International Council on Systems Engineering
A better world through a systems approach

Exploring the Use of SysMLv2 for Solution Architecture Development with the MagicGrid Framework

Zilvinas Strolia & Aiste Aleksandraviciene

INCOSE International Symposium 2025 | Ottawa, Canada

Zilvinas Strolia

CATIA Cyber Systems Industry Process Senior Specialist

- MS in Economics and BS in Electronics Engineering
- >10 year in Systems Engineering
- Certified professional: CSEP and OCSMP

Aistė Aleksandravičienė

CATIA Cyber Systems Industry Process Senior Specialist

- MS in Software Systems Engineering
- 20 years in Software and Systems Engineering
- Certified professional: ASEP, OCSMP, and UAF MU
- Co-author of MagicGrid Book of Knowledge

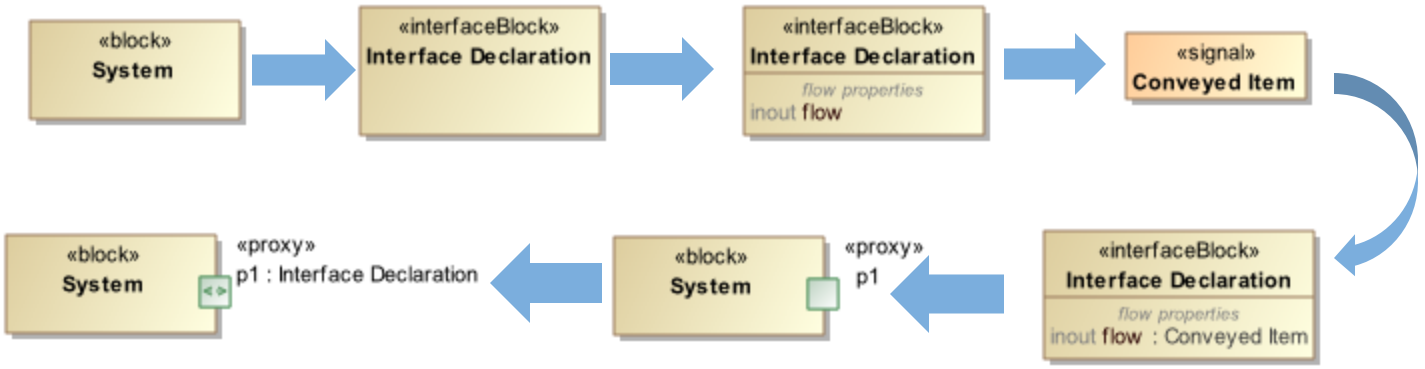
Agenda

- Introduction to SysMLv2
- MagicGrid Overview
- Back to Dublin [Problem Domain]
- From Guinness to Maple Syrup [Solution Domain]
  - Building System Architecture Step-by-Step
  - Textual Notation
- Summary

INTRODUCTION TO SYSMLv2

GREATNESS OF SYSMLv1

Complexity



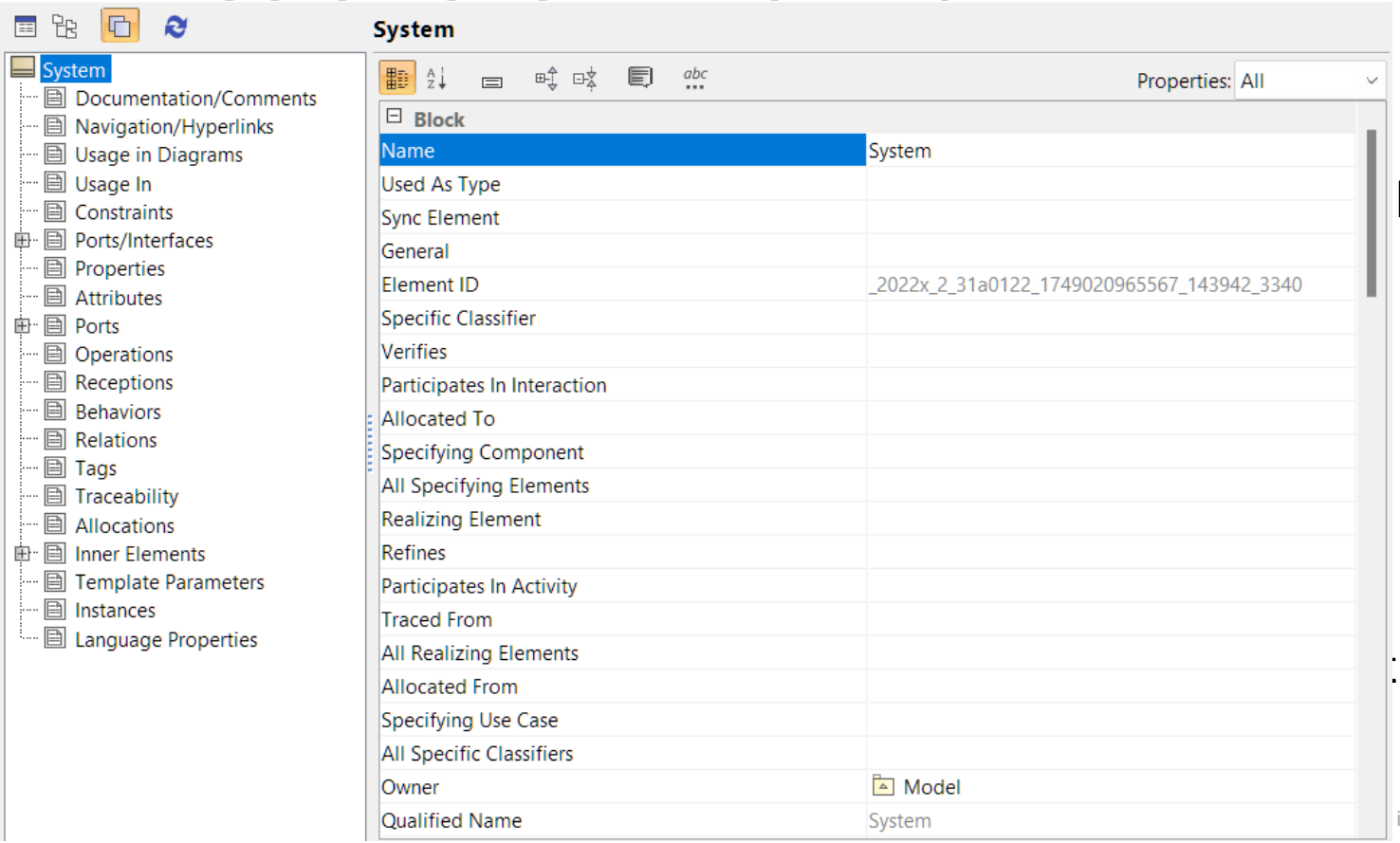
Inconsistent Terminology

Usage	Part Property	Action	Proxy Port	Full Port
Definition	Block	Activity	Interface Block	Block

GRE

Legac
UML

Tool
Interc



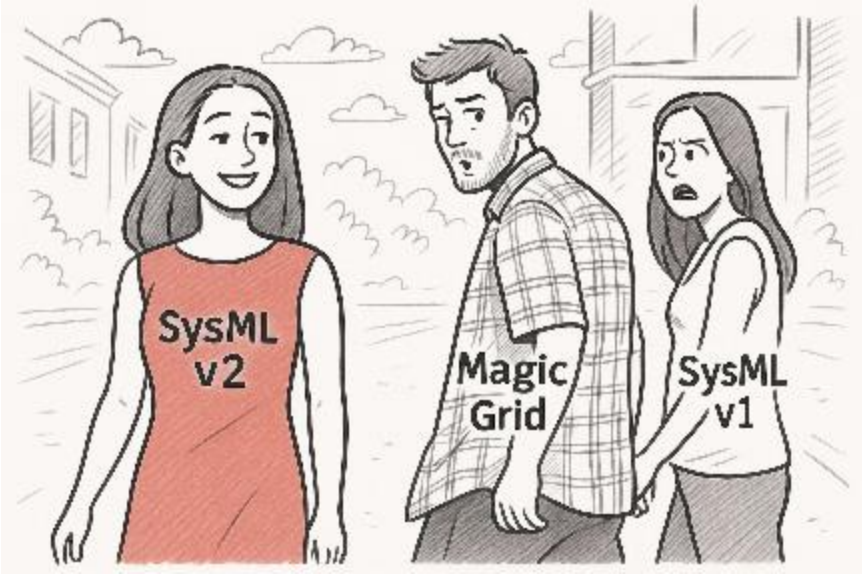
Block

NEW SALVATION – SYSMLv2

- In 2017, OMG initiated the SysMLv2 Request for Proposal process to define a next-generation systems modeling language
- As of 2025, SysMLv2 is under finalization and early adoption
- Key features of SysMLv2:
  - Separation from UML - independent metamodel
  - Supports of graphical, tabular and **textual** notation
  - Formal semantics for better precision
  - API access for tool interoperability
  - Consistent definition and usage pattern throughout the language

NO SILVER BULLET – METHODOLOGY STILL MATTERS

- As its predecessor SysMLv2 is just a language
- To be effective in practice, SysMLv2 must be used together with a modeling methodology
- One of the widely used modeling methodologies is **MagicGrid**
- This paper/presentation examines early use of SysMLv2 in combination with MagicGrid



MAGICGRID OVERVIEW

BASIC INFORMATION

- Framework on how to use SysML with SE projects
- Initial version introduced in 2015
- Widely recognized – 7 papers & MagicGrid BoK approved by external experts
- Tool-agnostic and “vanilla” SysMLv1-compatible
- In alignment with ISO15288 technical processes

MAGICGRID LAYOUT

			Pillar			
Domain			Requirements	Structure	Behavior	Parameters
	Problem	Black Box	Stakeholder Needs	System Context	Use Cases	Measures of Effectiveness
		White Box		Conceptual Subsystems	Functional Analysis	MoEs for Subsystems
	Solution		System Requirements	System Structure	System Behavior	System Parameters
			Subsystem Requirements	Subsystem Structure	Subsystem Behavior	Subsystem Parameters
			Component Requirements	Component Structure	Component Behavior	Component Parameters

PAPER FOCUS – SOLUTION DOMAIN

Domain			Pillar: Requirements	Pillar: Structure	Pillar: Behavior	Pillar: Parameters
Problem	Black Box		Stakeholder Needs	System Context	Use Cases	Measures of Effectiveness
Problem	White Box		Stakeholder Needs	Conceptual Subsystems	Functional Analysis	MoEs for Subsystems
Solution			System Requirements	System Structure	System Behavior	System Parameters
Solution			Subsystem Requirements	Subsystem Structure	Subsystem Behavior	Subsystem Parameters
Solution			Component Requirements	Component Structure	Component Behavior	Component Parameters

BACK TO DUBLIN [PROBLEM DOMAIN]

PREVIOUS RESEARCH

- Presented (as a poster) our **initial findings** on SysMLv2 application for the Problem Domain analysis using MagicGrid

			Pillar			
			Requirements	Structure	Behavior	Parameters
Domain	Problem	Black Box	Stakeholder Needs	System Context	Use Cases	Measures of Effectiveness
		White Box		Conceptual Subsystems	Functional Analysis	MoEs for Subsystems
	Solution		System Requirements	System Structure	System Behavior	System Parameters
			Subsystem Requirements	Subsystem Structure	Subsystem Behavior	Subsystem Parameters
			Component Requirements	Component Structure	Component Behavior	Component Parameters

PREVIOUS RESEARCH (cont.)

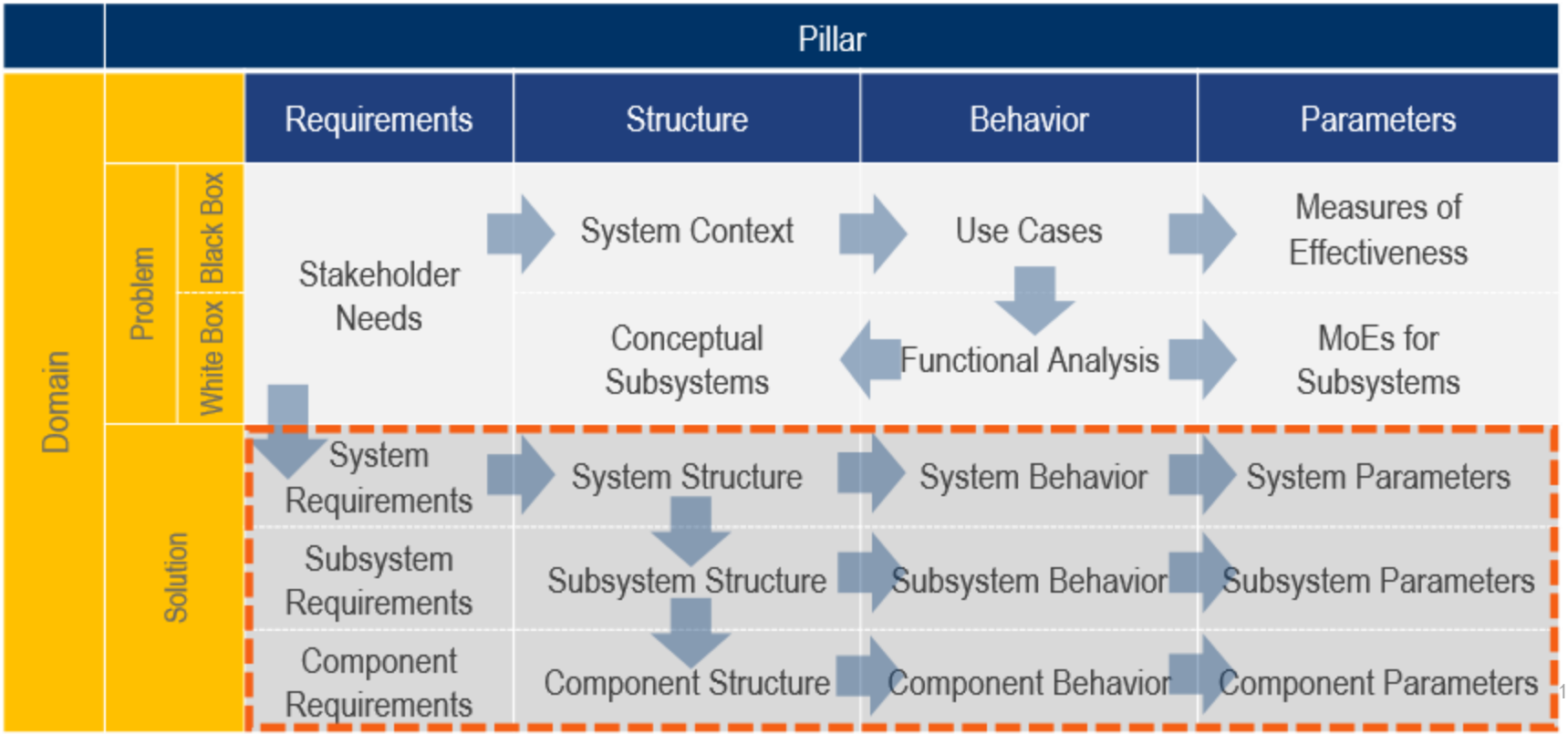
- Initial findings include:
  - MagicGrid cannot be applied with SysMLv2 the same way as it was with SysMLv1
  - SysMLv2 is more open, thus the methodology is even more important than in SysMLv1
  - Want to learn SysMLv2? Forget SysMLv1
- Many things changed since that and some conclusions/findings we made are no longer relevant

FROM GUINNESS TO MAPLE SYRUP [SOLUTION DOMAIN]

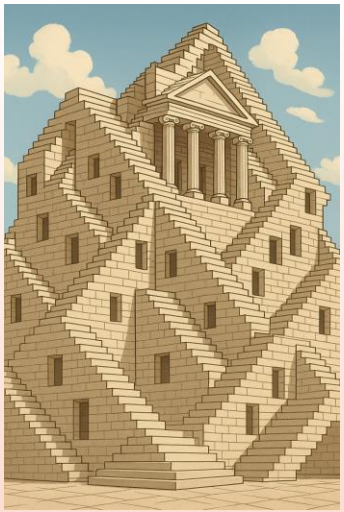
WHERE WE ARE: THE SOLUTION DOMAIN

- Solution Domain defines a cross-discipline logical architecture of the system
- This is not physical or 3D model
- Solution Architecture model consists of multiple levels (system, subsystem, …, component)
- Every level includes requirements, structure, behavior, and parameters

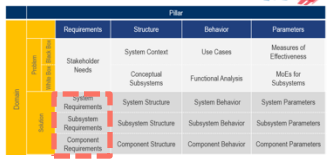
WHERE WE ARE: THE SOLUTION DOMAIN (cont.)



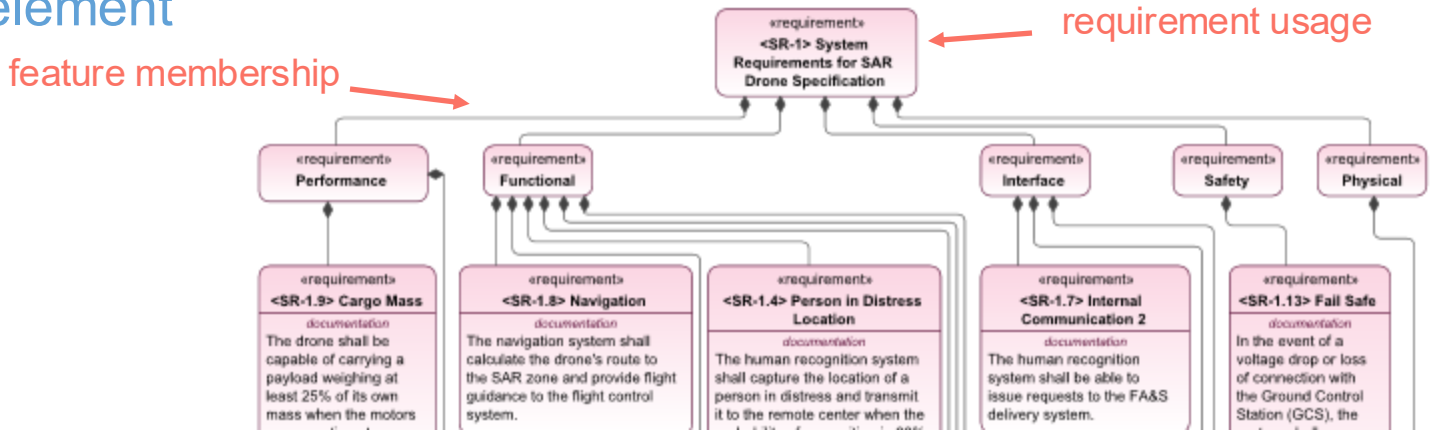
BUILDING SYSTEM ARCHITECTURE STEP-BY-STEP



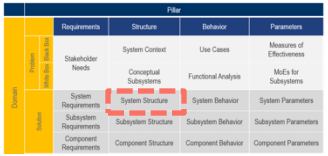
SYSTEM AND OTHER REQUIREMENTS



- System Requirements application does not differ much from how it was used with MagicGrid and SysMLv1 (same conclusion in previous paper)
- System Requirements we suggest to model as requirement usage element

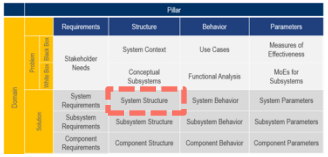


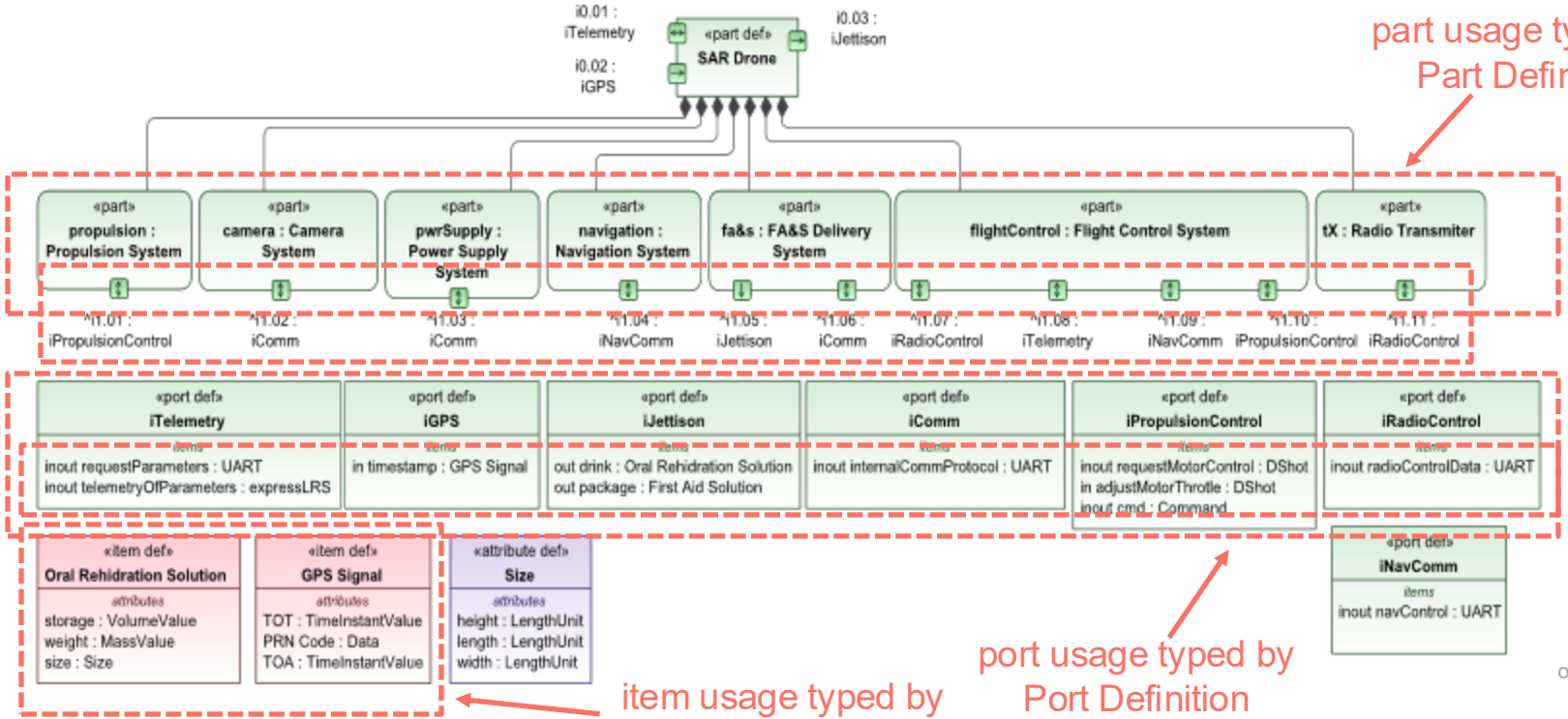
SYSTEM STRUCTURE



- SST team constantly emphasize usage modeling as one of the main advantages of SysMLv2 language
- Up to this cell of MagicGrid framework we tried to create usage models (including previous paper)
- Usually complex systems are not created from the “blank page”
- Structure elements are typically reused – this can only be achieved using part usages with definitions (SysMLv1 modeling style)
- Same applies to port usage/definitions and item usage/definitions

SYSTEM STRUCTURE (cont.)

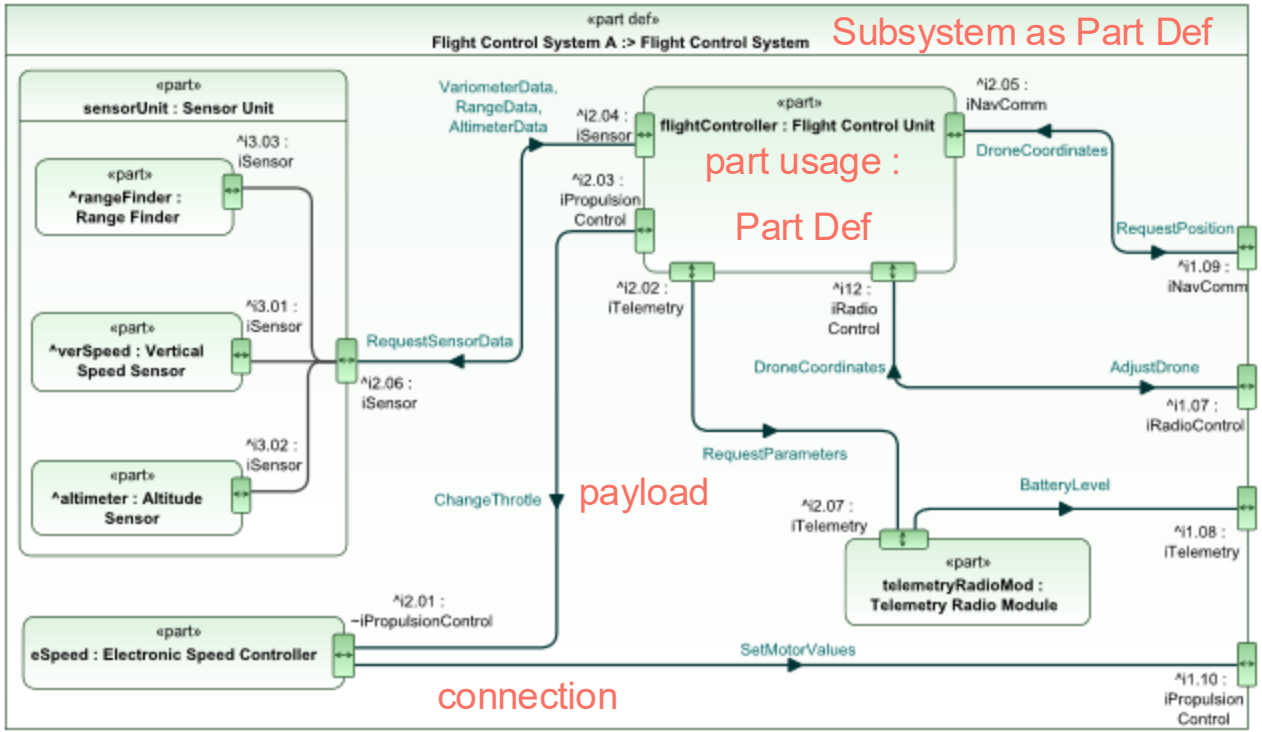




SUBSYSTEM & COMPONENT STRUCTURE

- From language perspective, there's no difference from the System Structure cell – same element types apply
- In SysMLv1 IBDs were used for internal subsystem/component views
- SysMLv2 eliminates IBDs, replacing them with a interconnection view that handles system internal structure
- Interconnection view is a subtype of general view
- Interconnection view ≠ IBD

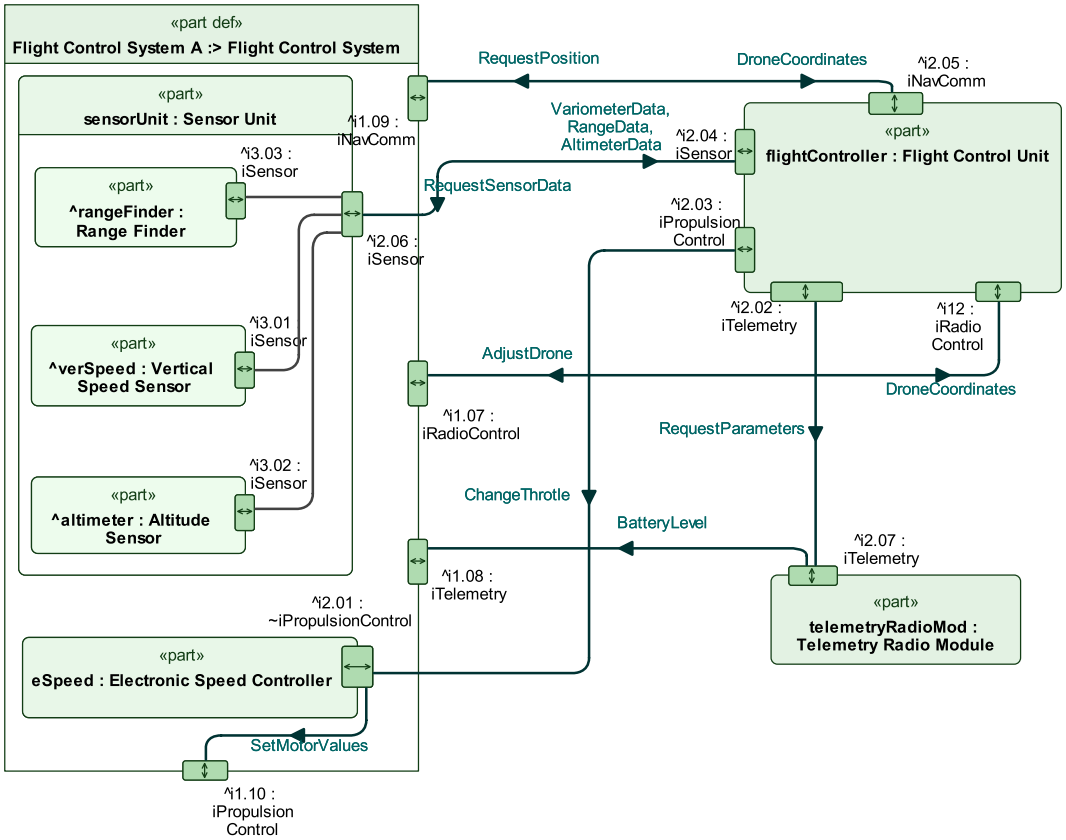
SUBSYSTEM & COMPONENT STRUCTURE (cont.)

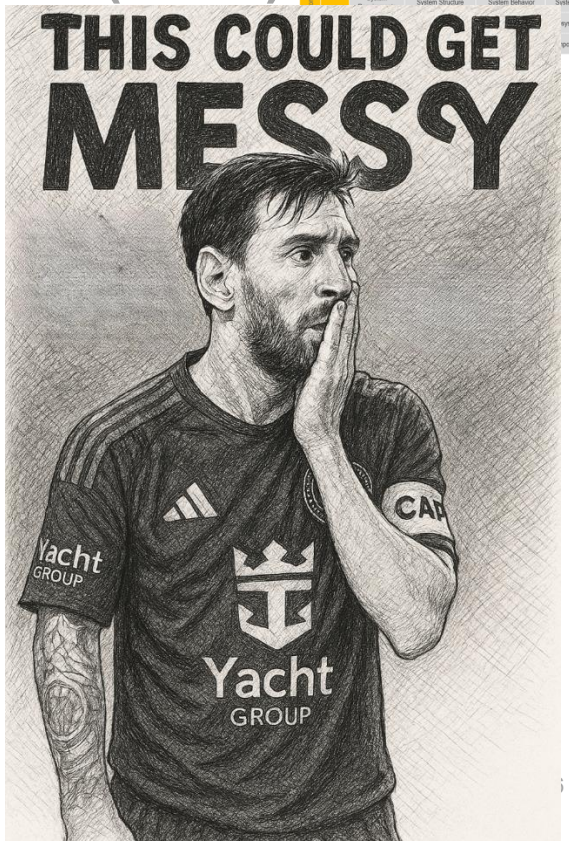


There's no strict requirement to place parts inside each other in this view (!= IBD)

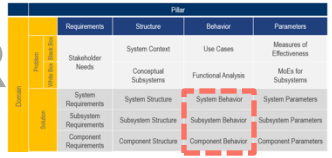
port usage : Port Def

SUBSYSTEM & COMPONENT STRUCTURE (cont.)





SYSTEM / SUBSYSTEM / COMPONENT BEHAVIOR



- MagicGrid with SysMLv1 treats system behavior as the aggregate of its internal subsystem behaviors and there is no dedicated view for that
- SysMLv2 adds no new concepts on this, so the same approach is used
- For inner system elements behavior description either state or action usages should be used depending on user needs
- Usage is preferred as definitions require reusage and we do not expect these behaviors to be reusable in system model scope (debatable)
- Behavior elements in SysMLv2 retains similar modeling logic
- Main difference lies in how these elements are connected to structural elements that behavior they describe

SYSTEM / SUBSYSTEM / COMPONENT BEHAVIOR

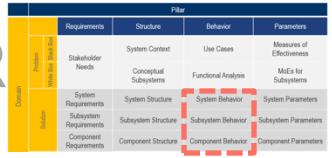
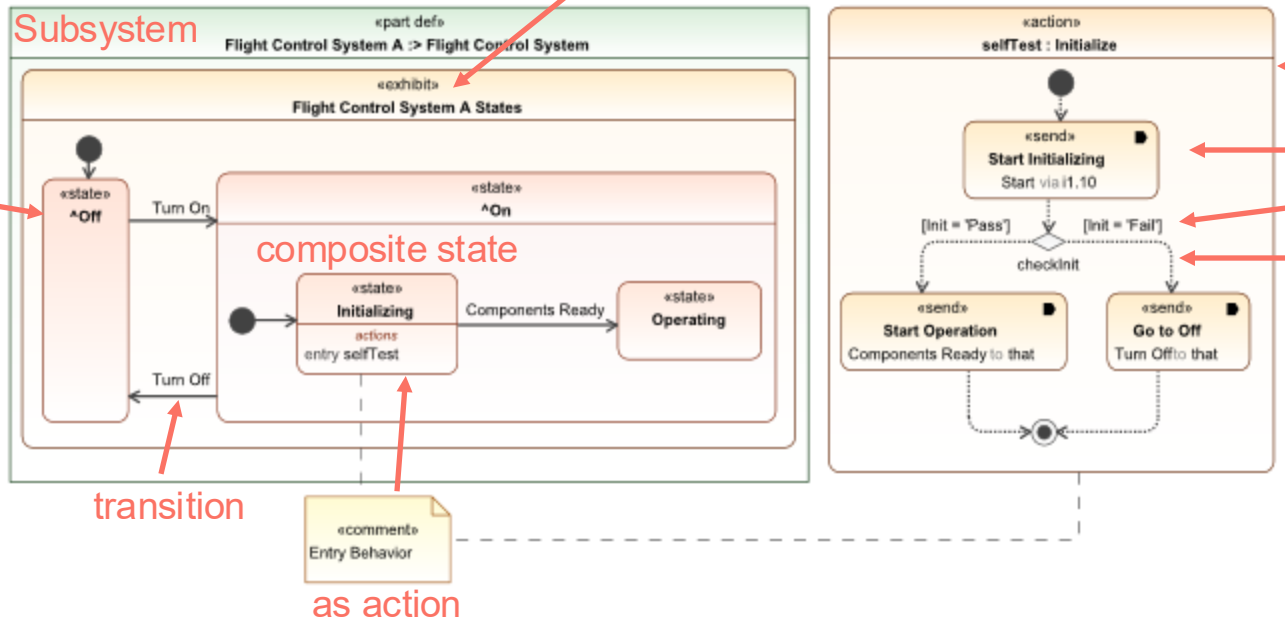


exhibit state – relates state to part usage or def



state usage

transition

as action

action usage

send action

guard expression

succession flow (control flow)

SYSTEM AND OTHER PARAMETERS

- SysMLv2 allows direct expressions on system attributes (e.g., a = b + c), without need of a parametric model
- Similar to programming languages, expressions can be written directly on attributes
- SysMLv2 still supports similar parametric modeling to SysMLv1 (terminology differs)
- We found no need for SysMLv1-style parametric definitions and recommend the simpler approach

SYSTEM AND OTHER PARAMETERS (cont.)

attributes

expression

System

«part def»
SAR Drone

attributes

totalWeight : MassValue = massPS + massCS + massNS + massFAS + massFCS + massPSM + massRT [kg]
ThrustToPower : Real = propulsion.TotalThrustEmpty / (totalWeight + fa&s.cargoWeight)

satisfy requirements

Thrust Weight Ratio

«part def»
Propulsion System

attributes

massPS : MassValue = 0.45 [kg]
TotalThrustEmpty : ForceValue = motorNo * motorThrust [kg]
motorNo : Integer = 4
motorThrust : ForceValue = thrustCoef * airDensity * rotationSpeed ^ 2 * propDiameter ^ 4 / 4 [kg]
thrustCoef : Real = 0.12
airDensity : MassDensityValue = 1.225 [kg·m⁻³]
rotationSpeed : Integer = RPM / 60
RPM : Integer = motorContstant * pwrSupply.SupVoltage
motorContstant : Integer = 1200
propDiameter : LengthUnit = 0.15 [m]

«part def»
Power Supply System

attributes

massPSM : MassValue = 0.5 [kg]
SupVoltage : ElectricPotentialValue = 14.8 [V]

«part def»
FA&S Delivery System

attributes

massFAS : MassValue = 0.15 [kg]
cargoWeight : MassValue = 0.625 [kg]

«part def»
Radio Transmiter

attributes

massRT : MassValue = 0.2 [kg]

«part def»
Camera System

attributes

massCS : MassValue = 0.4 [kg]

«part def»
Flight Control System

attributes

massFCS : MassValue = 0.3 [kg]

«part def»
Navigation System

attributes

massNS : MassValue = 0.2 [kg]

TEXTUAL NOTATION

DIAGRAMS FAIL, TEXT PREVAILS?

- SST presents textual notation as one of the language key characteristics and main advantages
- From MagicGrid perspective we found:
  - complex and cumbersome
  - slow to use and expansive
  - much less expressive than diagraming
- The benefits are derived more from general application than from adherence to a particular methodology
  - simpler interoperability and interchange between modeling tools
  - simpler to use in complex modeling situations
  - AI assistants to the rescue?

DIAGRAMS FAIL, TEXT PREVAILS? (cont.)

Operational Model | DistressSignalHandling | Creation Of Safe Place | Operational Connectivity... | OperationalLibrary [Read-Only]

```
package [$1] 'Operational Model' {
    package [$716] 'Operational Information' {
        #operationalInformation item def [$743] 'Rescue Aborted';
        #operationalInformation item def [$744] 'Rescue Available';
        #operationalInformation item def [$745] 'Rescue Complete';
        #operationalInformation item def [$746] 'Rescue Resource Assigned To Operation';
        #operationalInformation item def [$747] 'Rescue Resource Removed From Operation';
        #operationalInformation item def [$748] 'Rescue Status';
        #operationalInformation item def [$749] 'Rescue Unavailable';
        #operationalInformation item def [$750] Response;
        #operationalInformation item def [$751] 'Safe Place Disestablished';
        #operationalInformation item def [$752] 'Safe Place Disestablishment Req';
        #operationalInformation item def [$753] 'Safe Place Established';
        #operationalInformation item def [$754] 'Safe Place Establishment Req';
        #operationalInformation item def [$755] 'Safe Place Material And Personnell';
        #operationalInformation item def [$756] 'Safe Place Medical Status';
        #operationalInformation item def [$757] 'Safe Place Not Established';
        #operationalInformation item def [$758] 'Safe Place Operational';
        #operationalInformation item def [$759] 'Safe Place Patient Transport Arrival At Hs';
        #operationalInformation item def [$760] 'Safe Place Patient Transport Arrival';
        #operationalInformation item def [$761] 'Safe Place Patient Transport Dispatched';
        #operationalInformation item def [$762] 'Safe Place Patient Transport Request';
        #operationalInformation item def [$763] 'Safe Place Rescued Person Treatment Data';
        #operationalInformation item def [$764] 'SAR Operation Aborted';
        #operationalInformation item def [$765] 'SAR Operation Complete';
        #operationalInformation item def [$766] 'SAR Operation Initiated';
        #operationalInformation item def [$767] 'Search Aborted';
        #operationalInformation item def [$768] 'Search Available';
        #operationalInformation item def [$769] 'Search Resource Assigned To Operation';
        #operationalInformation item def [$770] 'Search Resource Removed From Operation';
        #operationalInformation item def [$771] 'Search Status';
        #operationalInformation item def [$772] 'Search Unavailable';
        #operationalInformation item def [$773] 'Start Safe Place Disestablishment';
        #operationalInformation item def [$774] 'Start Safe Place Establishment';
        #operationalInformation item def [$775] 'Tasking Ack';
        #operationalInformation item def [$776] 'Tasking Nack';
        #operationalInformation item def [$777] Tasking;
        #operationalInformation item def [$778] 'Updated Tracking Info';
        #operationalInformation item def [$779] 'Warning Order';
        #operationalInformation item def [$780] 'You Have Been Found';
    }
}
view [$781] 'Operational Connectivity diagram' : SysMLStandardViews::iv;
```

≈ 1000 LoC

SUMMARY

SUMMARY

- It is still up to debate when to use usage vs definitions in the scope of MagicGrid
- Concept names are consistent and more intuitive from SE perspective (e.g., structure or interface declaration)
- Language semantics can be convoluted (e.g., action vs perform vs perform action, 4 types of connections, etc.)
- SysMLv2 doesn’t provide specific mechanisms for establishing and managing relationships among projects and leaves this for tool vendors
- SysMLv2 must be restricted by methodology even more!

35th Annual INCOSE international symposium

hybrid event

Ottawa, Canada
July 26 - 31, 2025