Agenda

- **Part 1 – S. Friedenthal***
  - Background
  - Comparing SysML v2 with SysML v1
  - Considerations for transitioning from SysML v1 to SysML v2
  - Summary

- **Part 2 – E. Seidewitz**
  - SysML v2 Submission Team
  - Key language design principles
  - Rational and trade offs for some key language design decisions
  - Conclusion

---

***Reference:*** Friedenthal S., Seidewitz E., "SysML v2: Highlighting the Differences with SysML v1, Project Performance International (PPI)". Systems Engineering Newsletter, PPI SyEN 123, April 2023

[https://www.ppi-int.com/systems-engineering-newsjournal/ppi-syen-123/](https://www.ppi-int.com/systems-engineering-newsjournal/ppi-syen-123/)
PART 1
WHERE WE ARE
SANDY FRIEDENTHAL
Examples of the SysML v2 textual syntax were created using the open-source pilot 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 prototype visualization tool integrated with the pilot implementation, based on an open-source application called Plant UML. Note: Some SysML v2 views created in draw.io application.

The quality of the graphical visualization is limited but will be substantially improved when commercial tools become available.
BACKGROUND
The Future of Systems Engineering is Model-Based

- Part of the digital transformation
- Full life cycle from SoS to component level
- Agile system development including automated workflow and CM of the digital thread
- Model patterns and reuse

- Facilitates
  - managing complexity & risk
  - more rapidly respond to change
  - reuse and design evolution
  - reasoning about & analyzing systems
  - shared stakeholder understanding
  - automated documentation & reporting

Source: INCOSE SE Vision 2035
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

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>February 2023</td>
<td>Submitted Alpha Specifications</td>
</tr>
<tr>
<td>March 2023</td>
<td>Formed Finalization Task Forces</td>
</tr>
<tr>
<td>June 2023</td>
<td>Adopted by OMG Board of Directors</td>
</tr>
<tr>
<td></td>
<td>Published Beta Specifications</td>
</tr>
<tr>
<td>December 1, 2023</td>
<td>Public Comment Deadline</td>
</tr>
<tr>
<td>March 2024</td>
<td>Deliver Finalized Specifications</td>
</tr>
<tr>
<td></td>
<td>Establish Revision Task Forces</td>
</tr>
<tr>
<td>Mid 2024</td>
<td>Publish Formal Specifications</td>
</tr>
</tbody>
</table>
COMPARING SYSML V2
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 Capabilities

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

Structure
- decomposition
- interconnection
- classification

Requirements

Analysis
- analysis cases
- expression language

Verification
- verification cases

View & Viewpoint
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;
}
Definition and Usage
SysML v2 vs. SysML v1

- Reuse concept to define an element once and use it in different contexts

- SysML v1 informally introduces the concept of definition and usage (e.g., block and part property)
  - It is applied inconsistently across the language (e.g., blocks, activities, requirements)

- Definition and usage elements are formally part of SysML v2
  - Applies to virtually all elements (e.g., attributes, parts, ports, connections, actions, states, requirements, constraint, cases, views, …)
  - Supports consistent pattern of decomposition and specialization

- Benefits
  - Enables effective reuse
  - Facilitates learning and using the language
  - Enables automation
### Terminology (partial)
#### SysML v2 vs. SysML v1

<table>
<thead>
<tr>
<th>SysML v2</th>
<th>SysML v1</th>
</tr>
</thead>
<tbody>
<tr>
<td>part / part def</td>
<td>part property / block</td>
</tr>
<tr>
<td>attribute / attribute def</td>
<td>value property / value type</td>
</tr>
<tr>
<td>port / port def</td>
<td>proxy port / interface block</td>
</tr>
<tr>
<td>action / action def</td>
<td>action / activity</td>
</tr>
<tr>
<td>state / state def</td>
<td>state / state machine</td>
</tr>
<tr>
<td>constraint / constraint def</td>
<td>constraint property / constraint block</td>
</tr>
<tr>
<td>connection / connection def</td>
<td>connector / association block</td>
</tr>
<tr>
<td>requirement / requirement def</td>
<td>requirement</td>
</tr>
<tr>
<td>view / view def</td>
<td>view</td>
</tr>
</tbody>
</table>

*SysML v2 applies a consistent pattern of definition and usage*
SysML v1 and v2
Vehicle Block vs Part Decomposition

SysML v1
Block Decomposition

SysML v2
Part Decomposition
SysML v2 Requirement

- Builds on SysML v1 concept of a property-based requirement
- A constraint definition that a valid design solution must satisfy that can include:
  - Identifier
  - Shall statement
  - Constraint expression that can be evaluated to true or false
  - Attributes of the constraint expressions
  - Assumed constraint expression must be true for the requirement to be applicable

A SysML v2 Requirement Can be Evaluated by a Solver as Pass or Fail
SysML v1 Instances vs. SysML v2 Individuals and Snapshots

SysML v2 distinguishes the concept of an individual from a snapshot of an individual at a point in its lifetime.
SysML v2 Alias and Short Name

```
«part def»
SportUtilityVehicle

alias

«alias»
SUV for SportUtilityVehicle

short name

«part»
<1.1> suv:SUV
```
Language Extension
SysML v2 vs SysML v1

Library extension mechanism in SysML v2 can automatically combine the capability of specialization with stereotypes

SysML v1

SysML v2
Simple Vehicle Model
Connecting SysML v2 through the standard API

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

CM of the Digital Thread
Source: Syndeia with SysML v2

Graph Visualization
Source: Tom Sawyer with SysML v2

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 application of definition and usage pattern
  - More consistent terminology
  - Ability to decompose parts, actions,
  - More flexible model organization with package filters

- More precise
  - Textual syntax and expression language
  - Formal semantic grounding
  - Requirements as constraints

- More expressive
  - Variant modeling
  - Analysis case
  - 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
TRANSITIONING TO
SYSML V2
SysML v1 to v2 Transition Planning

- Integrate transition planning with existing MBSE/DE initiatives
  - MBSE improvement teams and community of practices
- Initiate pilots using the Jupyter environment to begin impact assessment
- Initiate tool vendor discussions on roadmap
- Prepare incremental plans
  - MBSE practices
  - Tool infrastructure
  - Training
  - Metrics
  - Reference models and reuse repositories
  - MBSE Community of Practice website
  - Criteria for project deployment

*Transition Guidance being developed by DoD office of DE, Modeling & Simulation*
SysML v1 Model

Source: SST Track 3 Presentation
Yves Bernard, Tim Weilkiens
08 February 2022
SysML v2
Creating a Culture of Model Quality

- Transition to SysML v2 provides an opportunity to improve model quality
  - Bring more rigor to MBSE to ensure model satisfies its intended purpose
  - Applies if transforming a SysML v1 model or developing a new model

- The need for rigor
  - Consistent high quality training material
  - Practitioner and instructor certifications
  - Modeling guidelines, patterns, practices, and metrics
  - Validation suites and correct by construction
  - Review processes
  - Validated reference models
  - ...
Summary

- SysML v1 is based on UML which was originally designed as a software modeling language.
- SysML v2 was designed to address the SysML v1 limitations and 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.
  - Consistent definition and usage pattern enables reuse, usability, and automation.

- Progress/Plans:
  - Awaiting OMG approval for SysML v2 beta specifications leading to final adopted specification in 2024.
  - Will continue to evolve specification with domain specific extensions.

- Organizations should begin SysML v2 transition planning to advance their MBSE capabilities:
  - Treat as an opportunity to improve model quality.
PART 2
HOW WE GOT HERE
ED SEIDEWITZ
"The tendency of small, elegant, and successful systems to be succeeded by over-engineered, bloated systems, due to inflated expectations and overconfidence."


Fred Brooks, *The Mythical Man-Month*, Chapter 5
SYSML V2 SUBMISSION TEAM
SysML v2 Submission Team (SST)

SysML v2 Requests for Proposals
Language: December 2017
API and Services: June 2018

SST formed December 2017
Leads: Sandy Friedenthal, Ed Seidewitz

A broad team of end users, vendors, academics, and government liaisons
Grew to 200+ members from 80+ organizations

Developed submissions to both RFPs
Final submission: February 2023
### SST Participating Organizations

- Aerospace Corp
- Airbus
- ANSYS medini
- Aras
- Army Aviation & Missile Center
- Army CBRND
- BAE
- BigLever Software
- Boeing
- U.S. Army DEVCOM Armaments Center
- CalTech CTME
- CEA
- Contact Software
- Defence Science and Technology Group
- DEKonsult
- Delligatti Associates
- Draper Lab
- ESTACA
- Ford
- Fraunhofer FOKUS
- General Motors
- George Mason University
- GISE
- Georgia Tech/GTRI
- IBM
- Idaho National Laboratory
- IncQuery Labs
- Intercax
- Itemis
- Jet Propulsion Lab
- John Deere
- Kenntnis
- KTH Royal Institute of Technology
- LieberLieber
- Lightsheet Consulting
- Lincoln Lab
- Lockheed Martin
- MathWorks
- Maplesoft
- Mercury Systems
- Mgnite Inc
- MID
- MITRE
- Model Alchemy Consulting
- Model Driven Solutions
- Model Foundry
- NIST
- No Magic/Dassault Systems
- OAR
- Obeo
- OOSE
- Ostfold University College
- Phoenix Integration/ANSYS
- PTC
- Qualtech Systems, Inc (QSI)
- Raytheon
- Rolls Royce
- Saab Aeronautics
- SAF Consulting *
- SAIC
- Siemens
- Sierra Nevada Corporation
- Simula
- Space Cooperative
- Sodius Willert
- System Strategy *
- Tata Consultancy Services
- Thales
- Thematix
- Tom Sawyer
- Twingineer
- UFRPE
- University of Western Switzerland (Rosas Center)
- University of Cantabria
- University of Alabama in Huntsville
- University of Detroit Mercy
- University of Kaiserslautern / VPE
- Vera C. Rubin Observatory
- Vitech
- 88solutions

---

**Academia/Research**
- Tool Vendor
- Government Rep
- End User

**INCOSE rep *"
<table>
<thead>
<tr>
<th>Date</th>
<th>Event/Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>December 2017</td>
<td>SysML v2 RFP issued; SST formed</td>
</tr>
<tr>
<td>June 2018</td>
<td>SysML v2 API &amp; Services RFP issued</td>
</tr>
<tr>
<td>August 2019</td>
<td><strong>Internal Review</strong></td>
</tr>
<tr>
<td>August 2020</td>
<td>Initial Submission</td>
</tr>
<tr>
<td><strong>February 2021</strong></td>
<td><strong>Stakeholder Review</strong></td>
</tr>
<tr>
<td>August 2021</td>
<td>1st Revised Submission</td>
</tr>
<tr>
<td>November 2021</td>
<td>2nd Revised Submission</td>
</tr>
<tr>
<td><strong>September 2022</strong></td>
<td><strong>Specification Review (2½ days)</strong></td>
</tr>
<tr>
<td>November 2022</td>
<td>3rd Revised Submission</td>
</tr>
<tr>
<td><strong>December 2022</strong></td>
<td><strong>Established Change Board</strong></td>
</tr>
<tr>
<td>February 2023</td>
<td>Final Submission</td>
</tr>
</tbody>
</table>
KEY DESIGN PRINCIPLES
Consistency

For example, consistent pattern of definition and usage

<table>
<thead>
<tr>
<th>SysML v2</th>
<th>SysML v1</th>
</tr>
</thead>
<tbody>
<tr>
<td>part / part def</td>
<td>part property / block</td>
</tr>
<tr>
<td>attribute / attribute def</td>
<td>value property / value type</td>
</tr>
<tr>
<td>port / port def</td>
<td>proxy port / interface block</td>
</tr>
<tr>
<td>action / action def</td>
<td>action / activity</td>
</tr>
<tr>
<td>state / state def</td>
<td>state / state machine</td>
</tr>
<tr>
<td>constraint / constraint def</td>
<td>constraint property / constraint block</td>
</tr>
<tr>
<td>connection / connection def</td>
<td>connector / association block</td>
</tr>
<tr>
<td>requirement / requirement def</td>
<td>requirement</td>
</tr>
<tr>
<td>view / view def</td>
<td>view</td>
</tr>
</tbody>
</table>
Unification of Concepts

Feature typing (definition), subsetting and redefinition are all kinds of specialization.

Actions and subactions are related by feature membership, just like parts and subparts.

Object flows are the same as information flows.
Specialization in Context

Vehicle

- wheels [4]
  - lugnuts [4]
    - attributes:
      - torque: 20

- vehicle1
  - front wheels [2]
    - lugnuts: Lugnut [6]
      - attributes:
        - torque: 30

- rear wheels [2]
  - lugnuts: Lugnut [6]
    - attributes:
      - torque: 30

Features are specialized in the context their owner.

Redefinition can be different in different contexts.

Nested features can be redefined locally.
There are corresponding textual and graphical notations for each language construct.
There is a comprehensive expression language.
Textual notations can be used consistently on graphical diagrams.
DESIGN DECISIONS:
RATIONALE AND TRADE OFFS
Reified Relationships

All relationships are reified as model elements.

Owning membership

Feature membership

Feature typing

Rationale
Allows consistent graph-oriented navigation across a model

Trade off
Essentially doubles the number of model elements needed to represent a model
Compact Notation

User Conception

- «part» engine
- «satisfy»
- «requirement» engine power

Actual Meaning

- «part» engine
- «feature reference» «feature value» «subject»
- «requirement» «satisfy requirement»
- «requirement» engine power

Rationale
Maintains simple user concept, avoids use of dependency relationships without formal semantics.

Trade off
Simple surface notation hides underlying complexity, which must be navigated in a tool or repository.
Rationale
Provides underlying ontological semantics in an extensible library.

Trade off
Tools need to account for implied relationships, even if not physically added.
Conclusion

- The SST ran for over 5 years, with no significant conflict, losing no participating organizations.
- Pilot implementation was released (almost) every month from November 2018 to February 2023.
- Submitted specifications met their objectives and about 90% of the RFP requirements.
- There is already a SysML v2 user community, and there is great interest in moving to SysML v2 in the wider MBSE community.
But There are Trade-Offs

- SysML v2 is not just a simple evolution from SysML v1
  - New foundation not based on UML
  - Reified and implied relationships
  - Textual in addition to graphical notation

- SysML v2 is bigger than SysML v1
  - New functionality
  - Extensive model libraries

- SysML v2 is not easy to implement
Nevertheless... Many Implementations in Progress!

Dassault/3DS
   Cameo

IBM
   Rhapsody

PTC
   Windchill Modeler

Sparx
   Enterprise Architect

Intercax
   Syndeia

Siemens

Ansys
SST Public Repositories
Current Release: 2023-02

- 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)
THANK YOU!