The Next Generation Systems Modeling Language (SysML v2)

INCOSE Sector III

May 2, 2025

Sanford Friedenthal
OMG SMC Cochair
INCOSE Liaison to OMG
safriedenthal@gmail.com

Copyright © 2019-2025 by Sanford Friedenthal

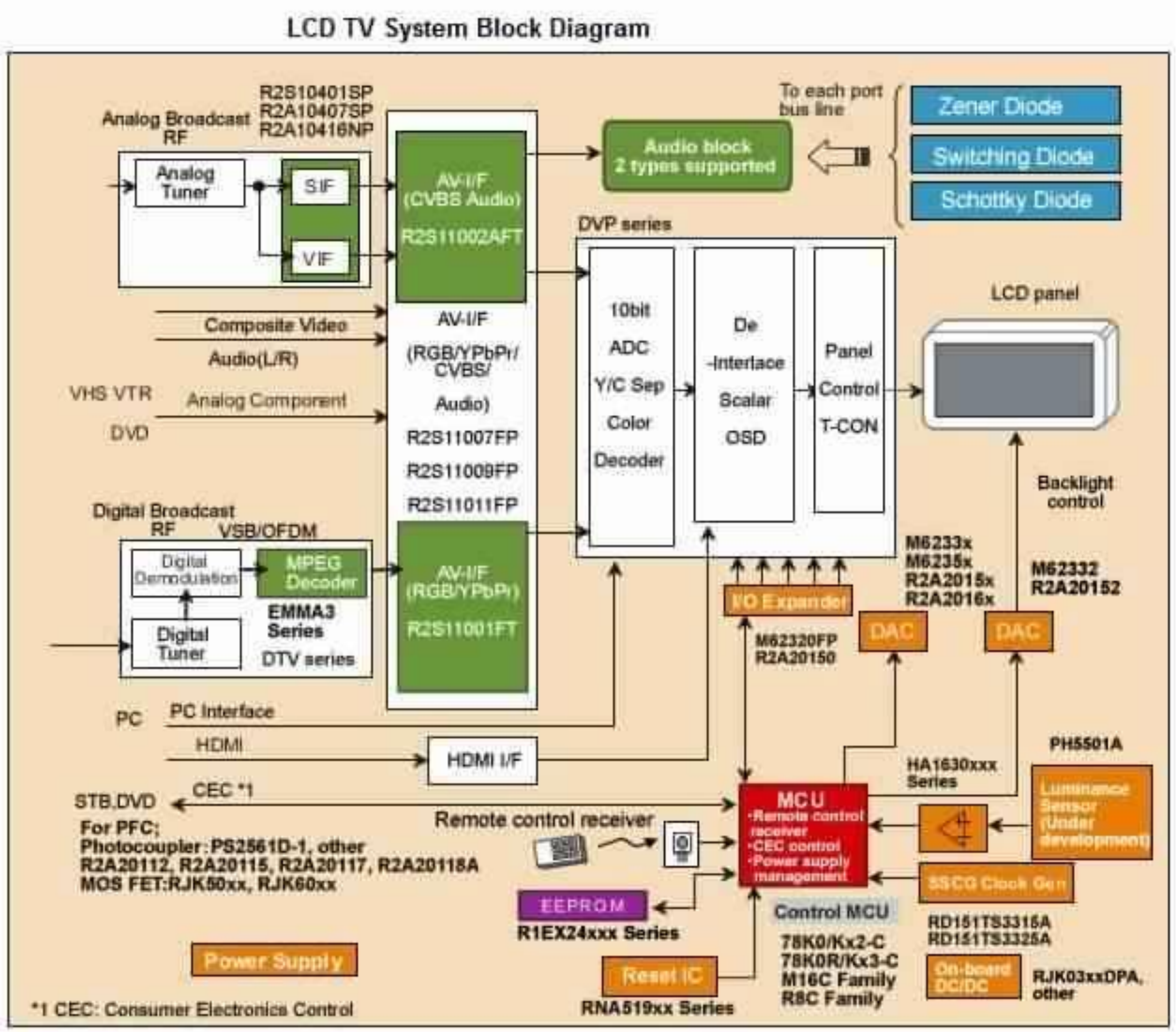
Purpose and Agenda

- Purpose
  - Provide background, status, and overview of SysML v2 and contrast with SysML v1
- Agenda
  - MBSE Background
  - SysML v2 Status
  - SysML v2 Overview & Comparison with SysML v1
  - Extended SysML v2 Community
  - Methodology Considerations
  - SysML v1 to SysML v2 Transition
  - Wrap-up

MBSE Background

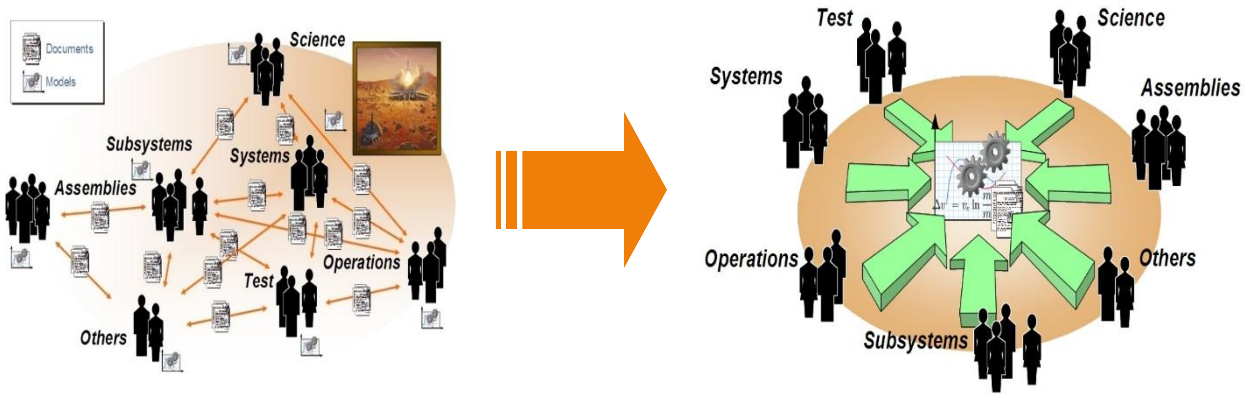
Traditional System Block Diagram

- System architecture captured using informal diagramming notation
- Good domain content but imprecise description of:
  - Component hierarchy
  - Interfaces
  - Functions vs components
  - Succession vs connection
- Disconnected from other system views
- Lack of traceability



Model-Based Systems Engineering (MBSE)

- A systems engineering approach where information about the system is captured in a system model
  - The model is the source of the data and managed throughout the lifecycle
- Contrasts with a document-based approach where the information is captured in a variety of documents, informal diagrams, and spreadsheets
- Provides a more complete, consistent, and traceable system design



From: System specification and design data related through documents

To: Shared system model with multiple views, and connected to discipline models

Source: Jet Propulsion Laboratory

The Future of Systems Engineering is Model-Based

- **Part of the digital transformation**
- Leverage technology advances such as AI
- Full life cycle and from system of systems (SoS) to component level
- Leverage patterns and reference architectures
- Agile system development process supported by automated workflow & configuration management of the digital thread

- Facilitates
  - managing complexity & risk
  - more rapid response to change
  - reuse across programs and design evolution
  - reasoning about & analyzing systems
  - common understanding of the system
  - automated documentation & reporting

Source: INCOSE SE Vision 2035

SysML v2 Status

SysML v2 Status

- SysML v2 RFP issued by the OMG in December, 2017
- SysML v2 specifications developed by the SysML v2 Submission Team (SST)
- SysML v2 beta specifications (i.e., KerML, SysML v2, Systems Modeling API & Services) approved by the OMG in June 2023
- Submitted final specifications for adoption in March, 2025
  - Proceed through OMG process (review, voting, spec editing, …)
    - Planned adoption in June 2025
    - Formal publication anticipated in September 2025
- Tool vendors working on their implementations
  - Anticipate commercial and open-source tools in 2025
- OMG Systems Modeling Community (SMC) in collaboration with INCOSE advancing the modeling practice with SysML v2

SysML v2 Overview
& Comparison with SysML v1

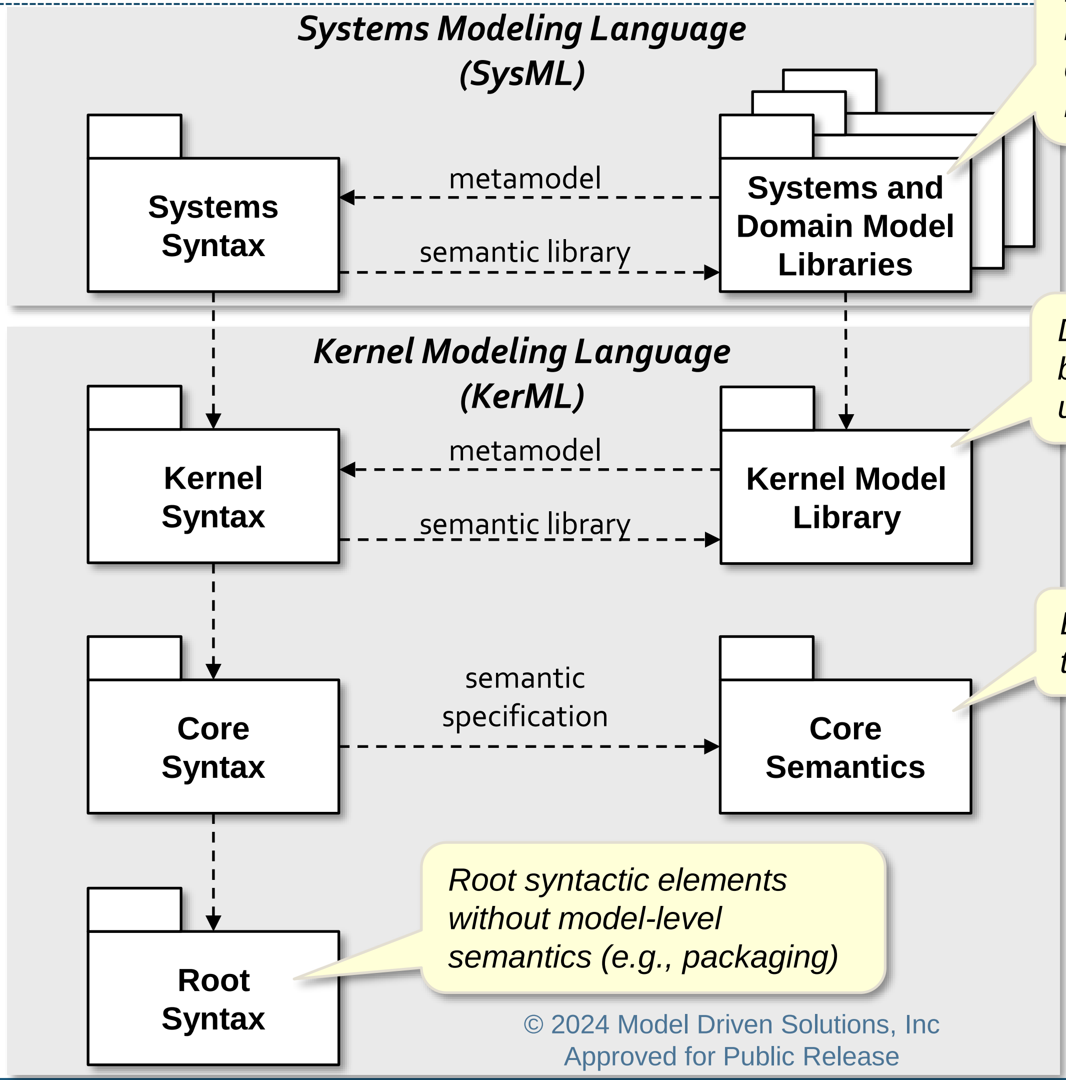
SysML v2 Objectives

- **Increase adoption and effectiveness of MBSE with SysML v2 to support the evolving MBSE practice by enhancing…**
    - Precision and expressiveness of the language
    - Consistency and integration among language concepts
    - Interoperability with other engineering models and tools
    - Usability by model developers and consumers
    - Extensibility to support domain specific applications
    - Migration path for SysML v1 users and implementors

Key Elements of SysML v2

- New Metamodel that is not constrained by UML
  - Preserves most of UML modeling capabilities with a focus on systems modeling
  - Grounded in formal semantics
- Robust visualizations based on flexible view & viewpoint specification
  - Graphical, Tabular, Textual
- Standardized API to access the model

SysML v2 Language Architecture



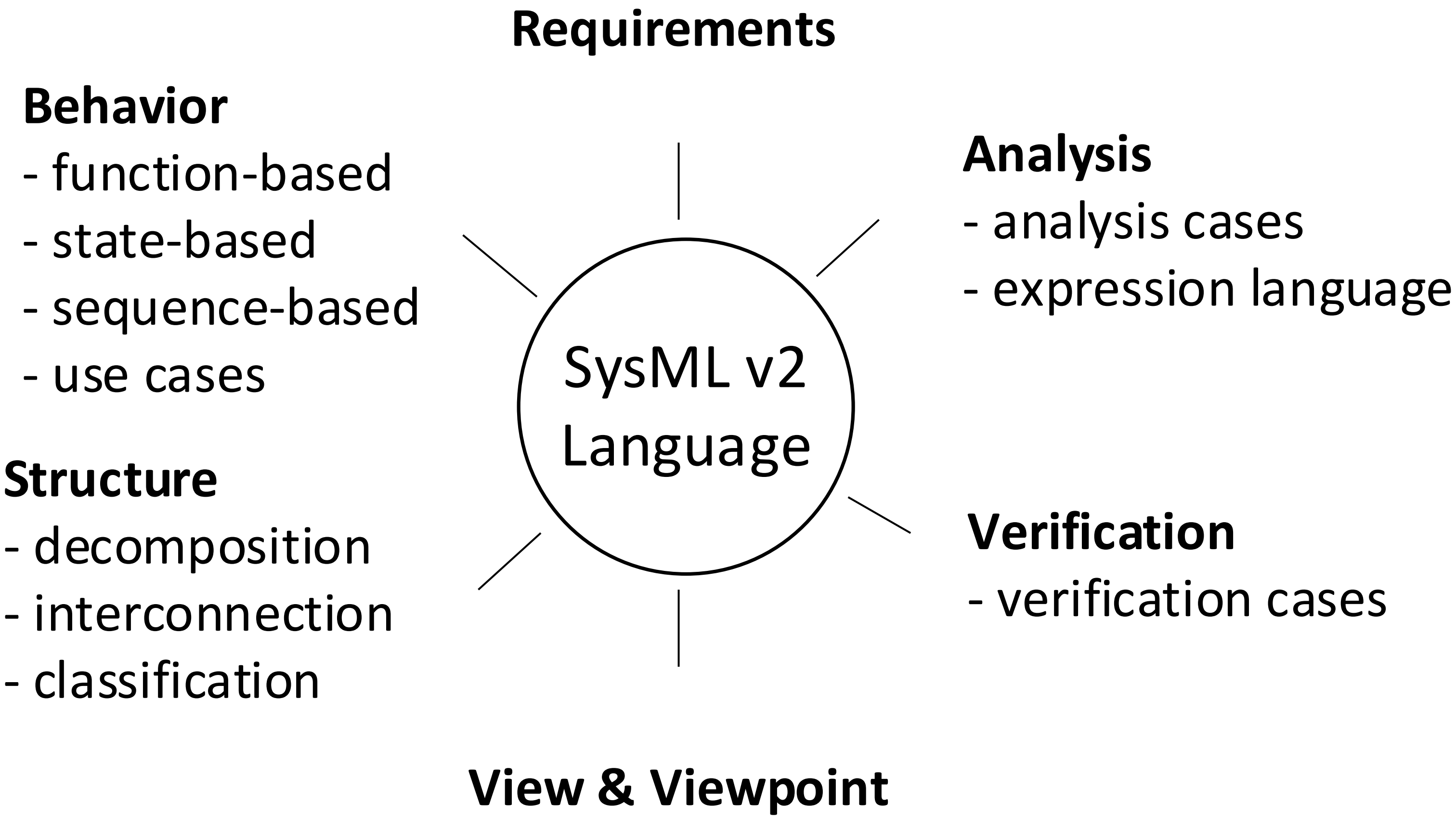
Declarative semantic base elements and domain-specific libraries modeled using SysML

Declarative semantic base elements modeled using KerML

Direct semantic mapping to formal logic

Root syntactic elements without model-level semantics (e.g., packaging)

SysML v2 Language Capabilities



Requirements

Behavior
- function-based
- state-based
- sequence-based
- use cases

Analysis
- analysis cases
- expression language

SysML v2 Language

Structure
- decomposition
- interconnection
- classification

Verification
- verification cases

View & Viewpoint

SysML v2 Reuse Pattern

- Definition and usage
  - A definition element defines an element such as a part, action, or requirement
  - A usage element is a usage of a definition element in a particular context
  - Pattern is applied consistently throughout the language

SysML v2 to v1
Terminology Mapping (partial)

SysML v2	SysML v1
part / part def	part property / block
attribute / attribute def	value property / value type
port / port def	proxy port / interface block
action / action def	action / activity
state / state def	state / state machine
constraint / constraint def	constraint property / constraint block
requirement / requirement def	requirement
connection / connection def	connector / association block
view / view def	view

SysML v2 applies a consistent pattern of definition and usage

Acknowledgement

- The SysML v2 diagrams in this slide set were created using an early version of Dassault Systèmes Cameo Systems Modeler / CATIA Magic Cyber Systems Engineer.

Add one of these to every slide that has diagrams produced from CSM/MCSE:

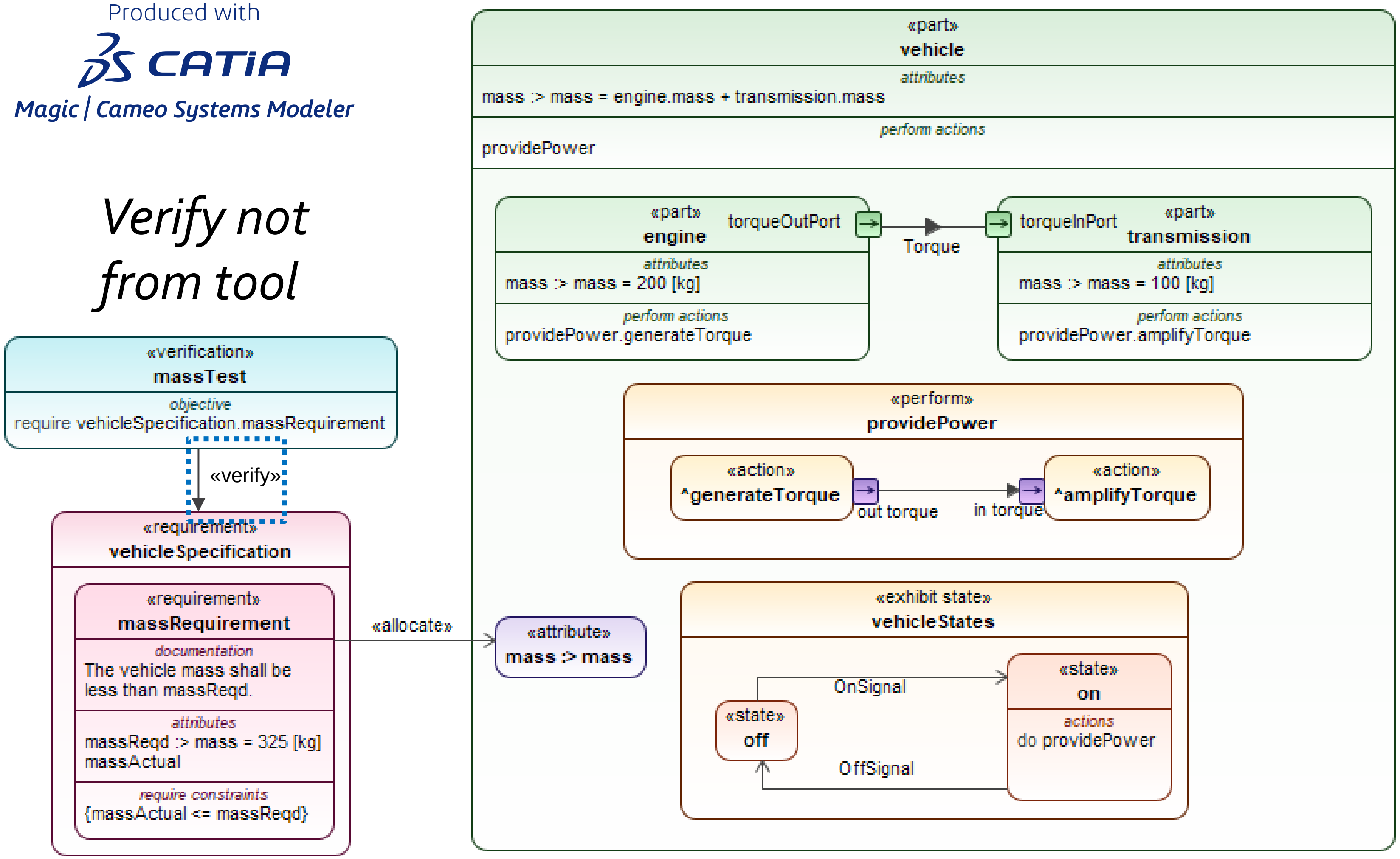
Produced with
CATIA
Magic | Cameo Systems Modeler

Produced with CATIA | Magic
Cameo Systems Modeler

Vehicle Example

Produced with CATIA Magic | Cameo Systems Modeler

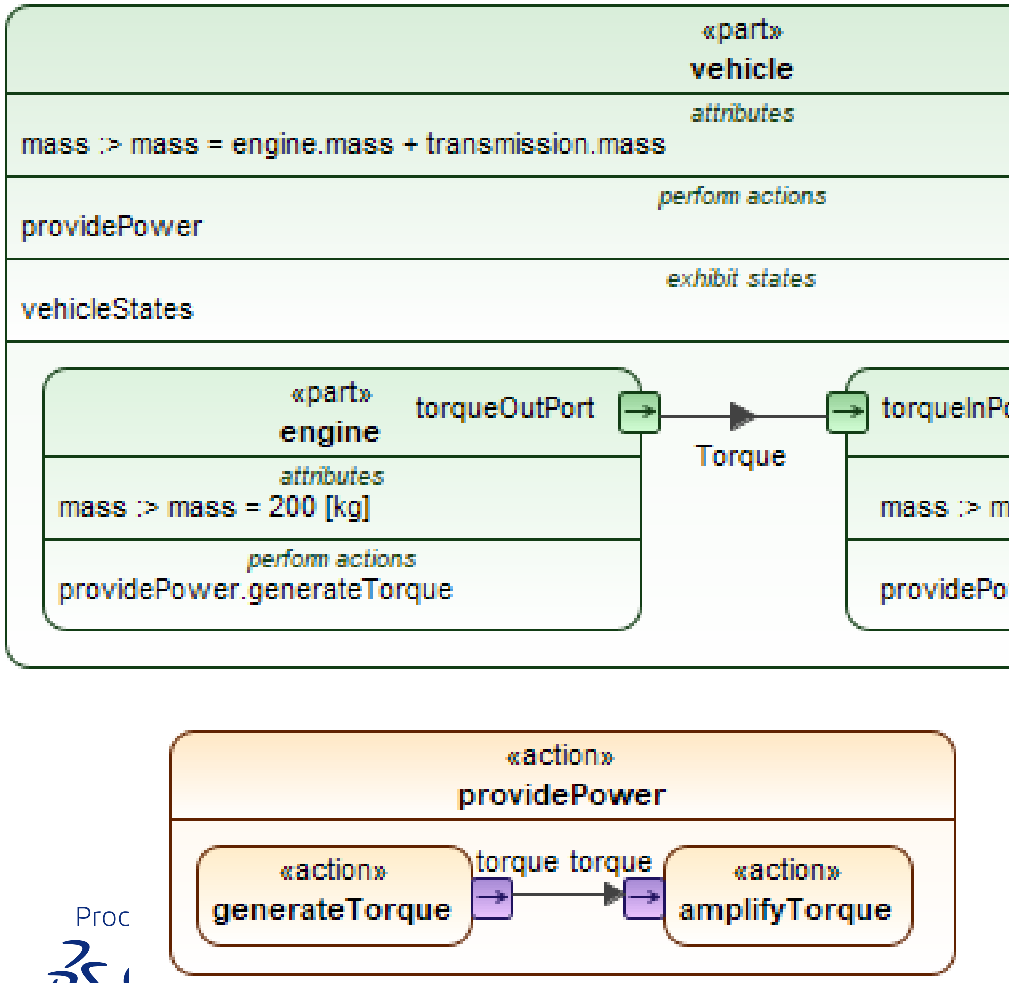
Verify not from tool



```
package vehicle_example {
  view vehicle_example;
  private import SI::*;
  item def OnSignal;
  item def OffSignal;
  attribute def Torque;

  part vehicle {
    attribute mass :> ISQBase::mass = engine.mass + transmission.mass;
    perform providePower;
    part engine {
      attribute mass :> ISQBase::mass = 200 [kilogram];
      port torqueOutPort {
        out attribute torque : Torque;
      }
      perform providePower.generateTorque;
    }
    part transmission {
      attribute mass :> ISQBase::mass = 100 [kilogram];
      port torqueInPort {
        in attribute torque : Torque;
      }
      perform providePower.amplifyTorque;
    }
    connect engine.torqueOutPort to transmission.torqueInPort {
      flow of Torque from source.torque to target.torque;
    }
    exhibit state vehicleStates {
      state off;
      state on {
        do providePower;
      }
      transition off_to_on first off accept OnSignal then on;
      transition on_to_off first on accept OffSignal then off;
    }
  }
  action providePower {
    action generateTorque { out attribute torque; }
    action amplifyTorque { in attribute torque; }
    flow generateTorque.torque to amplifyTorque.torque;
  }
  requirement vehicleSpecification {
    requirement massRequirement {
      doc /* The vehicle mass shall be less than massReqd. */
      attribute massReqd :> mass = 325 [kilogram];
      attribute massActual;
      require constraint {massActual <= massReqd}
    }
  }
  allocate vehicleSpecification.massRequirement to vehicle.mass;
  verification massTest {
    objective {
      verify vehicleSpecification.massRequirement;
    }
  }
}
```

Vehicle Example Graphical/Textual Correspondence



«part»
vehicle

attributes
mass :> mass = engine.mass + transmission.mass

perform actions
providePower

exhibit states
vehicleStates

«part»
engine

attributes
mass :> mass = 200 [kg]

perform actions
providePower.generateTorque

torqueOutPort

Torque

torqueInPo

mass :> m

providePo

«action»
providePower

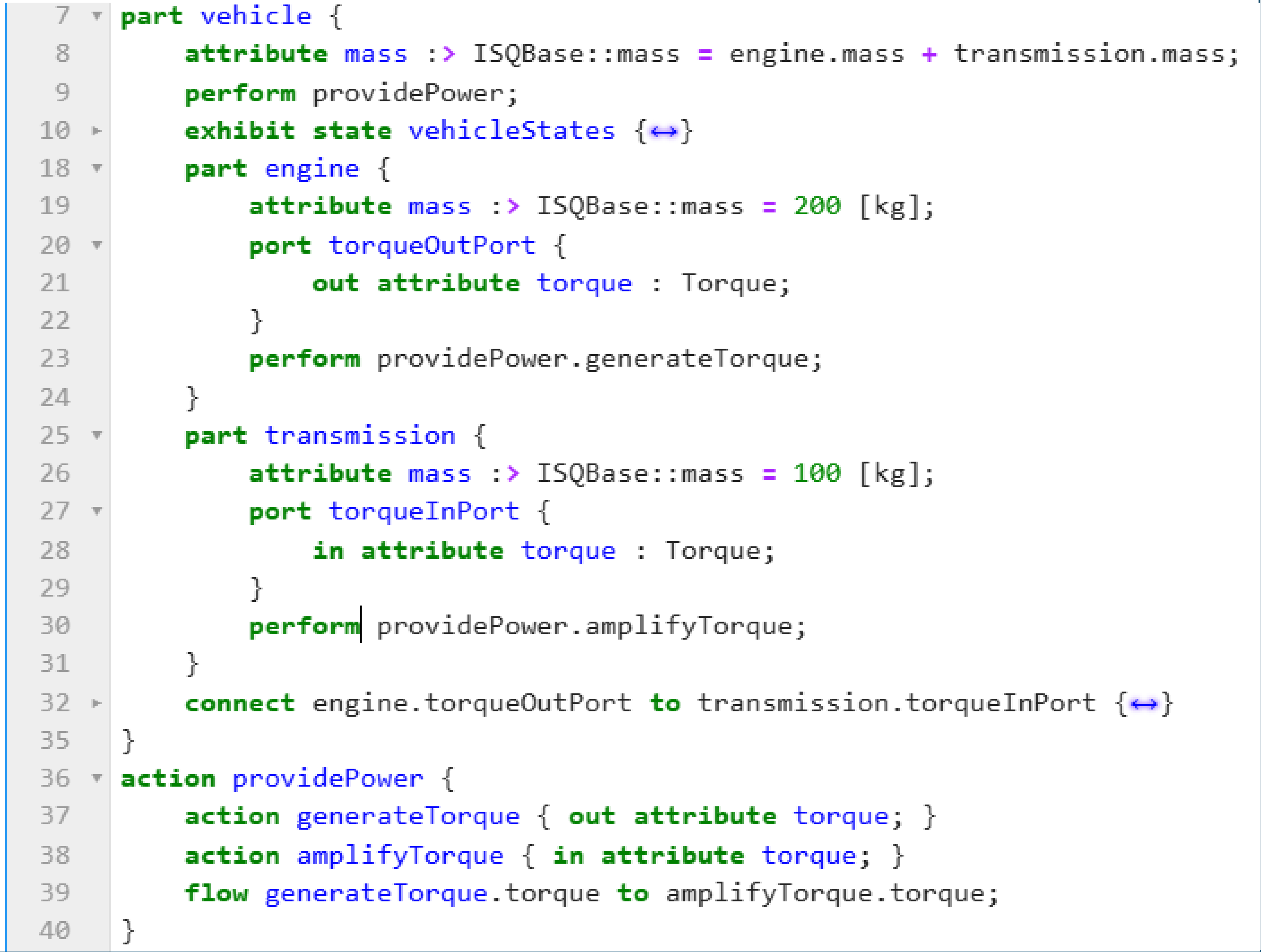
«action»
generateTorque

torque torque

«action»
amplifyTorque

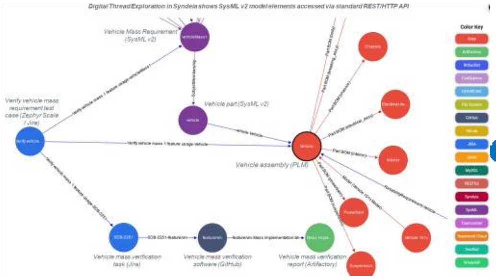
Proc

Magic | Cameo Systems Modeler



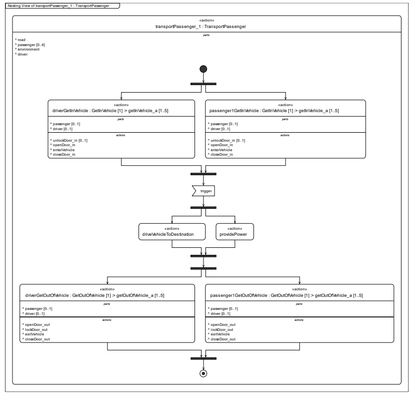
```
part vehicle {
    attribute mass :> ISQBase::mass = engine.mass + transmission.mass;
    perform providePower;
    exhibit state vehicleStates {↔}
    part engine {
        attribute mass :> ISQBase::mass = 200 [kg];
        port torqueOutPort {
            out attribute torque : Torque;
        }
        perform providePower.generateTorque;
    }
    part transmission {
        attribute mass :> ISQBase::mass = 100 [kg];
        port torqueInPort {
            in attribute torque : Torque;
        }
        perform providePower.amplifyTorque;
    }
    connect engine.torqueOutPort to transmission.torqueInPort {↔}
}
action providePower {
    action generateTorque { out attribute torque; }
    action amplifyTorque { in attribute torque; }
    flow generateTorque.torque to amplifyTorque.torque;
}
```

Connecting SysML v2 through the standard API



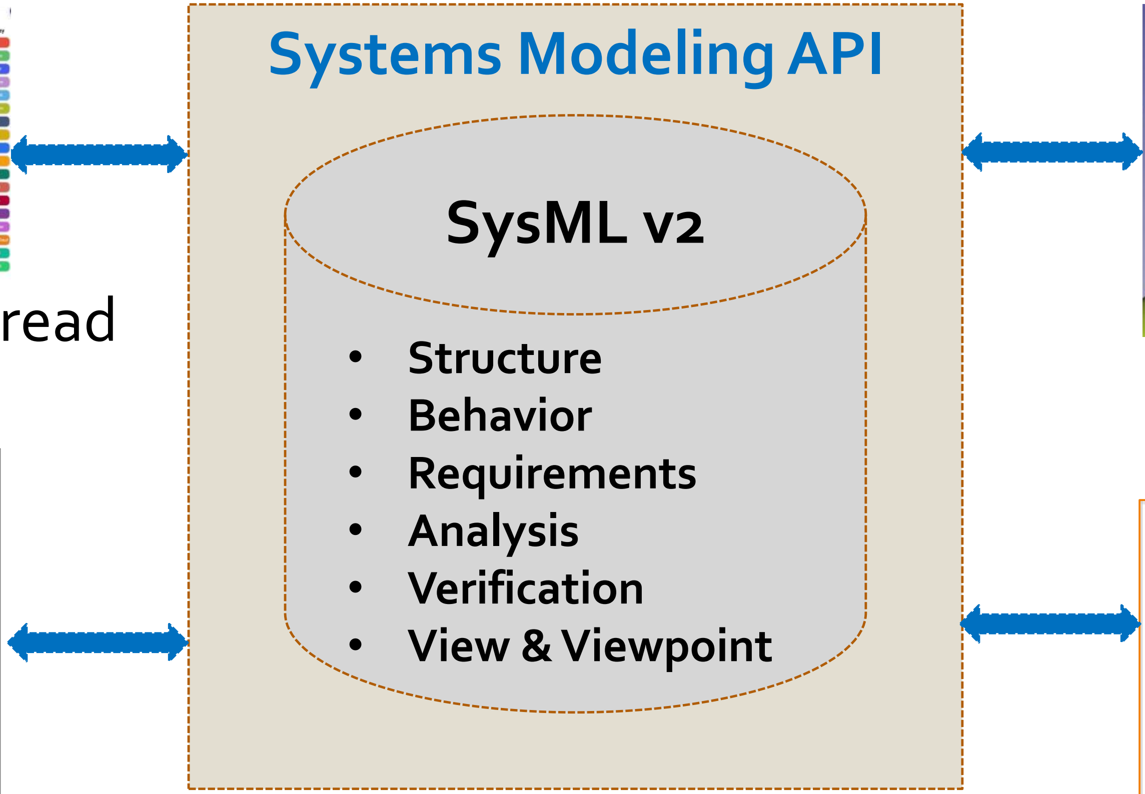
CM of the Digital Thread

Source: Syndeia with SysML v2



Graph Visualization

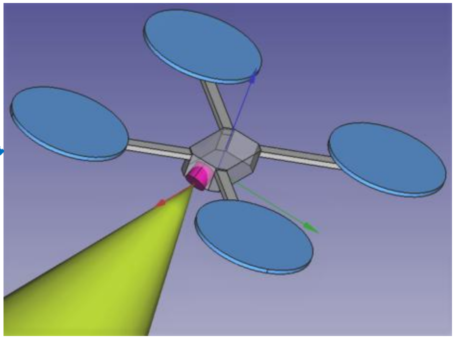
Source: Tom Sawyer with SysML v2



Systems Modeling API

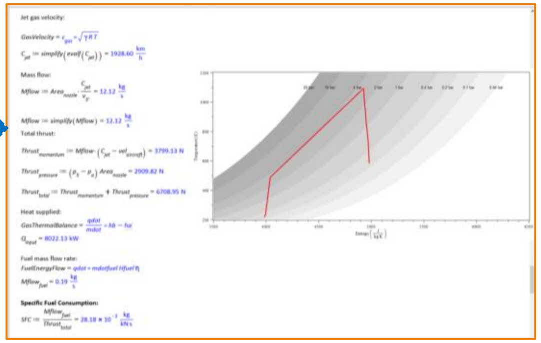
SysML v2

- Structure
- Behavior
- Requirements
- Analysis
- Verification
- View & Viewpoint



CAD/CAD Viewer

Source: FreeCAD with SysML v2



Analysis Solver

Source: Maple with SysML v2

Comparing SysML v2 with SysML v1

- **Simpler to learn and use**
  - Systems engineering concepts designed into metamodel versus added-on
  - Consistent definition and usage pattern
  - More consistent terminology
  - Ability to decompose parts, actions, …
- **More precise**
  - Textual syntax and expression language
  - Formal semantic grounding
  - Requirements as constraints
  - Integrated structure and behavior
- **More expressive**
  - Specialization of behavior
  - Variant modeling
  - Analysis case and trade-off analysis
  - Individuals, snapshots, time slices
  - More robust quantitative properties (e.g., vectors, ..)
  - Simple geometry
  - Query/filter expressions
  - Metadata
- **More extensible**
  - Simpler language extension capability
    - Based on model libraries
- **More interoperable**
  - Standardized API

Extended SysML v2 Community

OMG Systems Modeling Community (SMC)

https://www.omg.org/communities/systems-modeling-community.htm

- Working groups focused on advancing the SysML v2 specifications and practices
  - API & Services WG – Manas Bajaj
  - Conformance WG - Vince Molnár
  - Digital Engineered Verification WG – Mark Malinoski
  - Execution WG – Richard Page
  - Formal Methods WG - Vince Molnár
  - Graphical Specification WG – Sandy Friedenthal
  - Real-Time Embedded Safety-Critical Systems WG - Jerome Hugues, Gene Shreve
  - Reference Implementation WG – Ed Seidewitz
  - Semantics WG – Conrad Bock, Karen Ryan
  - SysML v2 Certification WG – Rick Steiner
  - Transformation & Interoperability WG – Yves Bernard
  - User WG – Sandy Friedenthal

OMG Systems Modeling Community (SMC) SysML v2 Public Repositories

- Open-source repositories
  - https://github.com/Systems-Modeling
- Current SysML v2 release
  - https://github.com/Systems-Modeling/SysML-v2-Release
- Release content
  - Specification documents for KerML, SysML and API
  - Training material for SysML textual notation
  - Training material for SysML graphical notation
  - Example models in textual notation (sysml/src)
  - Pilot implementation
    - Installer for Jupyter tooling
    - Installation site for Eclipse plug-in
  - Web access to prototype repository via SysML v2 API
- Google group for comments and questions
  - https://groups.google.com/g/SysML-v2-Release
    (to request membership, provide name, affiliation and interest)

INCOSE

- Broad OMG/INCOSE Collaboration via Memorandum of Understanding (MOU)
- Participation with INCOSE (Chapters, Working Groups, IW, IS)
- OMG Systems Modeling Standards Supported by INCOSE
  - Systems Modeling Language (SysML)
  - Unified Architecture Framework (UAF)
  - Risk Analysis and Assessment Modeling Language (RAAML)
  - Cubesat System Reference Model (CSRM)

SysML v2 Vendor Support

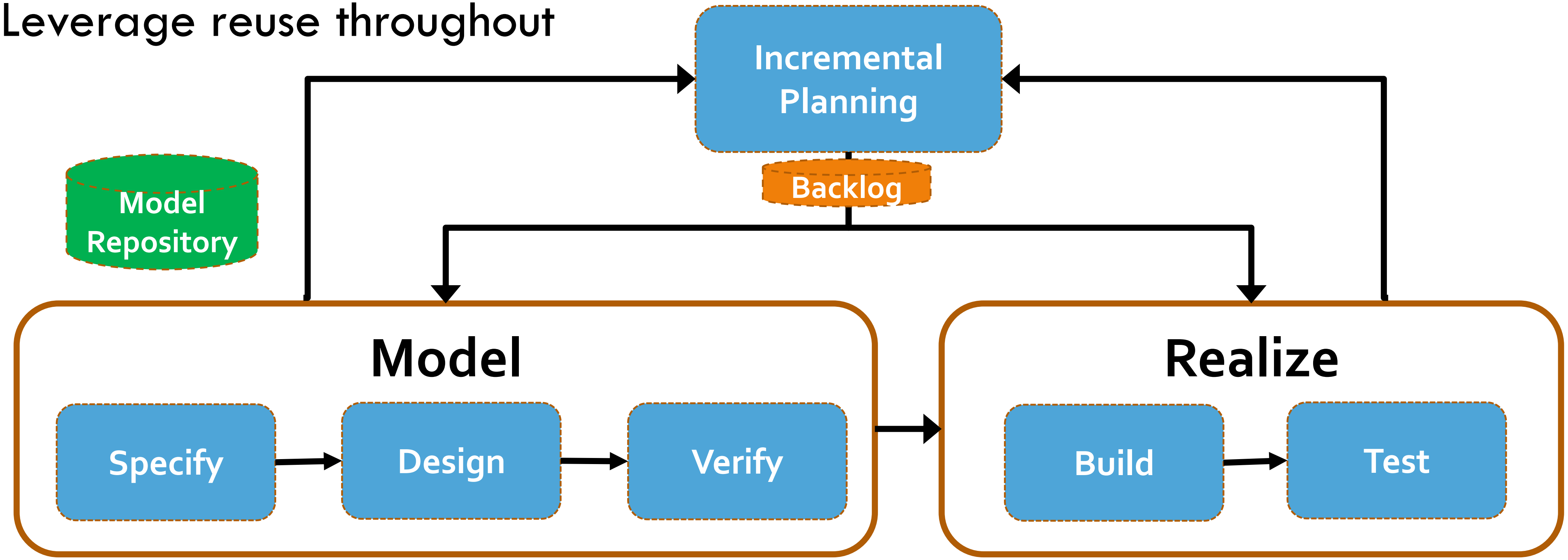
- The following vendors participated in the Vendor Roadmap Session at the INCOSE IW
  - Ansys *
  - Celedon
  - Dassault Systèmes *
  - Ellidiss
  - IBM *
  - IncQuery *
  - Intercax *
  - LieberLieber
  - MathWorks *
  - Mgnite Inc. *
  - Obeo *
  - PTC
  - SBE Vision
  - Sensmetry *
  - Siemens
  - Sodius Wilert
  - Sparx
  - Tom Sawyer Software *
  - Vendor Booths at the SysML v2 Summit *
    - Qualtech Systems *
    - SysGit *
    - Model Driven Solutions *

SysML v2 Vendor Roadmap Videos from INCOSE IW at mbse:incose_mbse_iw_2025 [MBSE Wiki]

Methodology Considerations

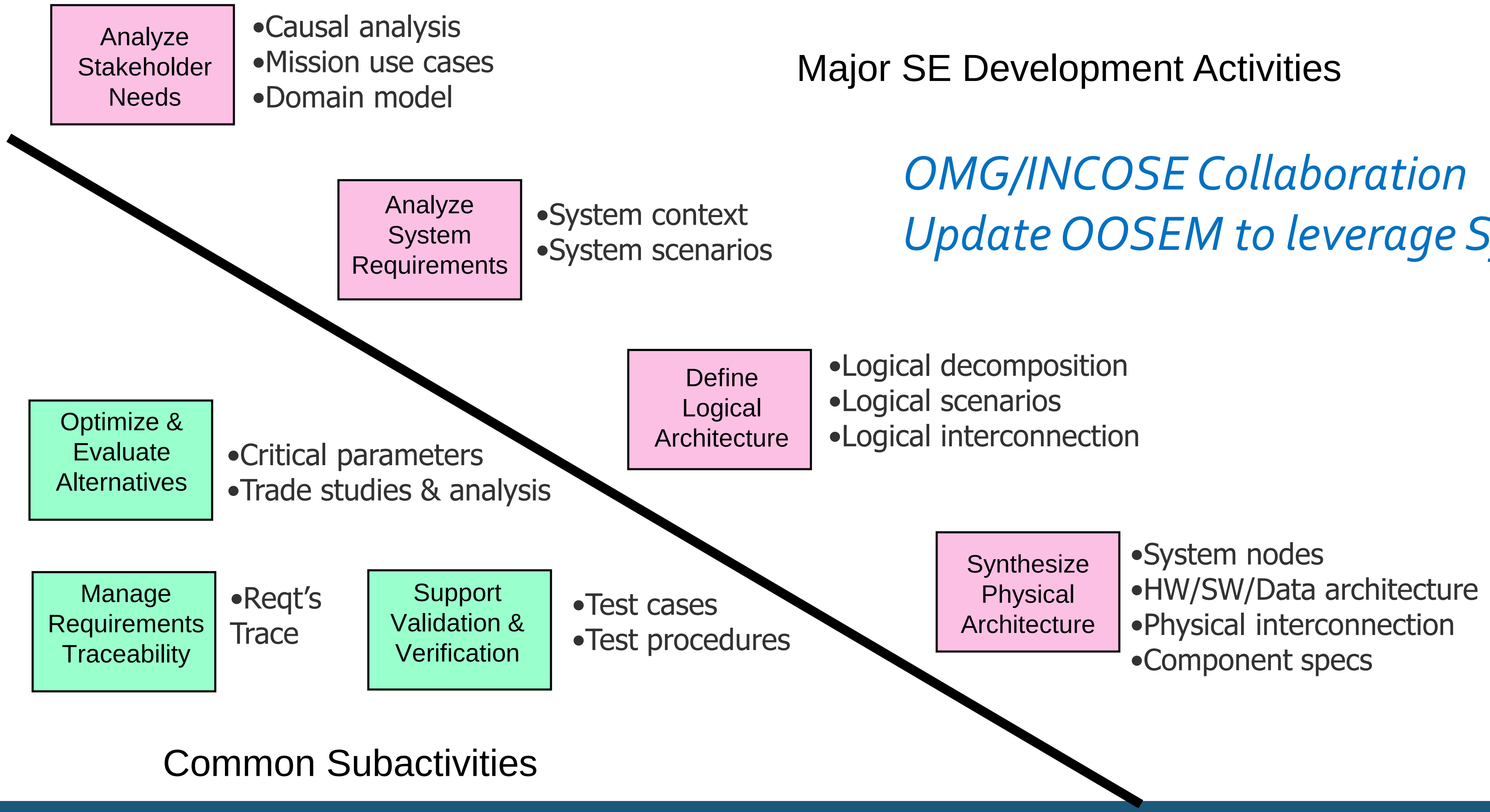
MBSE Process
Verify Before Build

- Increments can be use case driven (e.g., functional threads)
- Leverage reuse throughout



Facilitate Rigor and Agility

OOSEM Specify & Design Activity & Selected Modeling Artifacts



Major SE Development Activities

Analyze Stakeholder Needs
- Causal analysis
- Mission use cases
- Domain model

Analyze System Requirements
- System context
- System scenarios

Define Logical Architecture
- Logical decomposition
- Logical scenarios
- Logical interconnection

Synthesize Physical Architecture
- System nodes
- HW/SW/Data architecture
- Physical interconnection
- Component specs

Common Subactivities

Optimize & Evaluate Alternatives
- Critical parameters
- Trade studies & analysis

Manage Requirements Traceability
- Reqt's Trace

Support Validation & Verification
- Test cases
- Test procedures

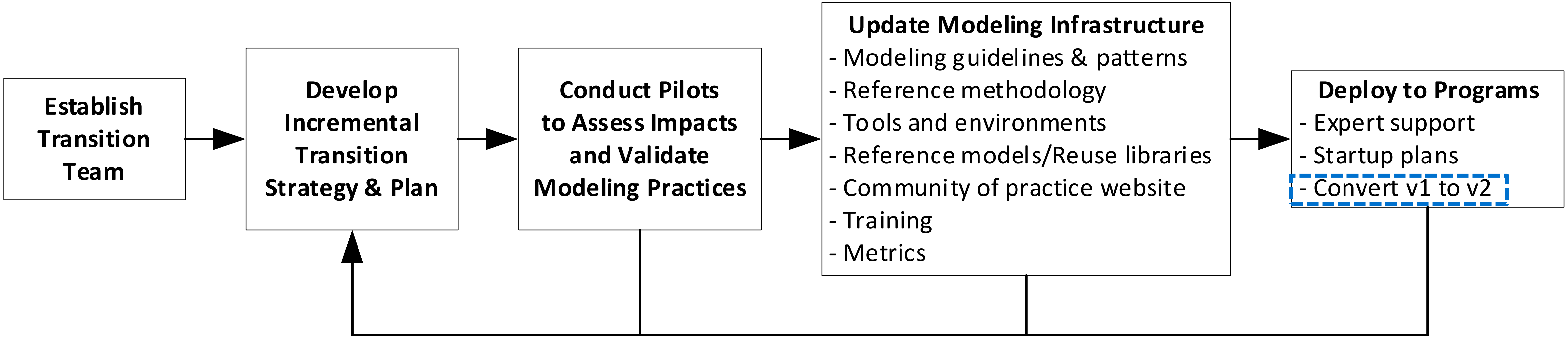
OMG/INCOSE Collaboration
Update OOSEM to leverage SysML v2

SysML v1 to SysML v2 Transition

SysML v1 to SysML v2 Transition Project

Sponsored by Director of Digital Engineering, Modeling and Simulation within the DoD Office under Secretary of Defense for Research and Engineering (OUSD(R&E) and the Naval Air Systems Command.

Transition guidance available at:
https://www.omgwiki.org/MBSE/doku.php?id=mbse:sysml_v2_transition


Establish Transition Team

Develop Incremental Transition Strategy & Plan

Conduct Pilots to Assess Impacts and Validate Modeling Practices

Update Modeling Infrastructure
- Modeling guidelines & patterns
- Reference methodology
- Tools and environments
- Reference models/Reuse libraries
- Community of practice website
- Training
- Metrics

Deploy to Programs
- Expert support
- Startup plans
- Convert v1 to v2

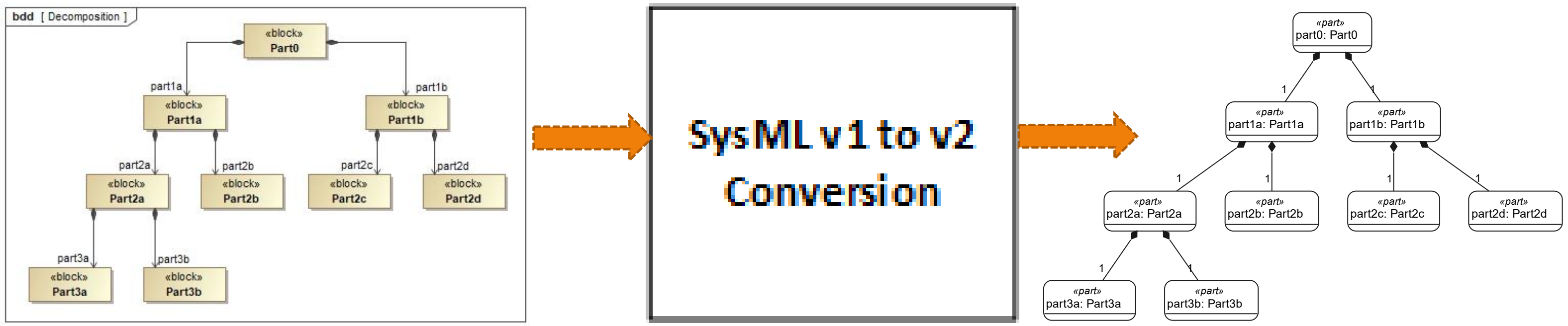
Implement Strategy and Plan Incrementally

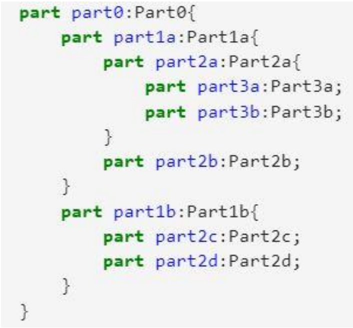
SysML v1 to SysML v2 Model Conversion

- Perform conversion incrementally
  - ○ Select portion of model to convert
  - ○ Pre-process as required
  - ○ Perform transformation
  - ○ Validate results
  - ○ Reorganize and refactor

SysML v1 Model

SysML v2 Model
Graphical & Textual Notation





```
part part0:Part0{
    part part1a:Part1a{
        part part2a:Part2a{
            part part3a:Part3a;
            part part3b:Part3b;
        }
        part part2b:Part2b;
    }
    part part1b:Part1b{
        part part2c:Part2c;
        part part2d:Part2d;
    }
}
```

Wrap-up

Summary

- SysML v2 is addressing SysML v1limitations to improve MBSE adoption and effectiveness
  - New metamodel with both graphical and textual syntax and standardized API to access the model
  - More precise, expressive, usable, interoperable, and extensible than SysML v1
  - Consistent definition and usage pattern enables reuse, usability, and automation
- Progress/Plans
  - SysML v2 specifications submitted to OMG for final adoption in 2025
  - Continue to evolve SysML v2 specification, modeling practices, and domain specific extensions through the OMG Systems Modeling Community (SMC) in collaboration with INCOSE
- Organizations and practitioners should initiate their SysML v2 transition planning
  - Refer to SysML v2 Transition Wiki at https://www.omgwiki.org/MBSE/doku.php?id=mbse:sysml_v2_transition

Thank You!!