SysML v2 Submission Team (SST)
SysML v2 Update
January 30, 2021

Sanford Friedenthal
SST Co-Lead
safriedenthal@gmail.com

www.incose.org/IW2021
Purpose

- Provide an update from the 2020 IW on the status of SysML v2

Agenda

- SysML v2 Background and Objectives
- SysML v2 Submission Team
- SysML v2 Approach
- SysML v2 Language Architecture
- SysML v2 API & Services
- Summary

26 January 2020
SysML v2
Background and Objectives
SysML has evolved to address user and vendor needs
- v1.0 adopted in 2006; v1.6 current version; v1.7 in process

SysML has facilitated awareness and adoption of MBSE

Much has been learned from using SysML for MBSE

Supports the specification, analysis, design, and verification and validation of complex systems that may include hardware, software, information, processes, personnel, and facilities
SysML v2 Objectives

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
SysML v2 Milestones

December, 2017  SysML v2 RFP issued

June, 2018  SysML v2 API & Services RFP issued

August, 2020  Initial Submission

August, 2021  Final Submission (to be confirmed)

4th Qtr 2021  Beta Specification (pending OMG approvals)

4th Qtr 2022  Finalized Specification (pending OMG approvals)
SysML v2 Submission Team (SST)
A broad team of end users, vendors, academics, and government liaisons
  - Over 100 members representing 70+ organizations

Developing submissions to both RFPs

Driven by RFP requirements and user needs
## SST Participating Organizations

<table>
<thead>
<tr>
<th>SST MLv2</th>
<th>SST MLv2</th>
<th>SST MLv2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aerospace Corp</td>
<td>IncQuery Labs</td>
<td>Qualtech Systems, Inc (QSI)</td>
</tr>
<tr>
<td>Airbus</td>
<td>Intercax</td>
<td>Raytheon</td>
</tr>
<tr>
<td>ANSYS medini</td>
<td>Itemis</td>
<td>Rolls Royce</td>
</tr>
<tr>
<td>Aras</td>
<td>Jet Propulsion Lab</td>
<td>Saab Aeronautics</td>
</tr>
<tr>
<td>Army Aviation &amp; Missile Center</td>
<td>John Deere</td>
<td>SAF Consulting *</td>
</tr>
<tr>
<td>Army CBRND</td>
<td>Kenntnis</td>
<td>SAIC</td>
</tr>
<tr>
<td>BAE</td>
<td>KTH Royal Institute of Technology</td>
<td>Siemens</td>
</tr>
<tr>
<td>BigLever Software</td>
<td>LieberLieber</td>
<td>Sierra Nevada Corporation</td>
</tr>
<tr>
<td>Boeing</td>
<td>Lightstreet Consulting</td>
<td>Simula</td>
</tr>
<tr>
<td>Army CCDC Armaments Center</td>
<td>Lockheed Martin</td>
<td>Space Cooperative</td>
</tr>
<tr>
<td>CEA</td>
<td>MathWorks</td>
<td>Sodius Willert</td>
</tr>
<tr>
<td>Contact Software</td>
<td>Maplesoft</td>
<td>System Strategy *</td>
</tr>
<tr>
<td>DEKonsult</td>
<td>Mgnite Inc</td>
<td>Tata Consultancy Services</td>
</tr>
<tr>
<td>Delligatti Associates</td>
<td>MITRE</td>
<td>Thales</td>
</tr>
<tr>
<td>Draper Lab</td>
<td>ModelAlchemy Consulting</td>
<td>Thematix</td>
</tr>
<tr>
<td>Elbit Systems of America</td>
<td>Model Driven Solutions</td>
<td>Tom Sawyer</td>
</tr>
<tr>
<td>ESTACA</td>
<td>Model Foundry</td>
<td>UFRPE</td>
</tr>
<tr>
<td>Ford</td>
<td>NIST</td>
<td>Vera C. Rubin Observatory</td>
</tr>
<tr>
<td>Fraunhofer FOKUS</td>
<td>No Magic/Dassault Systemes</td>
<td>Vitech</td>
</tr>
<tr>
<td>General Motors</td>
<td>OAR</td>
<td>88solutions</td>
</tr>
<tr>
<td>George Mason University</td>
<td>Obeo</td>
<td></td>
</tr>
<tr>
<td>GfSE</td>
<td>OOSE</td>
<td></td>
</tr>
<tr>
<td>Georgia Tech/GTRI</td>
<td>Ostfold University College</td>
<td></td>
</tr>
<tr>
<td>IBM</td>
<td>Phoenix Integration</td>
<td></td>
</tr>
<tr>
<td>Idaho National Laboratory</td>
<td>PTC</td>
<td></td>
</tr>
</tbody>
</table>
1. Project Management – Ed Seidewitz, Sandy Friedenthal
   • Infrastructure – John Watson, Chris Delp
2. Requirements V&V – Sandy Friedenthal
3. Profile Development – Yves Bernard, Tim Weilkiens
4. Metamodel Development – Karen Ryan, Chas Galey
5. API/Services Development – Manas Bajaj
SysML v2 Approach
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 and execution
  - Graphical, Tabular, Textual
- Standardized API to access the model
The following 16 validation cases capture required language functionality and reflect 2/3 of the SysML v2 RFP requirements:

- 1-Parts Tree
- 2-Parts Interconnection
- 3-Function-based Behavior
- 4-Functional Allocation
- 5-State-based Behavior
- 6-Individuals and Snapshots
- 7-Variant Configuration
- 8-Requirements
- 9-Verification
- 10-Analysis and Trades
- 11-View and Viewpoint
- 12-Dependency and Allocation
- 13-Model Containment
- 14-Language Extension
- 15-Properties, Values, & Expressions
- 16-Proxy validation case

Base capability as of January 2020 is in process.
SysML v2 Language
Capabilities

SysML v2
Language
Structure
- decomposition
- interconnection
- classification

Behavior
- function-based
- state-based

Analysis
- analysis cases
- expression language

Requirements & Constraints

Verification

View & Viewpoint

Individuals & Snapshots
• **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.
    There can be many different usages of the same definition element in either different contexts or the same context.
  - Pattern is applied consistently throughout the language.

• **Variability**
  - *Variation* points represent elements that can vary.
  - *Variation* applies to all definition and usage elements.
  - A variant represents a particular choice at a variation point.
  - A choice at one variation point can constrain choices at other variation points.
  - A system can be configured by making choices at each variation point consistent with specified constraints.
package 'Vehicle Parts Tree' {
    part vehicle {
        attribute mass;
        perform providePower;
        part engine {
            attribute mass;
            perform generateTorque;
            part cylinders [6];
        }
        part transmission {
            attribute mass;
            perform amplifyTorque;
        }
    }
    package 'Vehicle Action Tree' {
        action providePower {
            action generateTorque;
            action amplifyTorque;
        }
    }
}
SysML v2 Notation (2 of 2)
Textual and Graphical

```
interface def Drive {
    end enginePort : DrivePort;
    end transmissionPort : ~DrivePort;
}

part vehicle : Vehicle {
    part engine : Engine { port drivePort : DrivePort; }
    part transmission : Transmission { port drivePort : ~DrivePort; }
    interface : Drive
    connect engine::drivePort to transmission::drivePort;
}
```

Tom Sawyer Visualization Prototype
SysML v2 Language Architecture
SysML v2 Language Architecture

**Systems Modeling Language (SysML)**

- Root Syntax
- Core Syntax
- Kernel Syntax
- Systems Syntax

- Metamodel
- Semantic Library

**Core Syntax**

- Core Semantics

**Kernel Modeling Language (KerML)**

- Kernel Syntax
- Kernel Model Library

- Metamodel
- Semantic Library

**Core**

- Core (KerML)

**Kernel Syntax**

- Kernel Model (KerML)

- Systems and Domain Model Libraries

**Root Syntax**

- Root Syntactic Elements without Model-Level Semantics (e.g., packaging)

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

**Declarative semantic base elements modeled using KerML**

**Direct semantic mapping to formal logic**

**30 January 2021**

**Root syntactic elements without model-level semantics (e.g., packaging)**
SysML v2 API & Services

- Enables other tools and applications to access SysML models in a standard way
- Provides services to:
  - Create, update, and delete elements
  - Query and navigate model
  - Other services including support for model management, analysis, transformation, and file export generation
- Support systems engineering functional threads such as change impact assessment
- Facilitates use of different implementation technologies such as REST/HTTP, Java, or OSLC

30 January 2021
Pilot Implementation Using Standard API

High-Level Architecture of SysML v2 Testbed

- Programmatic Authoring
- Graphical Authoring Environment
- Textual Authoring Environment

OSLC 3.0 API (PSM 2)

REST/HTTP (PSM 1)

Java API (PSM 3)

Model CRUD Services

- RDBMS
- NoSQL
- GraphDB

API Definition (PIM)

Meta-Model based on KerML

Active validation based on Conforms to PIM

I/O conforms to

Schema can be influenced by

30 January 2021
SysML v2 API and Services Progress

- Mandatory Services
  - Model Navigation, Creation, Update, Deletion Services
  - External Relationship Management Service

- Non-Mandatory Services
  - Model Query Service
  - Advanced Model Construction Services (realized by client side API calls)
  - Model View and Viewpoint Management Services
  - Model Analysis Services
  - Model Management Services
    - Versioning Service
    - Branching Service
  - Model Transformation Services
  - General Services – Timestamp and UUID generation, API Call Back

- API Platform-Specific Models (API PSMs)
  - REST/HTTP binding
  - OSLC 3.0 binding

Base capability as of January 2021 in process
Summary
Contrasting SysML v1 with SysML v2

- **Simpler to learn and use**
  - Systems engineering concepts designed into metamodel versus added-on
  - Consistent use of 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
  - Reified relationships (e.g., membership, annotation)

- **More expressive**
  - Variant modeling
  - Analysis case
  - Trade-off analysis
  - Individuals, snapshots, time slices
  - More robust quantitative properties (e.g., vectors, ..)
  - Query expressions

- **More extensible**
  - Simpler language extension capability
    - Based on model libraries

- **More interoperable**
  - Standardized API
SST Public Repositories

- Current release: 2020-12 (2021-01 planned for early February)
- Monthly release repository
- Release content
  - Specification documents (for KerML, SysML and API)
  - Training material for SysML textual notation
  - Example models (in textual notation)
  - 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](https://github.com/Systems-Modeling)
- Google group for comments and questions
  - [https://groups.google.com/g/SysML-v2-Release](https://groups.google.com/g/SysML-v2-Release)
  (to request membership, provide name, affiliation and interest)
Summary

- SysML v2 is addressing SysML v1 limitations to improve MBSE adoption and effectiveness
  - Precision, expressiveness, usability
  - Interoperability with other engineering models and tools

- Initial approach
  - Simplified SysML v2 metamodel with formal semantics overcomes fundamental UML limitations
  - Flexible graphical notations and textual notation
  - Standardized API for interoperability

- Roadmap established to revised submission
Upcoming Events

• SysML v2 Session at IW - Demo and Q&A at IW (2 repeat sessions)
  o Session 1 on Sun, Jan 31, 13:00 – 15:00 US ET
  o Session 2 on Mon, Feb 1, 09:00 – 11:00 US ET

• SysML v2 Stakeholder Review (2 repeat sessions)
  o Session 1 on Tue, Feb 23, 09:00 – 15:00 US ET
  o Session 2 on Thu, Feb 25, 11:00 – 17:00 US ET
Primary References


Friedenthal S., Seidewitz E., A Preview of the Next Generation System Modeling Language (SysML v2), Project Performance International (PPI), Systems Engineering Newsletter, PPI SyEN 95 27 November, 2020
Thank you!!