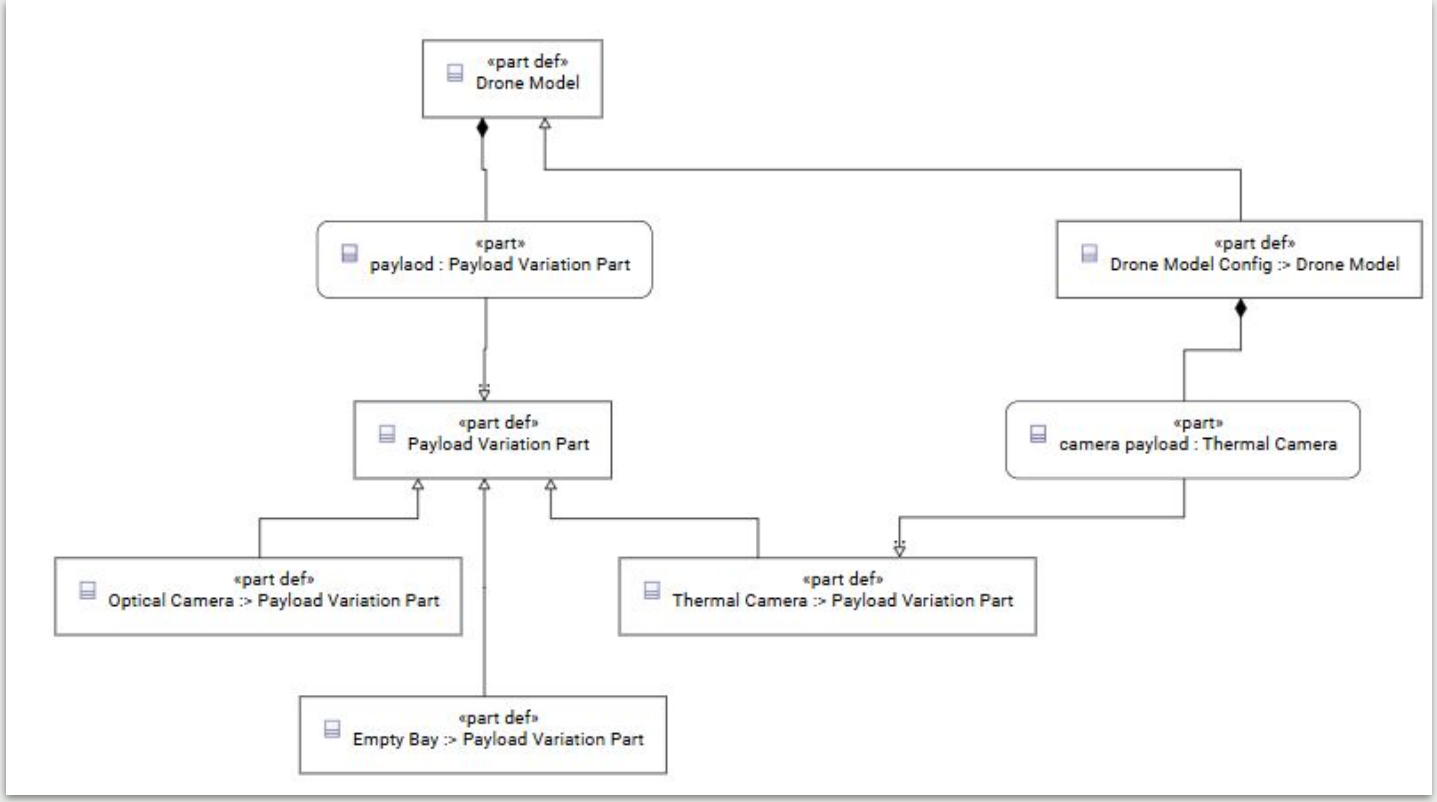
Then define multiple variant part options under that variation, each with its own structure.

Example: variation part PayloadModule

- variant part ThermalCamera
- variant part OpticalCamera
- variant part EmptyBay

Each drone configuration instantiates one variant based on mission needs.

Modeling Structural Alternatives



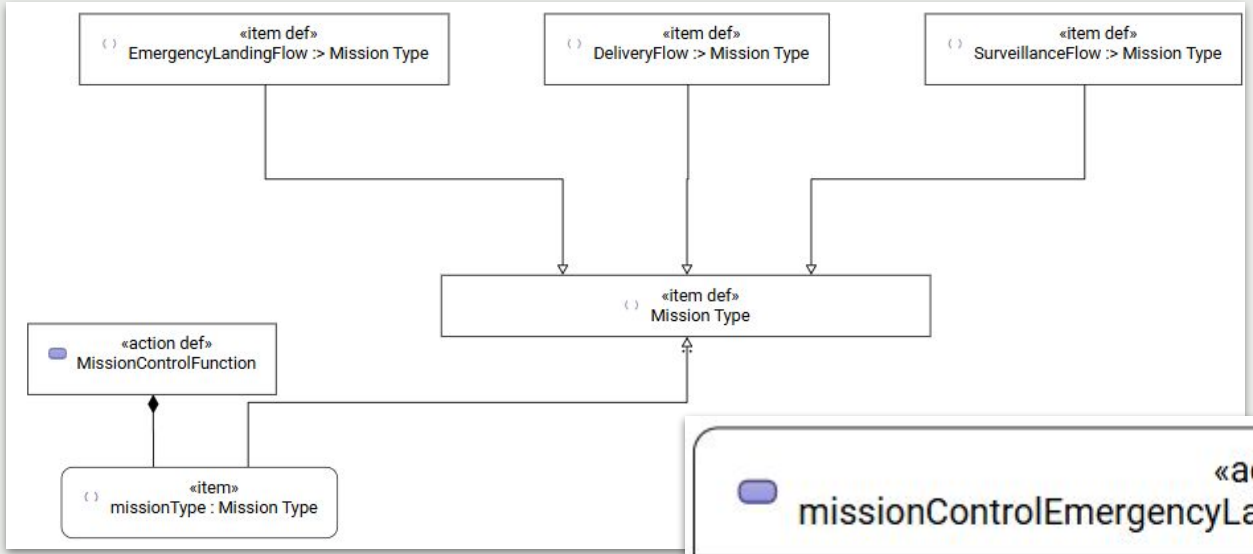
Modeling Behavioral Alternatives

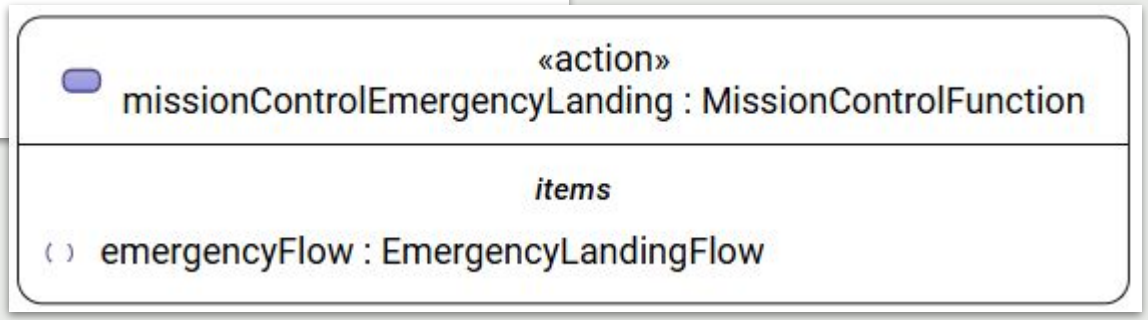
Behavioral variability is modeled using variation item to define a flexible point in a mission sequence.
You then define different variant item options to represent alternative behaviors or logic flows.
Example: variation item MissionType

- variant item SurveillanceFlow
- variant item DeliveryFlow
- variant item EmergencyLandingFlow

Each system configuration instantiates the mission behavior that fits its operational profile.

Modeling Behavioral Alternatives



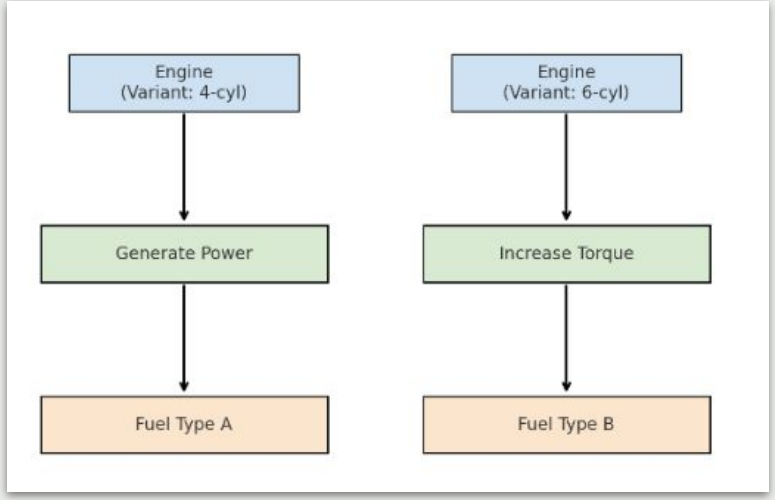


Linking Variants to Parts and Actions

Variants should integrate seamlessly with system architecture and logic:

- **variant part** instances are embedded within the structure of the system
- **variant item** instances participate in behaviors, interactions, or flows
- Variants can be traced to specific functions, constraints, or mission requirements

Linking structure and behavior ensures each configuration is both complete and testable.



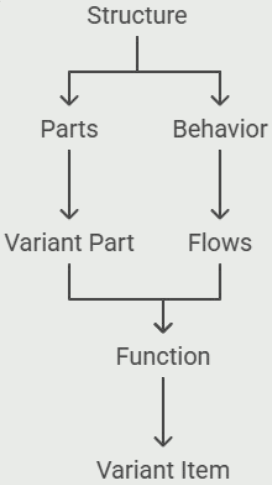


Diagram View — One System, Multiple Configurations

A single SysML v2 model can support multiple drone configurations through variation constructs.

- Example Configurations:
  - **Surveillance Drone:** ThermalCamera + SurveillanceFlow
  - **Delivery Drone:** OpticalCamera + DeliveryFlow
  - **Emergency Mode:** EmptyBay + EmergencyLandingFlow

Each configuration:

- Uses the same base model
- Binds different variants at variation points
- Remains fully traceable and reusable

Tips for Clean Variant Modeling

✅ Keep your variation points focused — don’t overuse variation part or variation item
✅ Group related variants under clear and descriptive names
✅ Avoid mixing structural and behavioral variation in one element
✅ Always link variants to use cases, requirements, or missions
✅ Use consistent naming for variant part def and variant item def

Good variability modeling keeps the system flexible **without clutter**.

Summary of Week 13

This week, you learned how to model configurable system architectures using SysML v2 variation constructs.

🔑 Key Concepts:

- variation part / variant part → for structural variability
- variation item / variant item → for behavioral or logical variability
- Configurations are realized by **instantiating variants** — no select keyword
- One model supports multiple drone types and missions
- Clean variation = reusable, traceable, and adaptable architecture

Variability transforms your model from a system to a **system family**.

QUESTION!