2023
Annual INCOSE
international workshop
HYBRID EVENT
Torrance, CA, USA
January 28 - 31, 2023

# SysML v2 Submission Team (SST)
# SysML v2 Update

Sanford Friedenthal
SST Co-Lead
safriedenthal@gmail.com

# Presentation
# Purpose & Agenda

SST

- Purpose
  - Provide an update from the 2022 IW on the status of SysML v2
- Agenda
  - SysML v2 Overview
  - SysML v2 Progress and Plans
  - Summary


SST

# SysML v2 Overview

# SysML v2 Objectives

SST

- Increase adoption and effectiveness of MBSE 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

SST

- 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

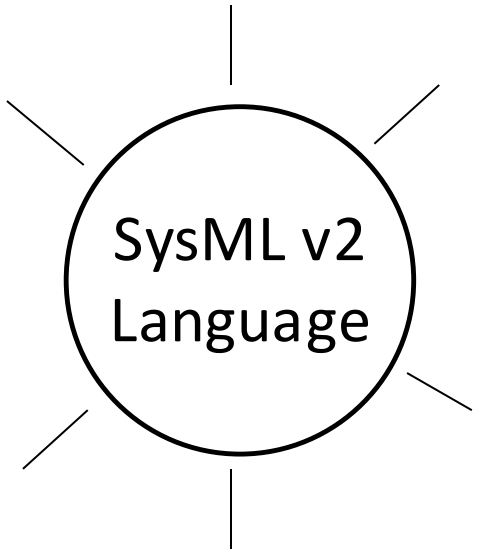
# SysML v2 Language Capabilities

SST

**Requirements**

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

**Structure**
- decomposition
- interconnection
- classification




**Analysis**
- analysis cases
- expression language

**Verification**
- verification cases

**View & Viewpoint**

# Vehicle Part Definition
## *Replaces SysML v1 Block*

SST

- The vehicle part definition is characterized by different kinds of features including
  - Attributes
  - Ports
  - Actions
  - States
  - …

| «part def»<br>Vehicle |
| --- |
| *attributes*<br>mass :> ISQ::mass = dryMass + cargoMass + fuelMass<br>dryMass :> ISQ::mass<br>cargoMass :> ISQ::mass<br>fuelMass :> ISQ::mass<br>position :> ISQ::length<br>velocity :> ISQ::speed<br>acceleration :> ISQ::acceleration<br>avgFuelEconomy :> distancePerVolume<br>electricalPower :> ISQ::power |
| *ports*<br>fuelCmdPort : FuelCmdPort<br>ignitionCmdPort : IgnitionCmdPort<br>vehicleToRoadPort : VehicleToRoadPort |
| *perform actions*<br>providePower |
| *exhibit states*<br>vehicleStates |

# Vehicle Part Definition
# Textual Syntax

SST

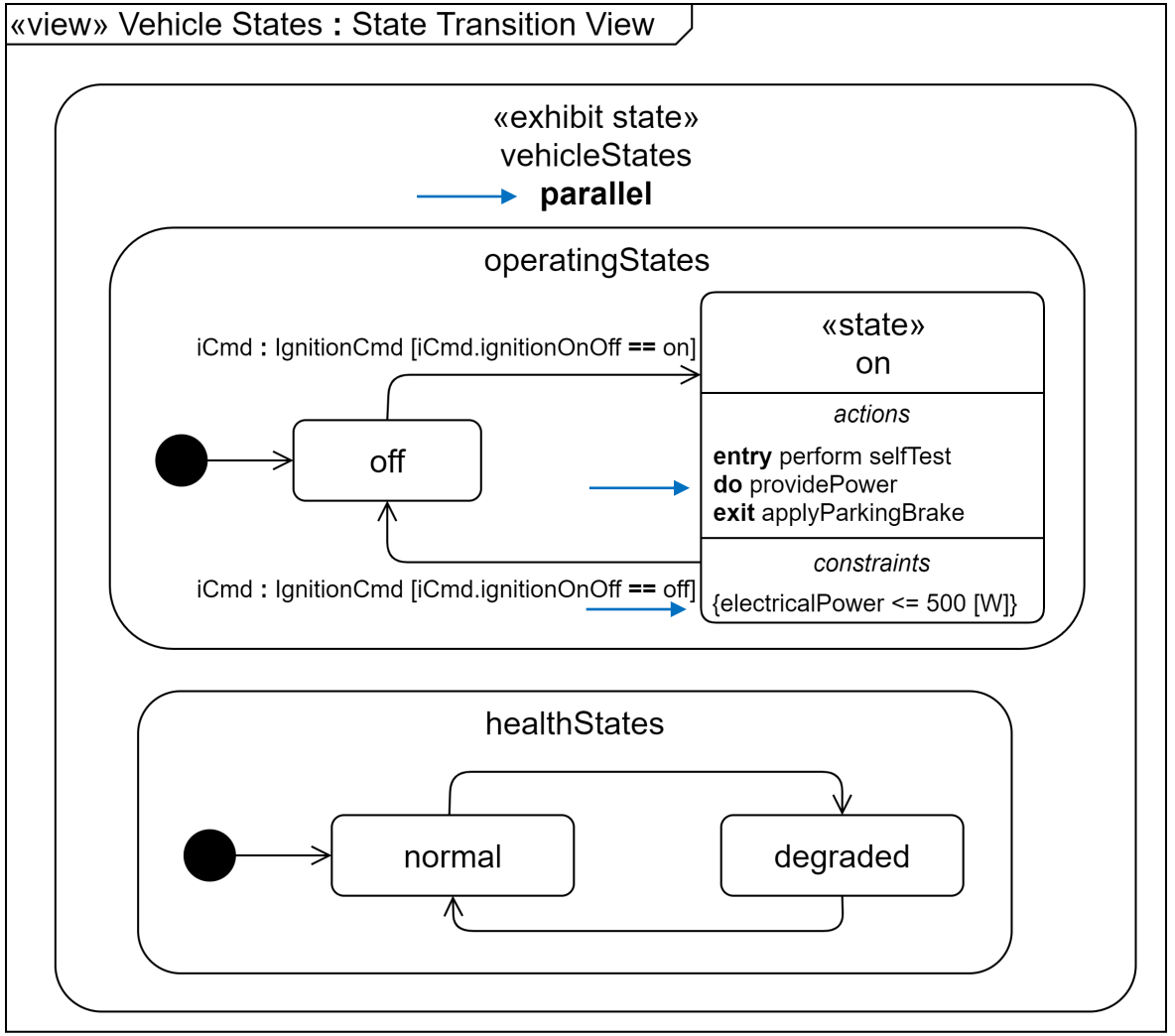
- The textual syntax reflects the same model as the graphical syntax

```
part def Vehicle{
    attribute mass :> ISQ::mass = dryMass + cargoMass + fuelMass;
    attribute dryMass:>ISQ::mass;
    attribute cargoMass:>ISQ::mass;
    attribute fuelMass:>ISQ::mass;
    attribute position:>ISQ::length;
    attribute velocity:>ISQ::speed;
    attribute acceleration:>ISQ::acceleration;
    attribute avgFuelEconomy:>distancePerVolume;
    attribute electricalPower:> ISQ::power;
    port fuelCmdPort:FuelCmdPort;
    port ignitionCmdPort:IgnitionCmdPort;
    port vehicleToRoadPort:VehicleToRoadPort;
    perform action providePower;
    exhibit state vehicleStates parallel {↔}
}
```

# Vehicle States

SST

- States are hierarchical and can include:
  - parallel states (e.g., concurrent states) and mutually exclusive states
  - entry, exit, and do actions
  - constraints



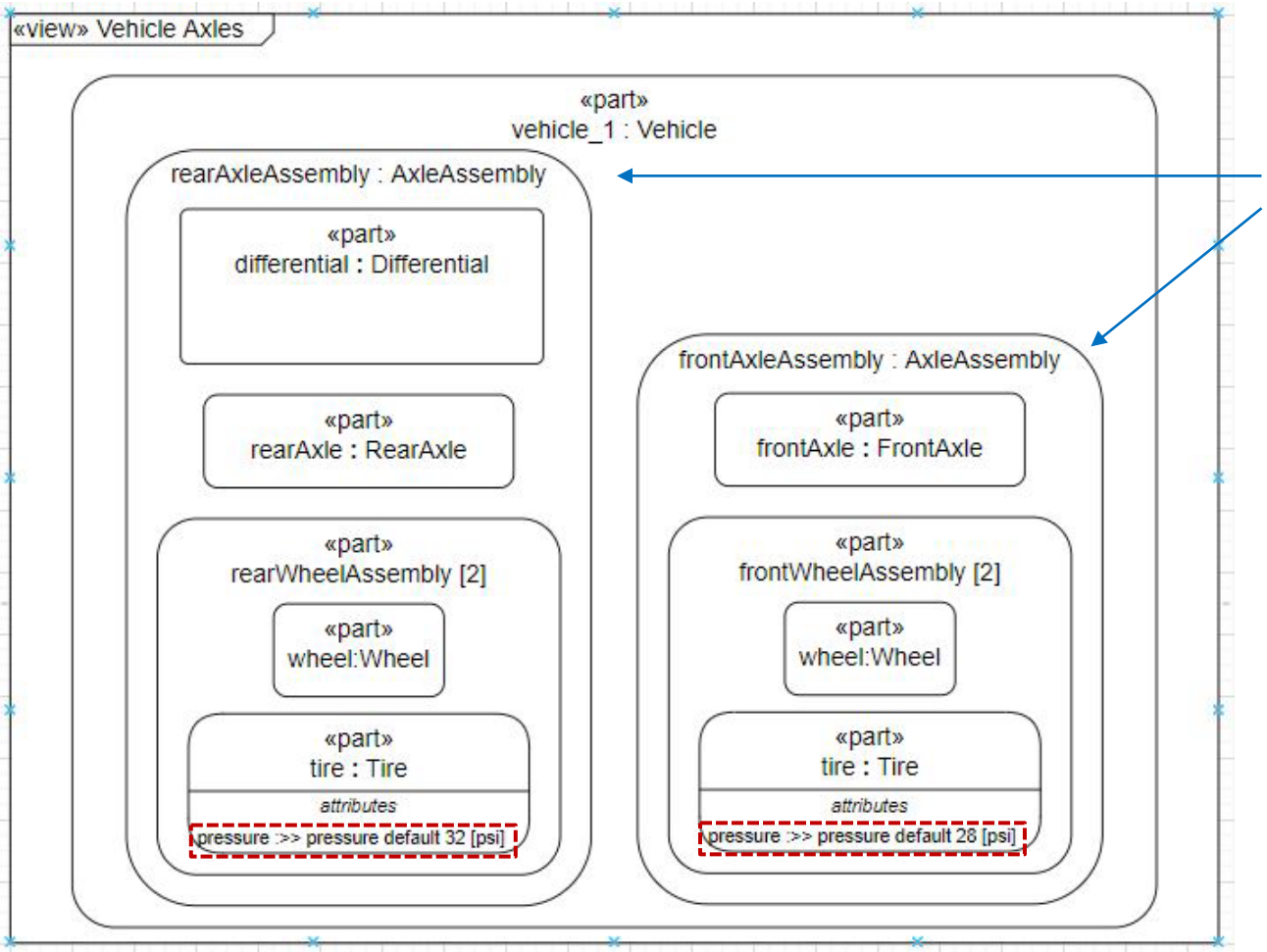
# Vehicle States
# Textual Syntax

SST

```
exhibit state vehicleStates parallel {
    state operatingStates {
        entry action initial;
        state off;
        state on {
            entry action performSelfTest;
            do providePower;
            exit action applyParkingBrake;
            constraint {electricalPower<=500[W]}
        }
        transition initial then off;
        transition off_To_on
            first off
            accept ignitionCmd:IgnitionCmd via ignitionCmdPort
                if ignitionCmd.ignitionOnOff==IgnitionOnOff::on
            then on;
        transition on_To_off
            first on
            accept ignitionCmd:IgnitionCmd via ignitionCmdPort
                if ignitionCmd.ignitionOnOff==IgnitionOnOff::off
            then off;
    }
    state healthStates {
        entry action initial;
        state normal;
        state degraded;
```

# Vehicle Usage Example
# Modifying Usages to their Context

SST




Different usages of Axle Assembly

Tire pressure is different on front and rear tires

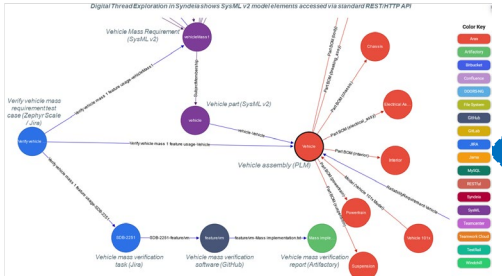
# SysML v2 Spec (Clause 7)
# SysML v2 Language Description

SST

7.2 Elements and Relationships
7.3 Annotations
7.4 Namespaces and Packages
7.5 Dependencies
7.6 Definition and Usage
7.7 Attributes
7.8 Enumerations
7.9 Occurrences
7.10 Items
7.11 Parts
7.12 Ports
7.13 Connections
7.14 Interfaces
7.15 Allocations
7.16 Actions
7.17 States
7.18 Calculations
7.19 Constraints
7.20 Requirements
7.21 Cases
7.22 Analysis Cases
7.23 Verification Cases
7.24 Use Cases
7.25 Views and Viewpoints
7.26 Metadata (incl. Language Extension)

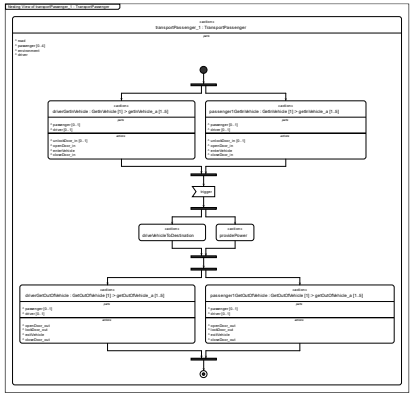
# Connecting SysML v2 through the API

SST



PLM/Version Mgmt

*Source:* Syndeia with SysML v2



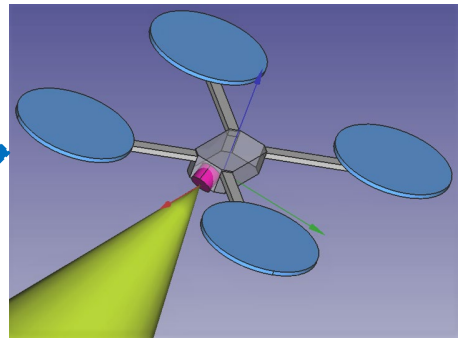
Graph Visualization

*Source: Tom Sawyer with SysML v2*

## Systems Modeling API

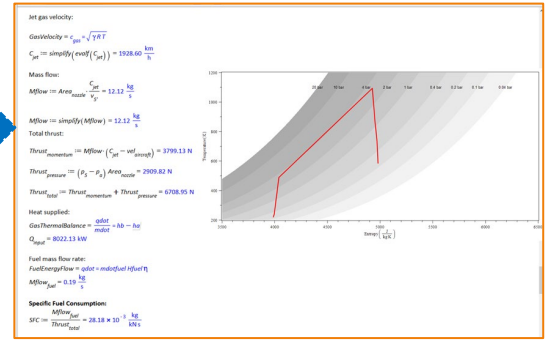
### SysML v2

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



CAD/CAD Viewer

*Source: FreeCAD with SysML v2*



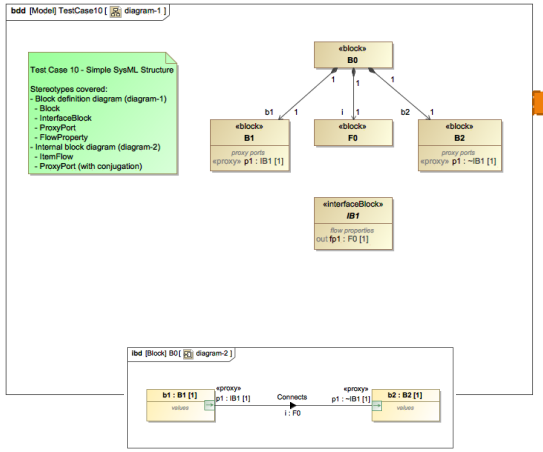
Analysis Solver

*Source: Maple with SysML v2*

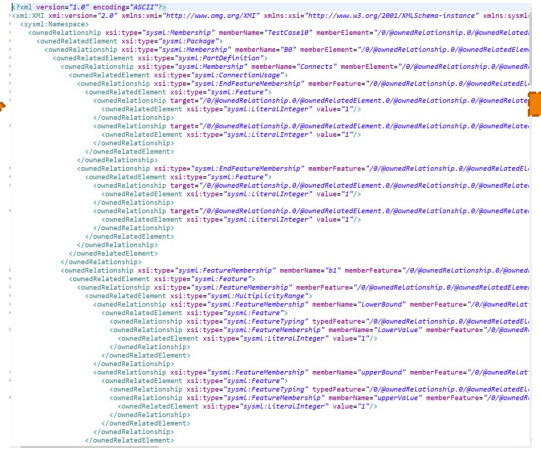
# SysML v1 to SysML v2 Transformation

SST

## SysML v1 Model




## SysML v2 XML



## SysML v2 Textual Notation

```
package eVehicleLibrary {
 attribute def ElectricEnergy;
 attribute def BatteryCapacity :> ScalarValues::Integer;
 attribute def Speed :> ScalarValues::Integer;
 port def PowerOutPort { out energy : ElectricEnergy;
 }
 interface def PowerInterface {
  end supplierPort : PowerOutPort;
  end consumerPort : ~PowerOutPort;
 }
}
package eVehicleDefinitions {
    import eVehicleLibrary::*;
 part def Wheel {
  value size : ScalarValues::Integer;
 }
 part def Battery {
  value capacity : BatteryCapacity;
 }
 part def Engine;
}
....
```

Source: SST Track 3 Presentation
Yves Bernard, Tim Weilkiens
08 February 2022




# Progress and Plans

# Planned vs Completed Work Since IW 2022

SST

## Language

- Finalize specification of graphical syntax
- Time semantics and change/time events
- Simple geometry (spatial semantics and shape library)
- Language extension
- Behavior execution guidance
- Model interchange
- Conformance cases
- SysML v1 to v2 transformation

## API & Services

- Cross project element referencing
- Conformance tests
- OSLC PSM
- Query specification updates
- API Recipes

*Work completed in 2022*
*Additional work to be done during finalization*


# SysML v2 Milestones

SST

| | |
|---|---|
| December, 2017 | SysML v2 RFP issued |
| June, 2018 | SysML v2 API & Services RFP issued |
| August, 2020 | Initial Submission |
| February, 2021 | Stakeholder Review |
| August, 2021 | Revised Submission |
| November, 2021 | 2nd Revised Submission (OMG evaluation initiated) |
| September, 2023 | Specification Review at OMG |
| November, 2022 | 3rd Revised Submission |
| 1st Qtr 2023 | Final Submission (beta specification) |
| 2024 | Adopted Specification (pending OMG approvals) |

# Summary


# Summary

SST

- SysML v2 is addressing SysML v1limitations to improve MBSE adoption and effectiveness
  - Precision, expressiveness
  - Regularity, usability
  - Interoperability with other engineering models and tools
- Approach
  - SysML v2 metamodel with formal semantics architected to overcome fundamental UML limitations
  - Flexible graphical notations and textual notation
  - Standardized API for interoperability
  - Transformation specification from SysML v1 to SysML v2
- Final submission planned for Q1 2023 and final adopted specification in 2024

# SST Public Repositories
# Current Release: 2022-12

SST

- 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)

# INCOSE IW
# Follow-up Sessions on SysML v2

SST

Monday, January 30

10:30 – 15:00 PT SysML v2 Overview and Demo (Pier 7)

16:00 – 18:00 PT SysML v2 Transition Workshop (Salon H)

# Thank you!!