Object Management Group's System Modeling Laungage (SysML) v2 Preview

Presented to the INCOSE Northstar Chapter

November 13, 2024

Systems Modeling

An OMG Managed Community

Brian R Larson

SysML (v1) brought me to INCOSE

- I attended the Systems Modeling Language (SysML) tutorial sponsored by our North Star chapter in 2008.
- My employer paid the non-member rate, so I became a member!
- The tutorial was presented by Sanford (Sandy) Friedenthal, one of the principal creators of SysML.
- By this time I had been invited to join the Architectural Analysis and Design Language standard committee (SAE International AS-2C).

AADL was better, by SysML won the market

OMG Systems Modeling
An OMG Managed Community

- I was unimpressed with SysML at the time:
  - it only had a graphical language
  - everything was a block
  - no semantics (lines and boxes)
- AADL came from avionics system design:
  - both textual (primary) and graphical representations
  - components representing software and hardware
  - distinguished component types (everything externally visible) from component implementations
  - informal semantics

SysML won because it was a profile of UML

- OMG's Unified Modeling Language had become the standard way of designing enterprise IT systems
- Because SysML was a profile of UML, all the UML tool vendors could field products almost immediately.
- AADL had the Open-Source AADL Tool Environment (OSATE) from the Software Engineering Institute and AADL Inspector from Ellidiss.

OMG created the Systems Modeling Community (SMC) for SysML v2

OMG Systems Modeling
An OMG Managed Community

- SMC has many working groups
- The Real-Time Embedded Safety-Critical Systems (RTESCS) Working Group met for the first time in March 2025 at Reston, VA.
- RTESCS Charter: *"Develop domain libraries w/ KerML & SysMLv2 to support the precise modeling of Real-Time Embedded Safety-Critical Systems. Integrate capabilities from domain-specific models like SAE AADL, OpenGroup FACE, OMG MARTE, & AutoSAR"*
- However, the WG is overwhelmingly member of the AADL standard committee, so most everyone calls it the AADL WG.
- We are trying to provide standard libraries and methodology for that portion of SysML v2 models to be high-assurance software and electronics.

3rd SysML Presentation in as many months

- September, Medtronic described their use of SysML for Model-Based Systems Engineering.
- October, Ahmed Hamza presented Siemens' vision of Model-Based Cybertronic Systems Engineering.
- By "Cybertronic" Ahmed means: "The engineering discipline of hw/sw functional co-architecture to realize new s/w defined, semiconductor enabled systems."

Some slides from Ahmed Hamza's presentation in October

The Current Status of System Engineering

SysML v1 – Cameo **vs SysML v2**

SIEMENS

SysML is not living up to the hype

Four critiques of SysML:

1. The learning curve is too steep
2. The mechanics are cumbersome
3. The barriers to organizational adoption are high
4. It is rarely used past the concept stage

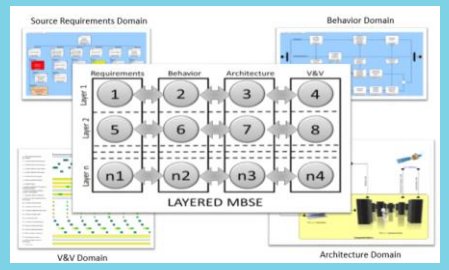
(26) Post | LinkedIn

SIEMENS

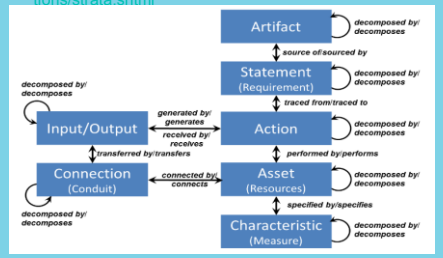
Current Status: Systems Engineering is not yet Model-Based enough

1

Multi-Domain
Diagram Centric



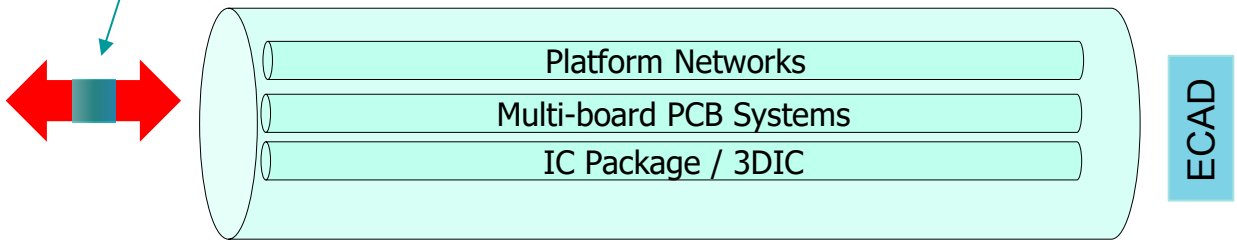
Source: http://www.vitechcorp.com/solutions/strata.shtml



Source: http://www.lifecyclemodeling.org/spec/LML_Specification_1_0.pdf

Integration Gaps!

Implementation Domains
Data Centric



Non-Deterministic database

Deterministic database

SIEMENS

Lack of Standard profile

2

SysML V1

All profiles and libraries bundled with CAMEO products are considered standard/system resources, non-modifiable, and essential for the proper performance of the tool.

We highly recommend not to modify our provided standard profiles and libraries as it could cause problems with version updates, plugins, core malfunctions, and model corruptions.

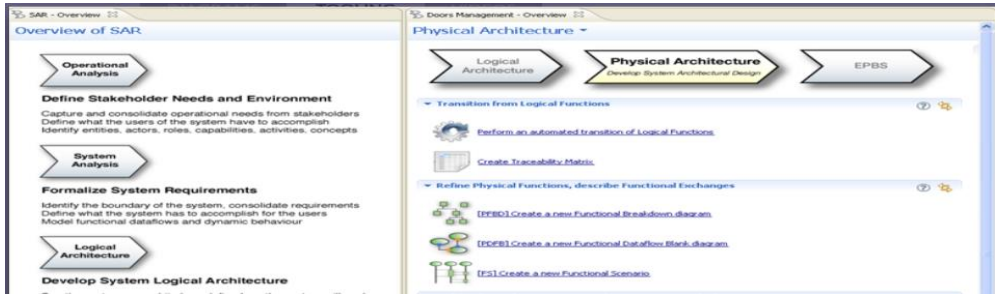
Intentional or unintentional attempt to modify profiles when:

- Opening profiles as projects.
- Using profiles in the read-write mode.
- Importing profiles into a project.
- Merging projects, and several other occasions.

AE SysML v2

Agree on a standard notation: In a team environment, it is important that others can understand your diagrams without much explanation. Choosing a standard notation enables others to quickly comprehend your diagrams without ambiguity.

The activity browser provides methodological access to all key activities of AE. It is the main entry point to a model and is both meant for beginners and power users.



SIEMENS

Lack of Methodology

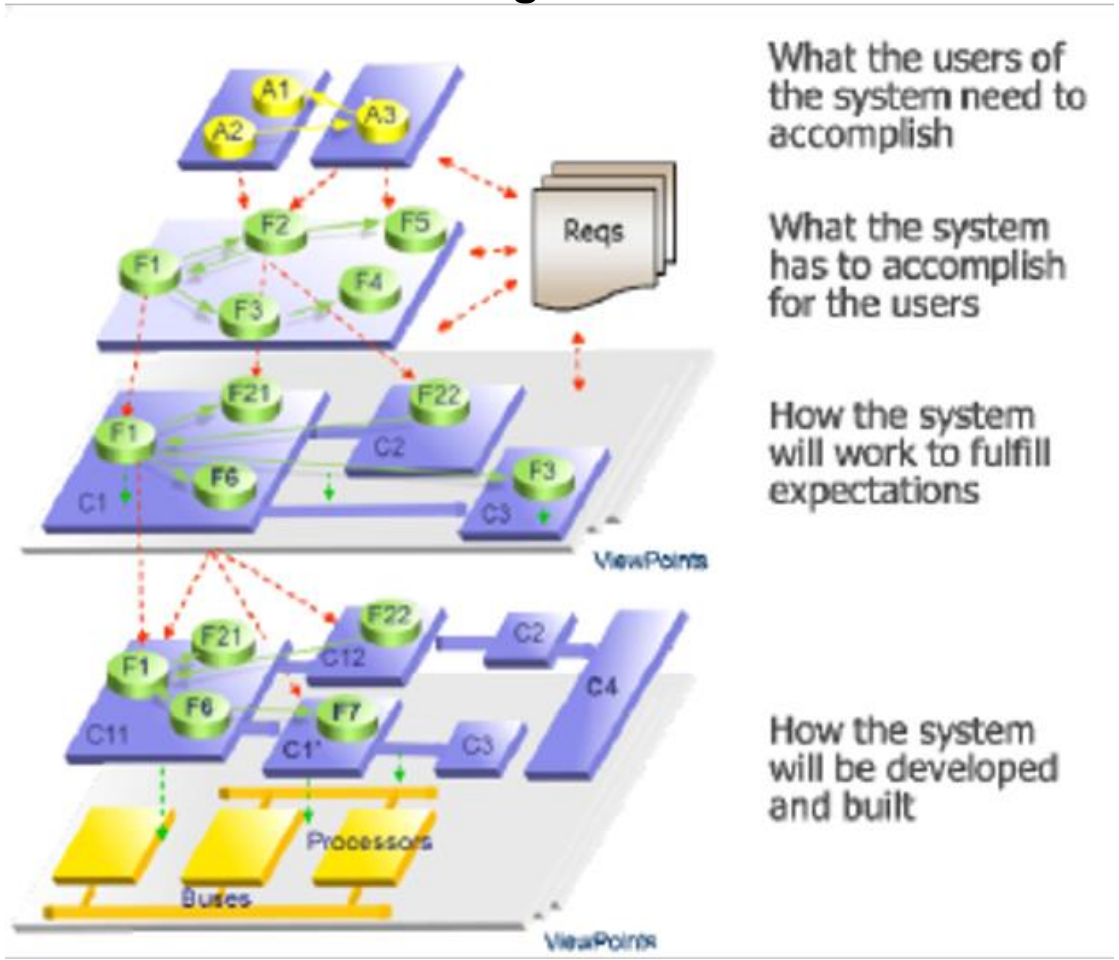
3

SysML V1

AE SysML v2

Arcadia is a tooled method devoted to systems and architecture engineering, supported by the Architecture Explorer modeling tool.

Methodological Guidance



SIEMENS

Use Abstraction and System to Sub-system Decomposition

4

SysML V1

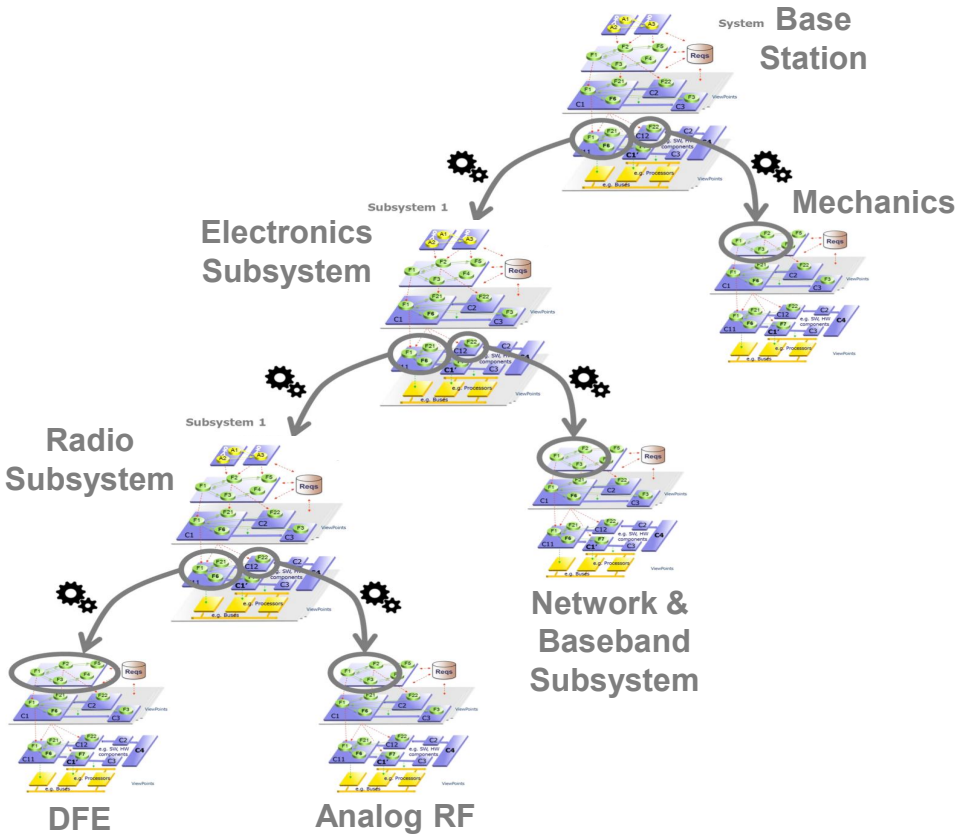
AE SysML v2

One of the most effective ways to manage system design complexity is to use abstraction and decomposition.

Abstraction is the process of hiding the details of a system and exposing only the essential characteristics and functionality.

Decomposition is the process of breaking down a system into smaller and simpler components that can be designed, implemented, tested, and maintained independently.

By using abstraction and decomposition, you can reduce the complexity of the system, increase the modularity and reusability of the components, and isolate the impact of changes.

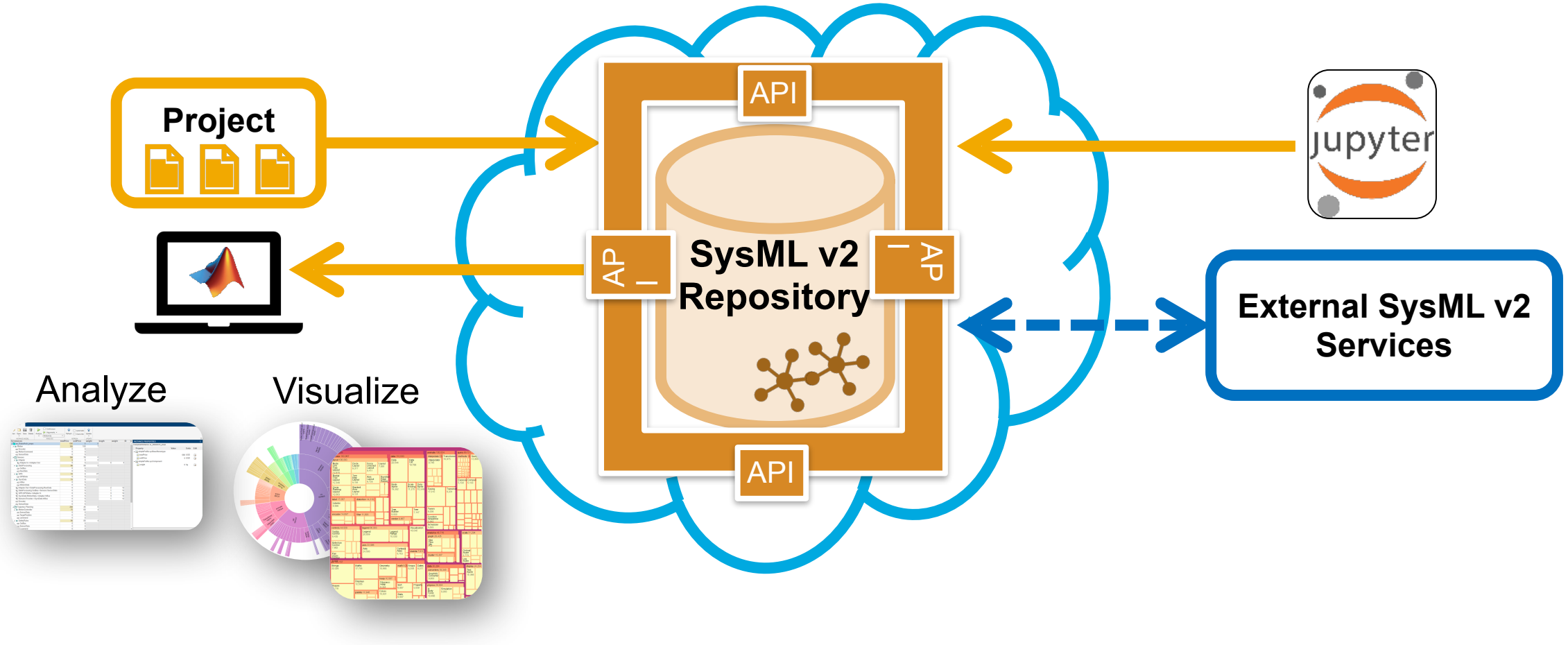


Source: https://eclipse.dev/capella/arcadia.html

SIEMENS

MathWorks Vision of Tool Interoperability

Our Vision for Interoperability



SysML v2 Overview

October 14, 2024

Sanford Friedenthal
safriedenthal@gmail.com

Copyright © 2019-2024 by Sanford Friedenthal

SysML v2 Status

Systems Modeling Language™ (SysML®)

Supports the specification, analysis, design, and verification and validation of complex systems that may include hardware, software, information, processes, personnel, and facilities

- SysML has evolved to address user and vendor needs
  - v1.0 adopted in 2006; v1.7 adopted 2022
- SysML v1 has facilitated awareness and adoption of MBSE
- Much has been learned from using SysML v1 for MBSE
- SysML v2 is the next generation systems modeling language intended to address some of the limitations of SysML v1

SysML v2 Status

- SysML v2 was developed by the SysML v2 Submission Team (SST) in response to the SysML v2 RFP issued by the OMG in December, 2017
- SysML v2 beta specifications (i.e., KerML, SysML v2, Systems Modeling API & Services) were approved by the OMG and are in the finalization phase
  - Finalization task force responds to issues raised by vendors as they develop their implementations
- Submit specifications for final adoption in 2025

SysML v2 Examples
Open-Source Pilot Implementation

- Examples of the SysML v2 textual syntax were created using the open-source reference implementation that was developed as part of the SysML v2 submission development effort
- The graphical views of the SysML v2 model were created using a drawing tool (draw.io) or the prototype visualization tool integrated with the reference implementation, based on an open-source application called Plant UML
- The quality and conformance of the graphical visualization is limited but will be substantially improved when commercial tools become available

SysML v2 Overview & Comparison with SysML v1

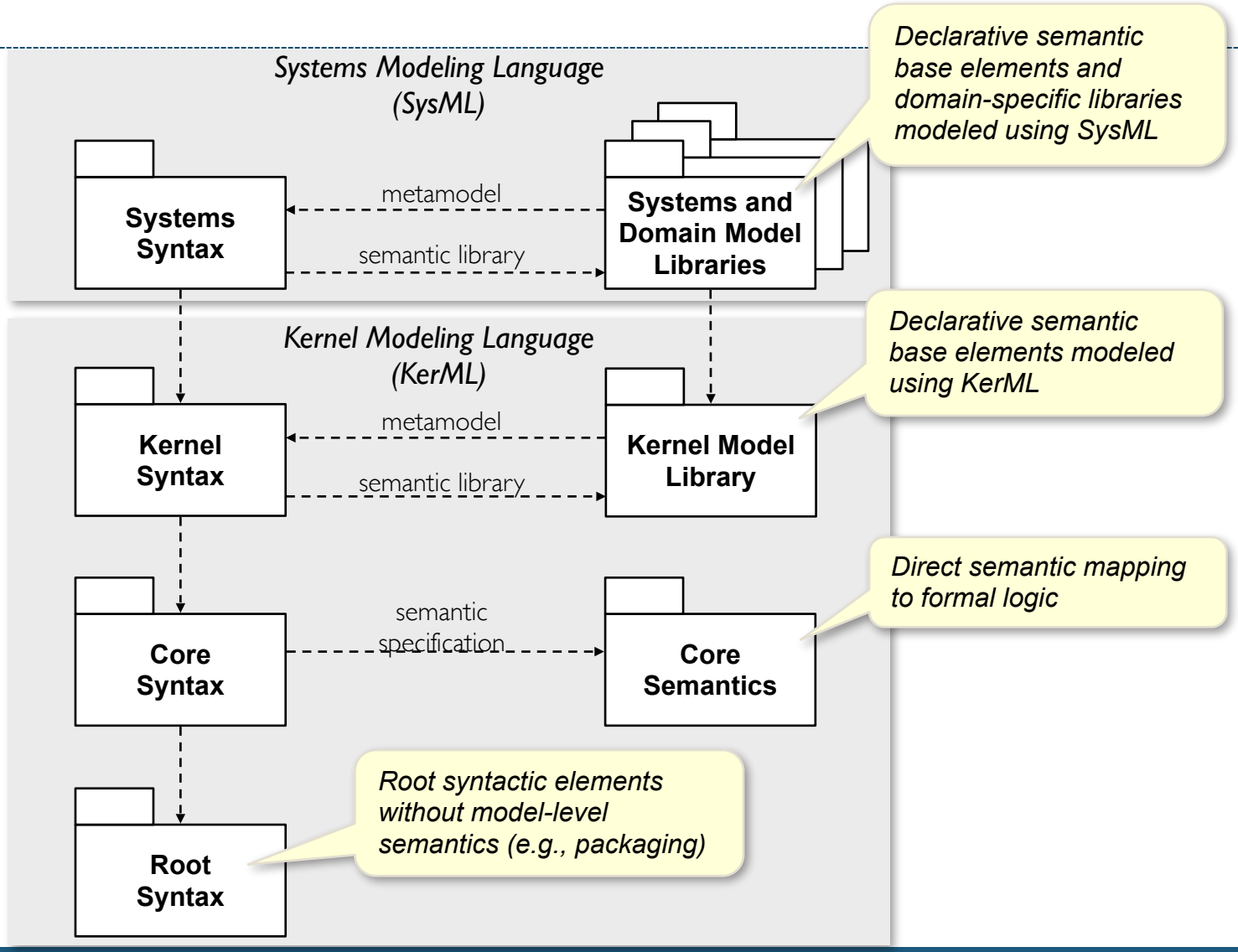
SysML v2 Objectives

- **Increase adoption and effectiveness of MBSE with SysML 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

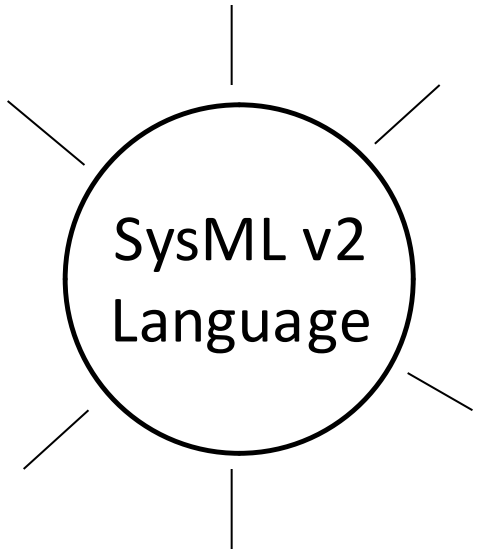


SysML v2 Language Capabilities

Requirements

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

Structure
- decomposition
- interconnection
- classification



Analysis
- analysis cases
- expression language

Verification
- verification cases

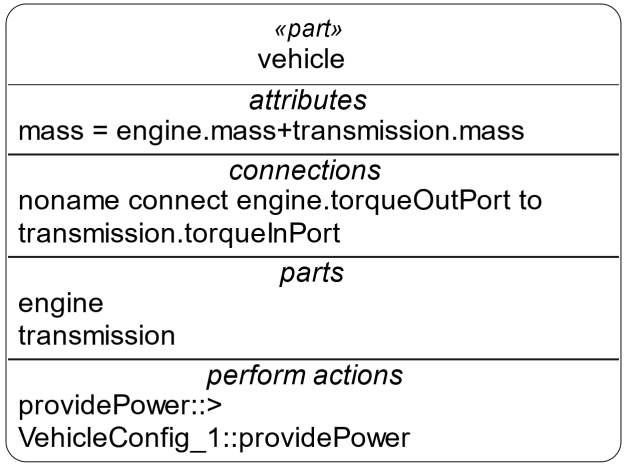
View & Viewpoint

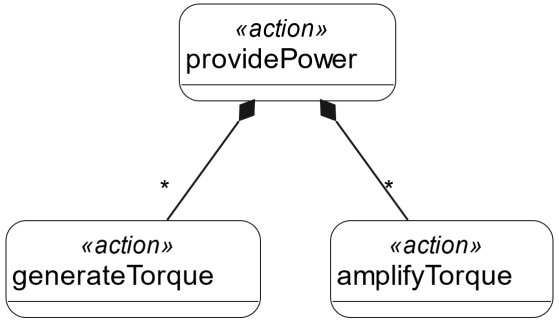
SysML v2 Reuse Pattern

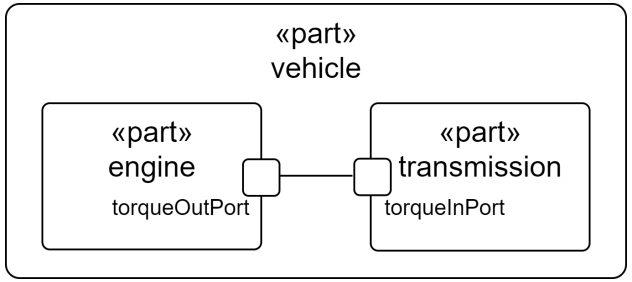
- Definition and usage
  - A definition element defines an element such as a part, action, or requirement
  - A usage element is a usage of a definition element in a particular context
  - Pattern is applied consistently throughout the language

Simple Vehicle Model
SysML v2 Textual and Graphical Syntax

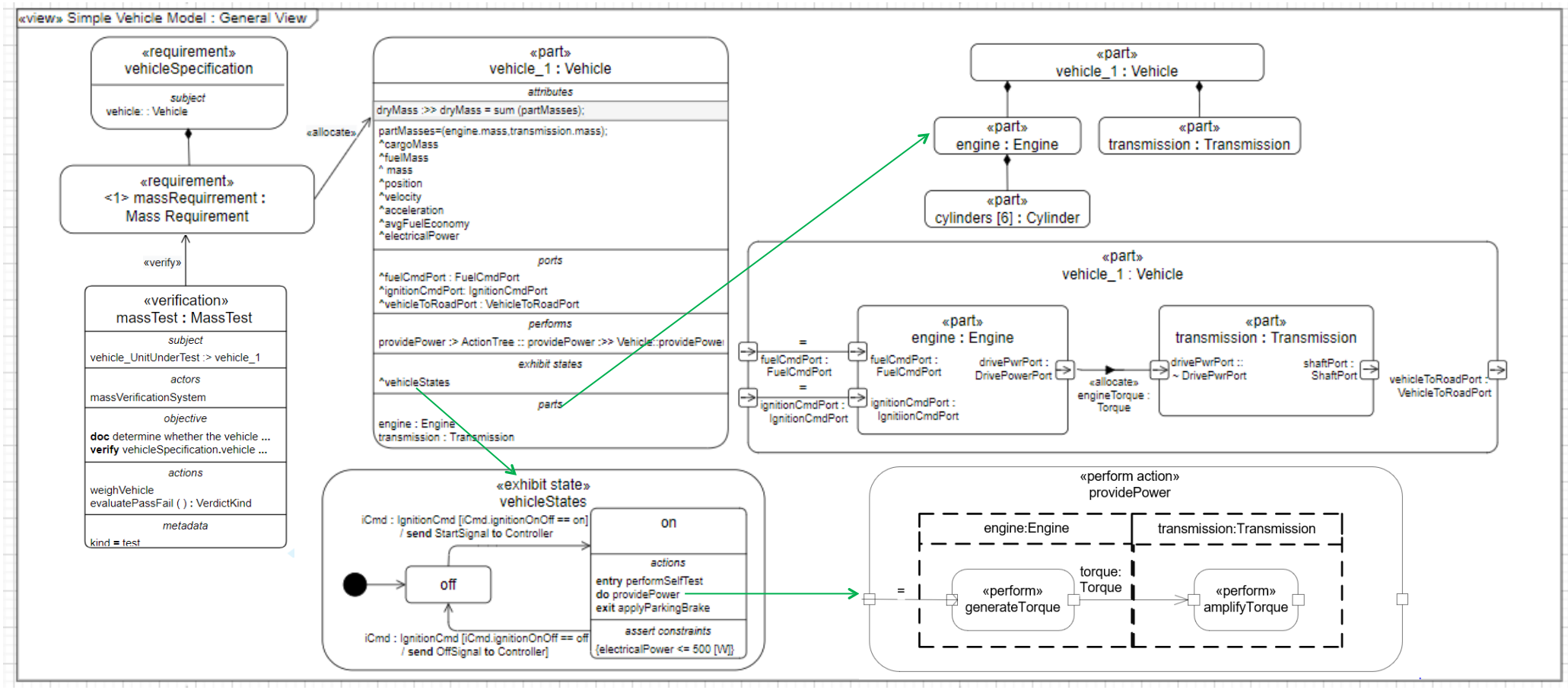
```
part vehicle{
    attribute mass = engine.mass+transmission.mass;
    perform providePower;
    part engine{
        attribute mass;
        port torqueOutPort;
        perform providePower.generateTorque;
    }
    part transmission{
        attribute mass;
        port torqueInPort;
        perform providePower.amplifyTorque;
    }
    connect engine.torqueOutPort to transmission.torqueInPort;
}
action providePower{
    action generateTorque;
    action amplifyTorque;
}
```



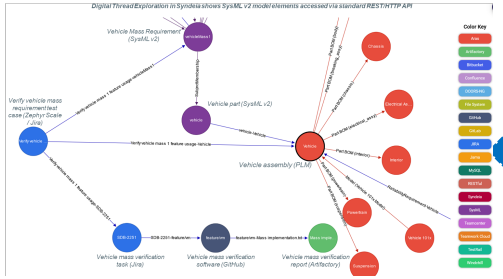




Simple Vehicle Model

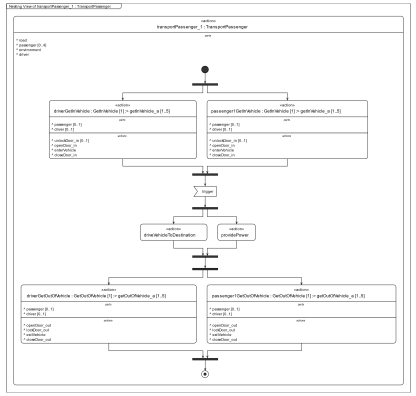


Connecting SysML v2 through the standard API



CM of the Digital Thread

Source: Syndeia with SysML v2



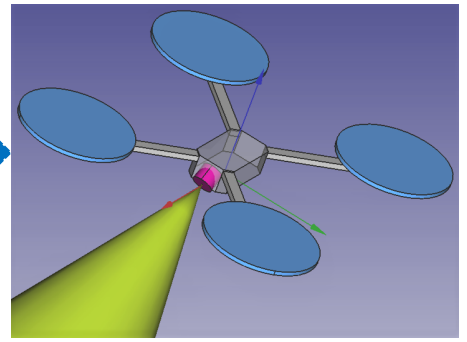
Graph Visualization

Source: Tom Sawyer with SysML v2

Systems Modeling API

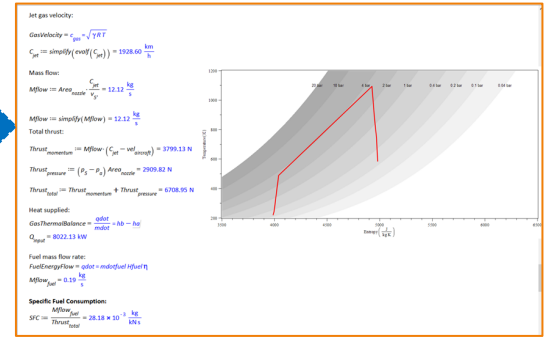
SysML v2

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



CAD/CAD Viewer

Source: FreeCAD with SysML v2



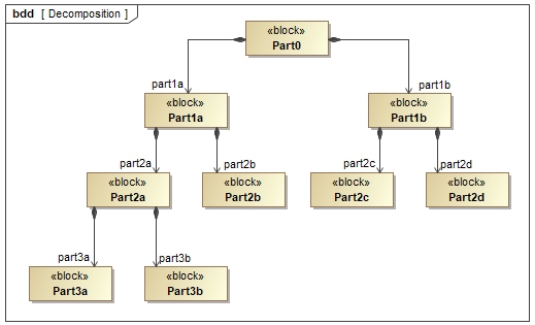
Analysis Solver

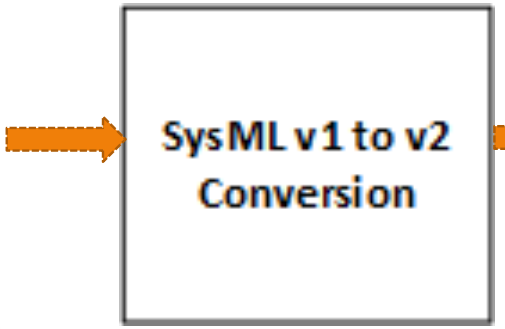
Source: Maple with SysML v2

SysML v1 to SysML v2 Model Conversion

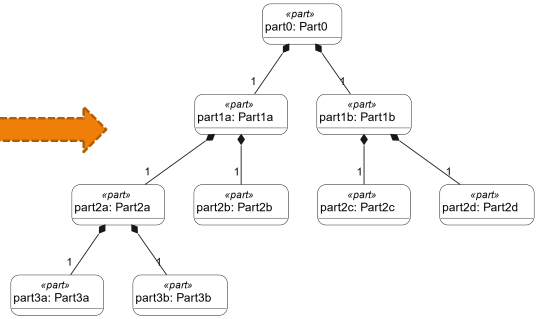
- Perform conversion incrementally
  - Select portion of model to convert
  - Pre-process as required
  - Perform transformation
  - Validate results
  - Reorganize and refactor

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;
    }
}
```

Summary

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
  - OMG approved SysML v2 beta specifications with specification to be submitted in 2025 for final adoption
  - Continue to evolve SysML v2 modeling practices, specifications, and domain specific extensions
- Organizations and practitioners should initiate their SysML v2 transition planning

SST Public Repositories
Current Release: 2024-08

- Monthly release repository
  - 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)
  - Pilot implementation
    - Installer for Jupyter tooling
    - Installation site for Eclipse plug-in
  - Web access to prototype repository via SysML v2 API
  - Web access to Tom Sawyer visualization tooling
- Open-source repositories
  - https://github.com/Systems-Modeling
- Google group for comments and questions
  - https://groups.google.com/g/SysML-v2-Release
    (to request membership, provide name, affiliation and interest)

Rushing to Release SysML v2.0 in March 2025

- Mandated by OMG charter for SysML v2
- Only the most significant, "blocker" issues to be resolved by then
- Already discussing v2.1
- SysML v1 now up to v1.4, so this should not be a surprise
- Get user experience; include in future releases
- Attempt to be backward compatible

Formal (Mathematical) Semantics (Meaning)

- Core KerML semantics "direct mapping to formal logic"
- Kernel KerML elements built upon Core elements
- SysML elements built upon Kernel elements
- So the meaning of SysML v2 models are defined with mathematical precision, right?

KerML Core Semantics Define Ontology

- Ontology in philosophy is the study of being (what is).
- Ontology in IT is groups of things (classes/categories) and relationships between them.
- KerML Core Semantics encode ontology of Core elements (types, classes, features) in sets of tuples.
- These semantics (in the current KerML specification) are incorrect, where not incomprehensible.
- Confirmed with Ed Seidewitz (co-char of SMC with Sandy Friedenthal) at the June meeting of SMC in Las Vegas, who suggested I write it up for the Semantics WG.
- Conrad Bock (principal author of the semantics) agreed with my suggestions to fix the math, and thanked me for close scrutiny.

"Supplemental" Semantics from Classical Logic and Set Theory

- When I couldn't understand the ontological semantics of KerML Core elements, I started to define their semantics in terms of classical (first-order) logic and set theory.
- The "supplemental" semantics define what the elements really are, not just their classes and classification predicates.
- In September, at the SMC meeting in Chicago, Conrad Bock agreed to work with me to ensure that the ontological semantics match the supplemental semantics on questions for which they both apply (subtyping, feature redefinition, feature chaining etc.).

Informal Time

- Timing behavior of KerML and SysML is ad hoc.
- Defined timing behavior with labels, and object-oriented type checking.
- Did the best they could to implement intuitive understanding of time with what they had.
- Many thorny issues confronted by the Semantics WG arise from informal timing semantics.

Formalizing Time with Temporal Logic

- Adapting the temporal logic created to prove correctness of AADL models having BLESS behavior specifications and state machines.
- Infusing KerML Core semantics with temporal logic.
- Adding temporal logic formulas to elements defined in standard libraries provides mathematical exactness, eliminates all semantic conundrums, and allows dramatic simplification (elimination of stuff) from standard libraries.

Supplemental Semantics will endow SysML models with mathematical exactness

- Only then will analysis/verification/validation/simulation/code generation interoperability apply to the same thing.
- Only then will SysML v2 models be Authoritative Source of Truth.
- ***Without supplemental semantics, SysML v2 will succumb to the same limitations as SysML v1.***

Interested?

- Supplemental Semantics document about half done.
- Standard libraries will need to be revised.
- Introductory slide deck has 200+ slides.
- Forming small study group.
- If you'd like to see the document (when it's ready) or the slide show, or join the study group, contact me: brl@multitude.net

Questions?

- SysIDE and Pilot Implementation demonstrations and semantic metadata annotations of SysML models if time permits.