Progress on SysML v2

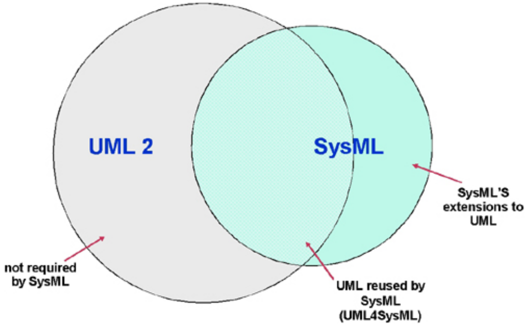
Hans Peter de Koning – ESA

14 November 2019, ADCSS 2019

ESA/ESTEC – Noordwijk – The Netherlands

What is SysML?

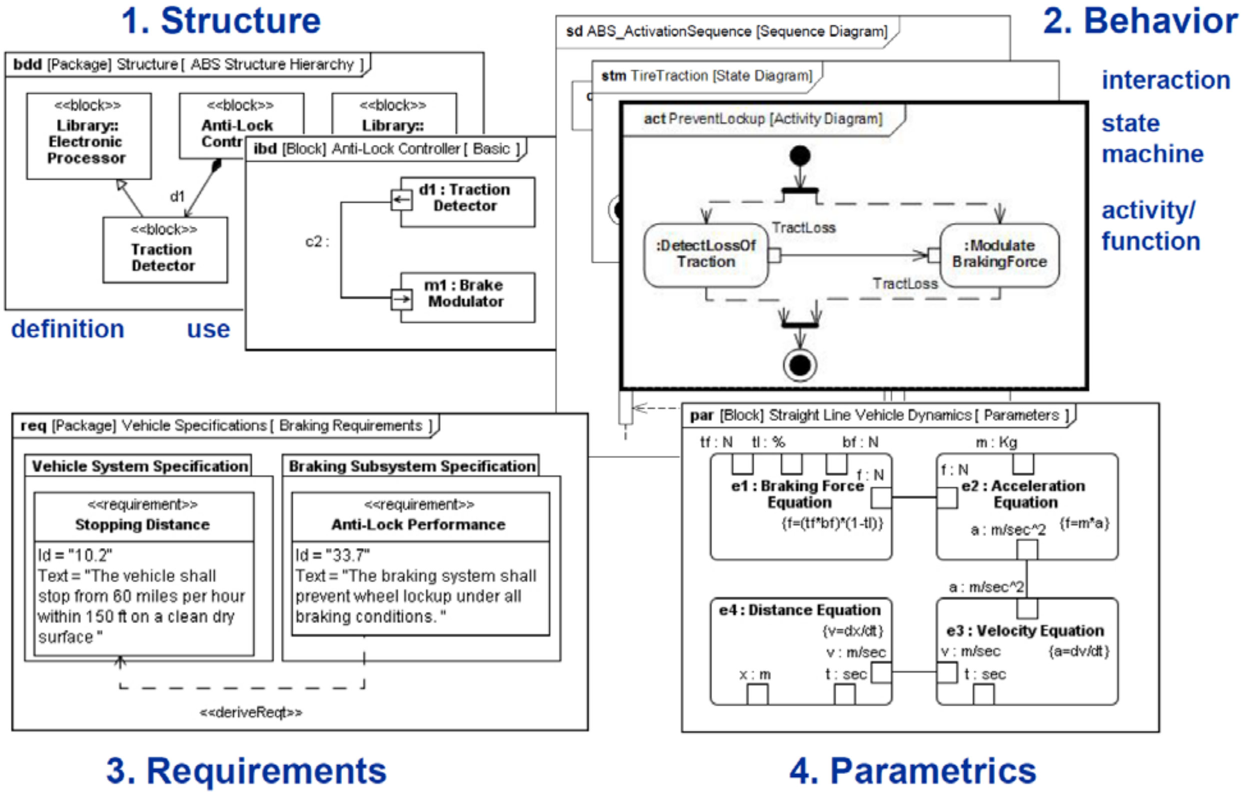
- Systems Modeling Language – profile / extension of UML



> general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities
> from http://www.omgsysml.org/

- OMG standard since 2007 – v1.0
- In real industrial use since 2010 – v1.2
- Currently v1.5 – released May 2017
- v1.6 almost published – v1.7 in development

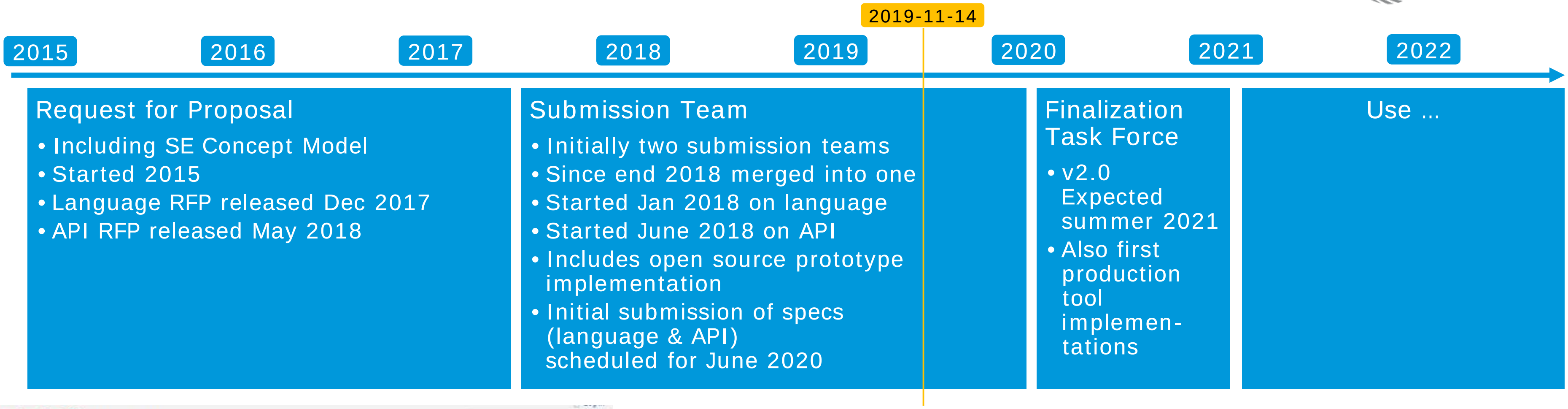
The four pillars of SysML


1. Structure
bdd [Package] Structure [ABS Structure Hierarchy]
<<block>> Library:: Electronic Processor
<<block>> Anti-Lock Contr
<<block>> Library::
d1
<<block>> Traction Detector
ibd [Block] Anti-Lock Controller [Basic]
d1 : Traction Detector
c2 :
m1 : Brake Modulator
definition
use
2. Behavior
sd ABS_ActivationSequence [Sequence Diagram]
stm TireTraction [State Diagram]
act PreventLockup [Activity Diagram]
:DetectLossOf Traction
TractLoss
:Modulate BrakingForce
TractLoss
interaction
state machine
activity/ function
req [Package] Vehicle Specifications [Braking Requirements]
Vehicle System Specification
Braking Subsystem Specification
<<requirement>> Stopping Distance
Id = "10.2"
Text = "The vehicle shall stop from 60 miles per hour within 150 ft on a clean dry surface"
<<requirement>> Anti-Lock Performance
Id = "33.7"
Text = "The braking system shall prevent wheel lockup under all braking conditions."
<<deriveReqt>>
3. Requirements
par [Block] Straight Line Vehicle Dynamics [Parameters]
tf : N
tl : %
bf : N
m : Kg
f : N
e1 : Braking Force Equation
{f=(tf*bf)*(1-tl)}
f : N
e2 : Acceleration Equation
a : m/sec^2
{f=m*a}
a : m/sec^2
e4 : Distance Equation
{v=dx/dt}
v : m/sec
x : m
t : sec
e3 : Velocity Equation
v : m/sec
{a=dv/dt}
t : sec
4. Parametrics

All diagram types:

- Requirement
- Structure
  - Block Definition
  - Internal Block
  - Parametric
  - Package
- Behavior
  - Activity
  - Sequence
  - State Machine

OMG SysML v2 Timeline



Request for Proposal

- Including SE Concept Model
- Started 2015
- Language RFP released Dec 2017
- API RFP released May 2018

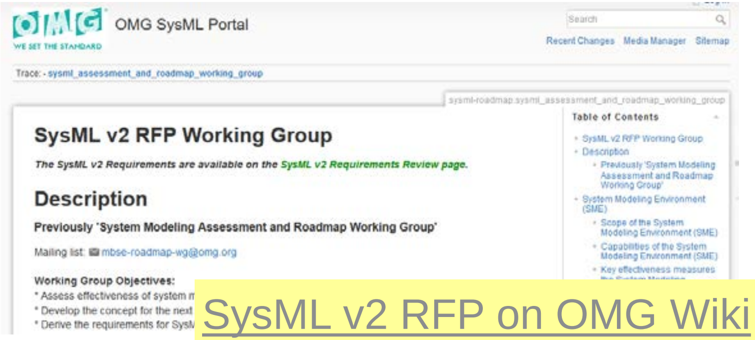
Submission Team

- Initially two submission teams
- Since end 2018 merged into one
- Started Jan 2018 on language
- Started June 2018 on API
- Includes open source prototype implementation
- Initial submission of specs (language & API) scheduled for June 2020

Finalization Task Force

- v2.0 Expected summer 2021
- Also first production tool implementations

Use ...



OMG SysML Portal

Trace: - sysml_assessment_and_roadmap_working_group

sysml-roadmap:sysml_assessment_and_roadmap_working_group

SysML v2 RFP Working Group

The SysML v2 Requirements are available on the SysML v2 Requirements Review page.

Description

Previously 'System Modeling Assessment and Roadmap Working Group'

Mailing list: mbse-roadmap-wg@omg.org

Working Group Objectives:

* Assess effectiveness of system m
* Develop the concept for the next
* Derive the requirements for SysM

Table of Contents

- SysML v2 RFP Working Group
- Description
  - Previously 'System Modeling Assessment and Roadmap Working Group'
- System Modeling Environment (SME)
  - Scope of the System Modeling Environment (SME)
  - Capabilities of the System Modeling Environment (SME)
  - Key effectiveness measures

SysML v2 RFP on OMG Wiki

MBSE Meeting at MODELS 2018, Copenhagen

SysML v2 and MBSE: The Next Ten Years
16 October 2018

Ed Seidewitz
Chief Technology Officer
Model Driven Solutions

Copyright © 2018 Model Driven Solutions, Inc.

Public overview SysML v2 approach by Ed Seidewitz

SysML v2 Requirements and Constraints

- Extensive RFP (http://www.omgsysml.org/SysML-2.htm)
  - Also very relevant input to MB4SE / future harmonisation
- SysML v2 shall be based on SMOF (Semantic Meta Object Facility)
  - Provides support for temporal aspects and multiple classifications
- Must provide migration path from SysML v1 – that can be automated

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

Substantially reduce learning curve for systems engineers

SysML v2 Submission Team (SST)

SST

- SysML v2 Submission Team (SST) formed December 2017
    - Leads: Sandy Friedenthal, Ed Seidewitz
- A broad team of end users, vendors, academics, and government liaisons
    - Currently 110 members from 61 organizations
- Developing submissions to both RFPs
- Driven by RFP requirements and user needs

Submission Team Tracks

SST

- Track 1: Project Management – Ed Seidewitz, Sandy Friedenthal
- Track 2: Requirements V&V
- Track 3: Profile Development
- Track 4: Metamodel Development
- Track 5: API/Services Development
- Track 6: Pilot Implementation

SST Participating Organizations

SST

Academia/Research | End User | Tool Vendors | Government Rep | INCOSE rep *

- Aerospace Corp
- Airbus
- ANSYS medini
- Aras
- ARDEC
- Army Aviation & Missile Center
- BAE
- BigLever Software
- Boeing
- CEA
- Contact Software
- Draper Lab
- Elbit Systems of America
- European Space Agency
- Ford
- Fraunhofer FOKUS
- General Motors
- George Mason University
- GfSE
- GTRI
- IBM
- Idaho National Laboratory
- IncQuery Labs
- Intercax
- Itemis
- Jet Propulsion Lab
- John Deere
- Kenntnis
- LieberLieber
- Lightstreet Consulting
- Lockheed Martin
- LSST
- Maplesoft
- MITRE
- ModelAlchemy Consulting
- Model Driven Solutions
- Model Foundry
- NIST
- No Magic
- Obeo
- OOSE
- Ostfold University College
- Phoenix Integration
- PTC
- Raytheon
- Rolls Royce
- SAF Consulting *
- SAIC
- Siemens
- Sierra Nevada Corporation
- Simula
- System Strategy *
- Tata Consultancy Services
- Thales
- Thematix
- Tom Sawyer
- University of Cantabria
- University of Alabama in Huntsville
- University of Detroit Mercy
- Vitech
- 88solutions

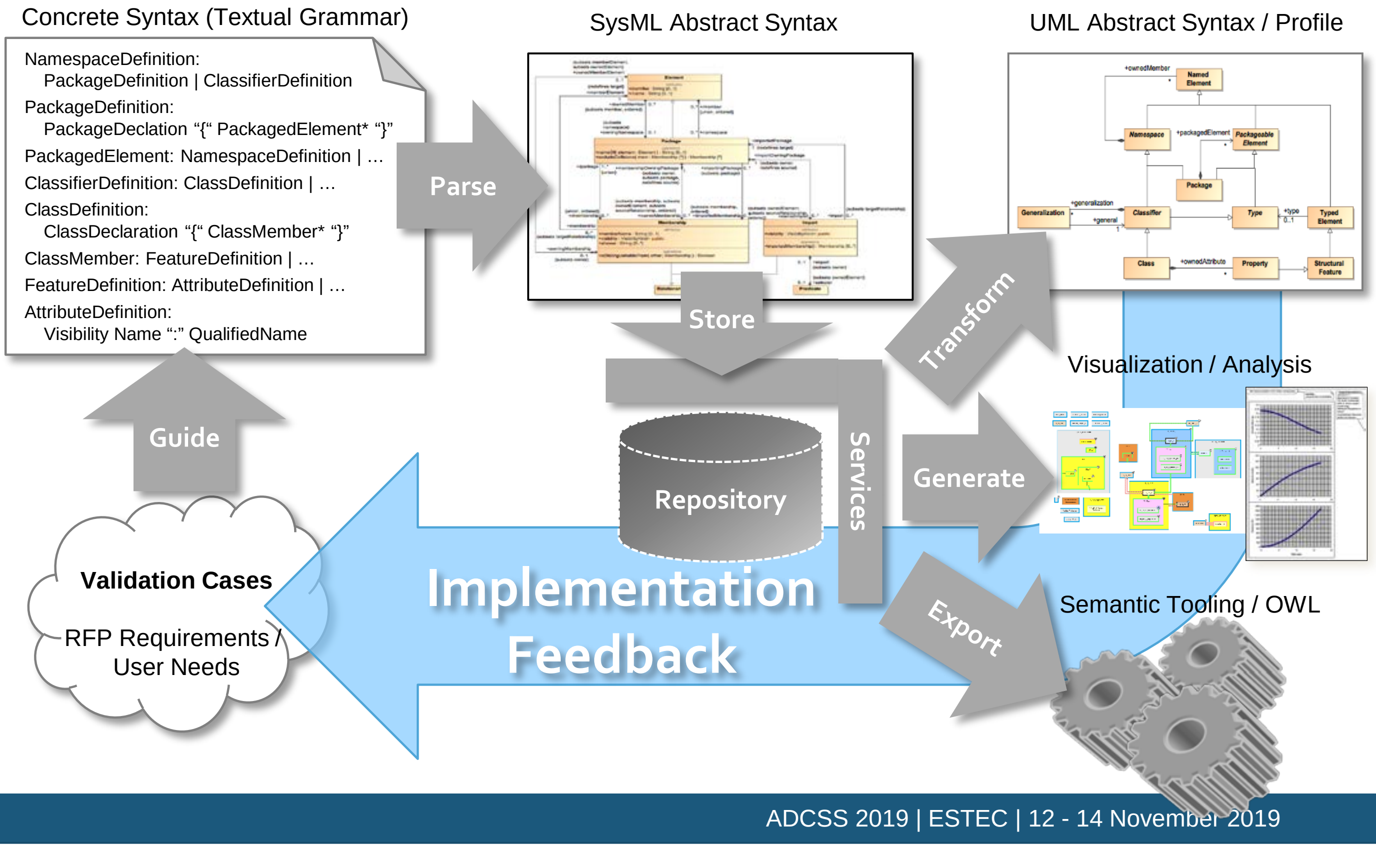
61 in total – many aerospace – most major vendors on board

Key Elements of SysML v2

SST

- New Metamodel that is not constrained by UML
  - Grounded in formal semantics
- Robust visualizations based on flexible view & viewpoint specification and execution
  - Graphical, Tabular, Textual
- Standardized API to access the model

SST Agile Collaborative Approach



Concrete Syntax (Textual Grammar)

```
NamespaceDefinition:
   PackageDefinition | ClassifierDefinition
PackageDefinition:
   PackageDeclation “{“ PackagedElement* “}”
PackagedElement: NamespaceDefinition | …
ClassifierDefinition: ClassDefinition | …
ClassDefinition:
   ClassDeclaration “{“ ClassMember* “}”
ClassMember: FeatureDefinition | …
FeatureDefinition: AttributeDefinition | …
AttributeDefinition:
   Visibility Name “:” QualifiedName
```

SysML v2 versus SysML v1 Comparison

SST

- Initial comparison highlights the following intended benefits
  - Additional functionality (e.g., variants, trade-off, ..)
  - Integrated concepts (e.g., between structure and behavior)
  - Ease of use (e.g., built in redefinition at every level of nesting)
  - Clarification of concepts (e.g., individuals/snapshots vs instances)

Usage Focused Modeling Approach

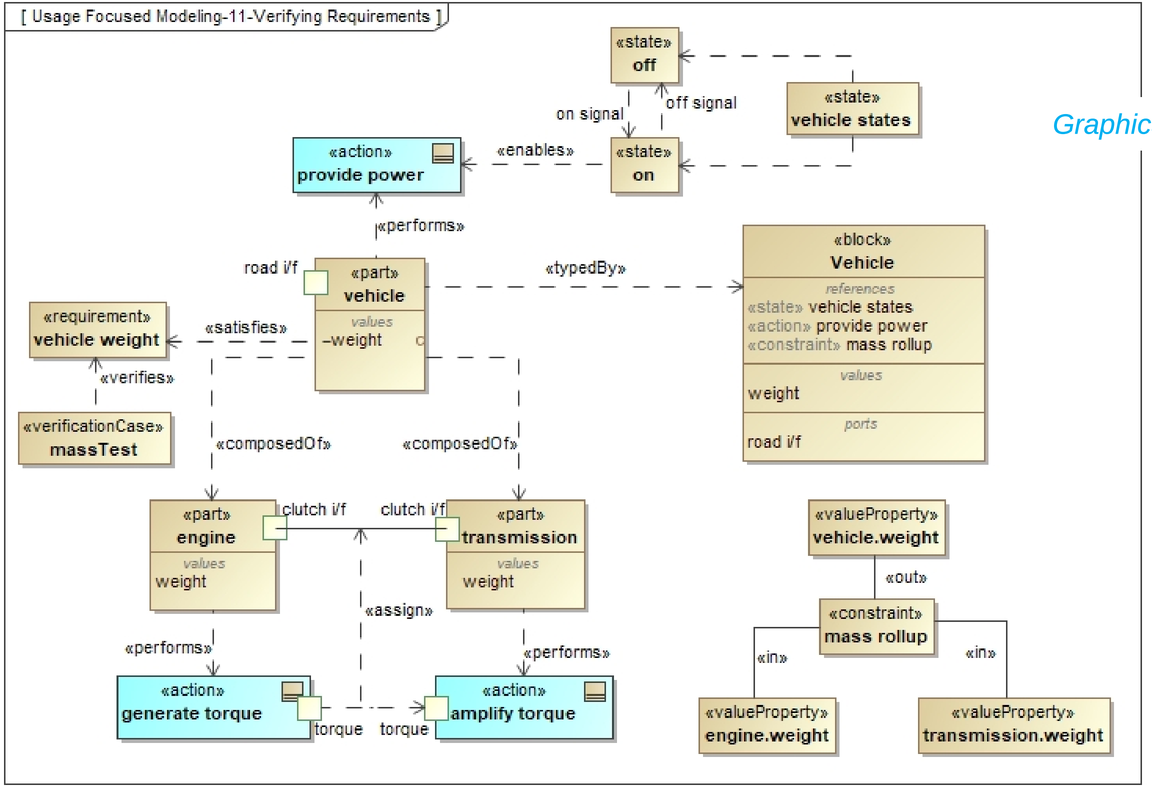
SST

A paradigm shift to make SysML v2 more precise and intuitive to use

- Emphasizes modeling of localized *usages (e.g., parts on an ibd)*
  - Decompose, connect, relate, and group usages
- Supports other language requirements
  - variant design configurations, individuals, analysis, verification, …
- Facilitates creating and modifying design configurations including structure and behavior to satisfy their requirements

Usage Focused Modeling Approach
Multiple Views of a System

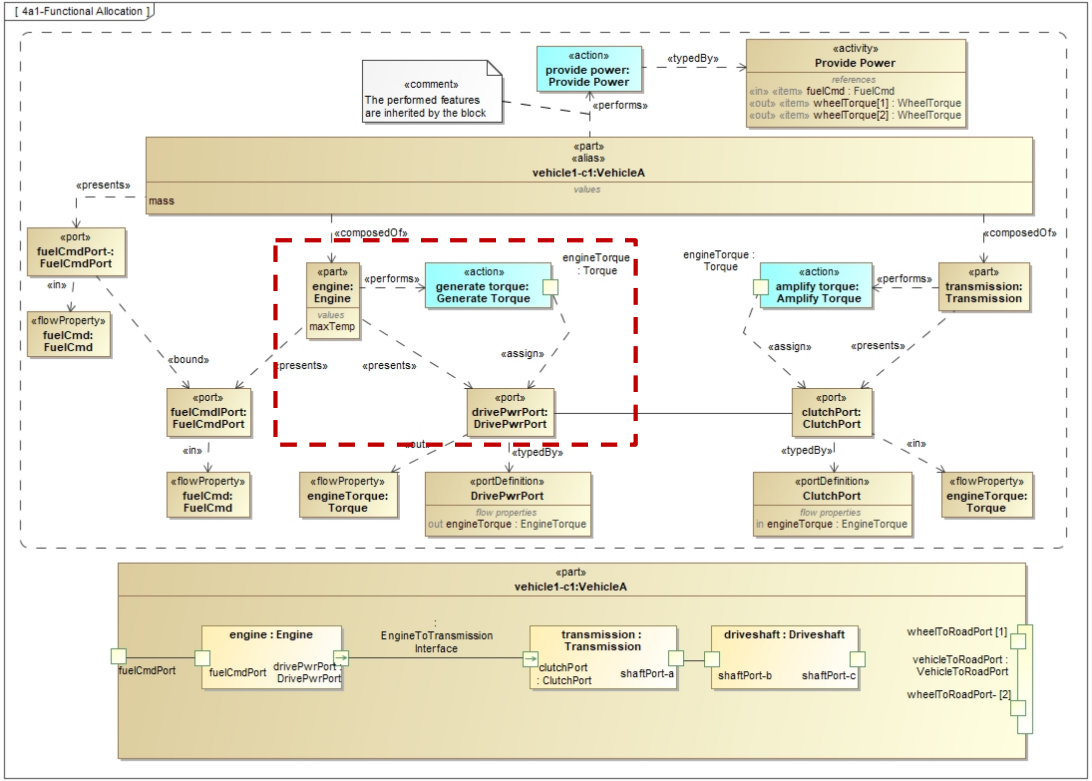
SST



Graphical notation for illustrative purposes only

Function-based Behavior
SysML v2 Functional Allocation

SST



Information Model Approach and Textual Syntax

- Started from KerML (Kernel Model Language)
  - Minimalistic meta-model (M2)
  - Normative / informative model libraries (M1, M0)
  - Feature (similar to UML property) is a ‘first class citizen’ and can be nested
    - Self-standing features can be defined
    - Addresses the deeply nested feature inconveniences of SysML v1
    - E.g. mass defined as `feature mass: MassValue[1..1]` can be used on a block directly: `MyBlock.mass = 24@[kg]`
- New textual syntax – based on fUML ALF
  - Very powerful and concise – alternative for graphical notation
  - Main work in Track 6 Prototype Implementation

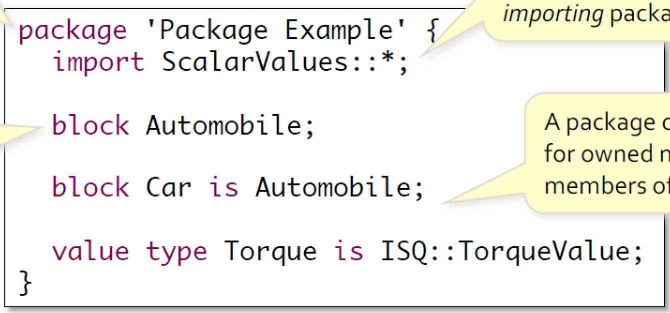
Textual Language Examples Packages

SST

A *package* acts as a *namespace* for its members and a *container* for its owned members.

ⓘ A name with spaces or other special characters is surrounded in singe quotes.

An *import* adds all the members of the *imported* package to the *importing* package.



```
package 'Package Example' {
  import ScalarValues::*;

  block Automobile;

  block Car is Automobile;

  value type Torque is ISQ::TorqueValue;
}
```

The *owned members* of a package are elements directly contained in the package.

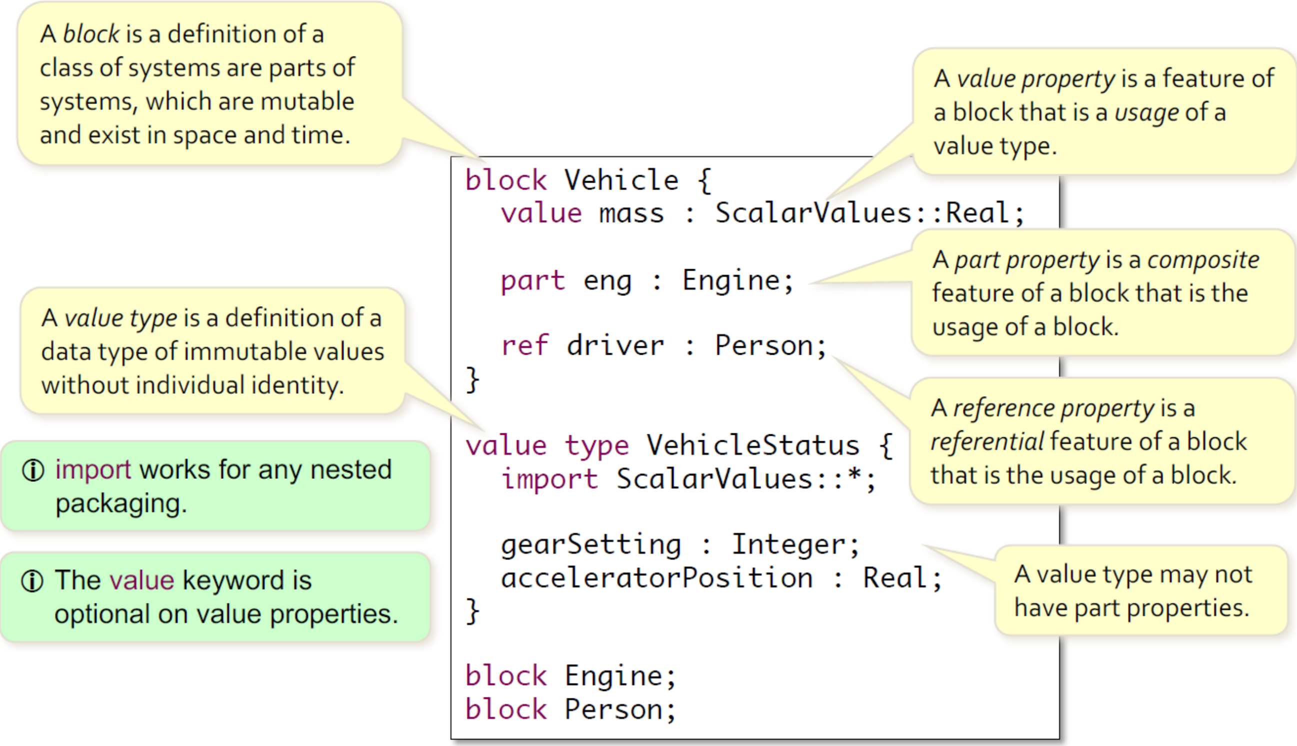
A package can introduce *aliases* for owned members or individual members of other packages.

ⓘ A *qualified name* is a package name (which may itself be qualified) followed by the name of one of its members, separated by :: .

Courtesy Ed Seidewitz, Model Driven Solutions

Textual Language Examples
Blocks and Value Types

SST



```
block Vehicle {
  value mass : ScalarValues::Real;

  part eng : Engine;

  ref driver : Person;
}

value type VehicleStatus {
  import ScalarValues::*;

  gearSetting : Integer;
  acceleratorPosition : Real;
}

block Engine;
block Person;
```

A *block* is a definition of a class of systems are parts of systems, which are mutable and exist in space and time.

A *value type* is a definition of a data type of immutable values without individual identity.

ⓘ import works for any nested packaging.

ⓘ The value keyword is optional on value properties.

A *value property* is a feature of a block that is a *usage* of a value type.

A *part property* is a *composite* feature of a block that is the usage of a block.

A *reference property* is a *referential* feature of a block that is the usage of a block.

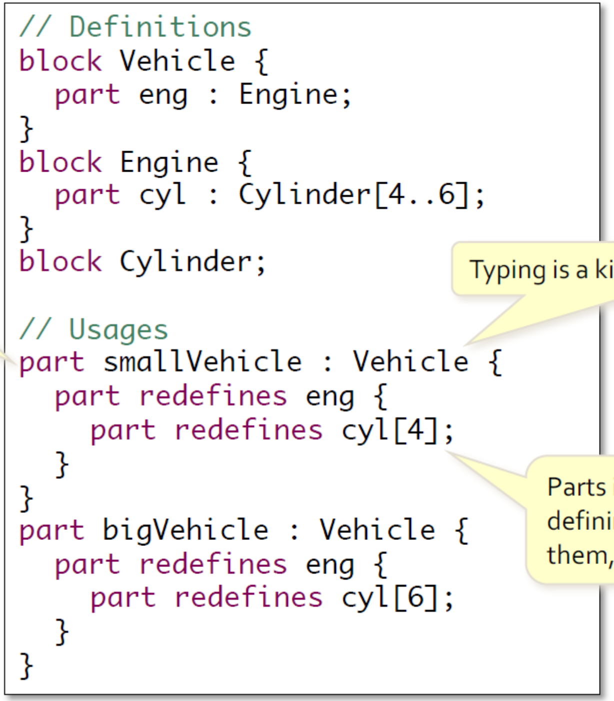
A value type may not have part properties.

Courtesy Ed Seidewitz,
Model Driven Solutions

SST

Textual Language Examples
Parts (and nested redefinitions)

Parts can be specified outside the context of a specific block.



```
// Definitions
block Vehicle {
  part eng : Engine;
}
block Engine {
  part cyl : Cylinder[4..6];
}
block Cylinder;

// Usages
part smallVehicle : Vehicle {
  part redefines eng {
    part redefines cyl[4];
  }
}
part bigVehicle : Vehicle {
  part redefines eng {
    part redefines cyl[6];
  }
}
```

Typing is a kind of generalization.

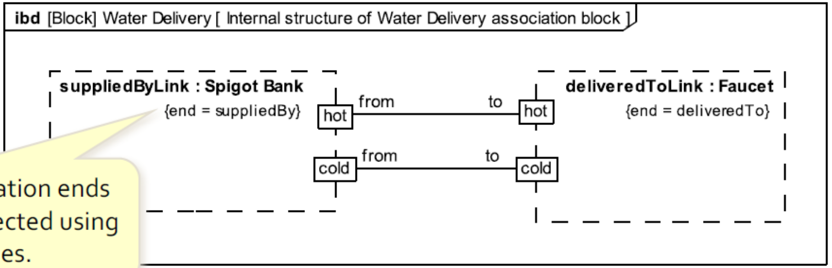
Parts inherit properties from their defining blocks and can redefine them, to any level of nesting.

Courtesy Ed Seidewitz, Model Driven Solutions

Textual Language Examples Interface Decomposition

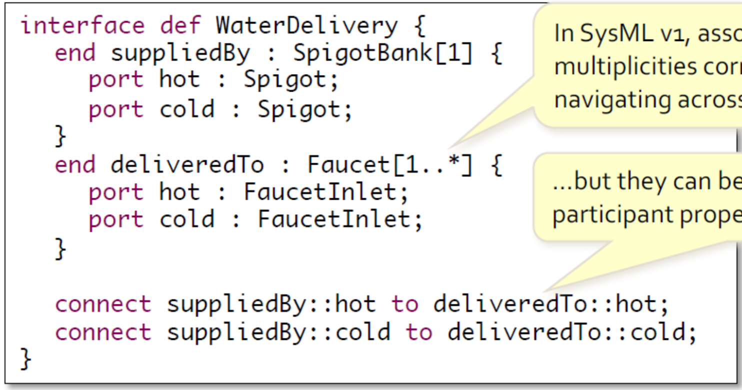
SST

From SysML v1.5 spec



ibd [Block] Water Delivery [Internal structure of Water Delivery association block]
suppliedByLink : Spigot Bank
{end = suppliedBy}
hot
cold
from
to
from
to
hot
cold
deliveredToLink : Faucet
{end = deliveredTo}

In SysML v1, association ends could only be connected using participant properties.



```
interface def WaterDelivery {
   end suppliedBy : SpigotBank[1] {
      port hot : Spigot;
      port cold : Spigot;
   }
   end deliveredTo : Faucet[1..*] {
      port hot : FaucetInlet;
      port cold : FaucetInlet;
   }

   connect suppliedBy::hot to deliveredTo::hot;
   connect suppliedBy::cold to deliveredTo::cold;
}
```

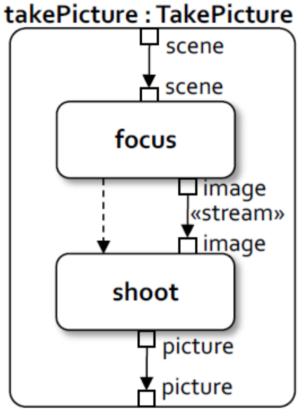
In SysML v1, association ends have multiplicities corresponding to navigating across the association…

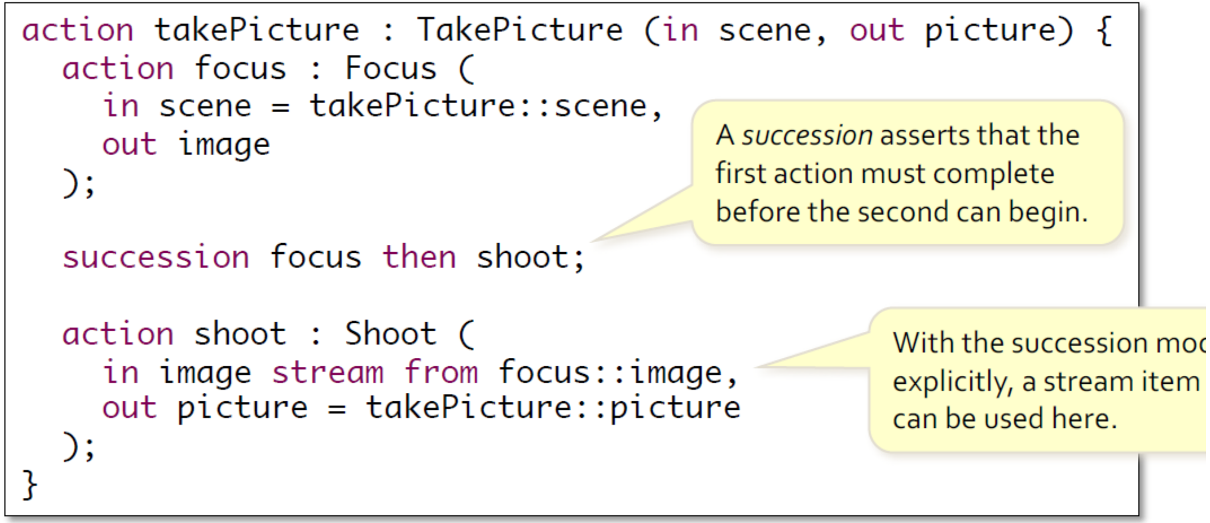
…but they can be interconnected like participant properties.

Courtesy Ed Seidewitz, Model Driven Solutions

Textual Language Examples
Activities/Functions (control flow / item flow)

SST





```
action takePicture : TakePicture (in scene, out picture) {
  action focus : Focus (
    in scene = takePicture::scene,
    out image
  );

  succession focus then shoot;

  action shoot : Shoot (
    in image stream from focus::image,
    out picture = takePicture::picture
  );
}
```

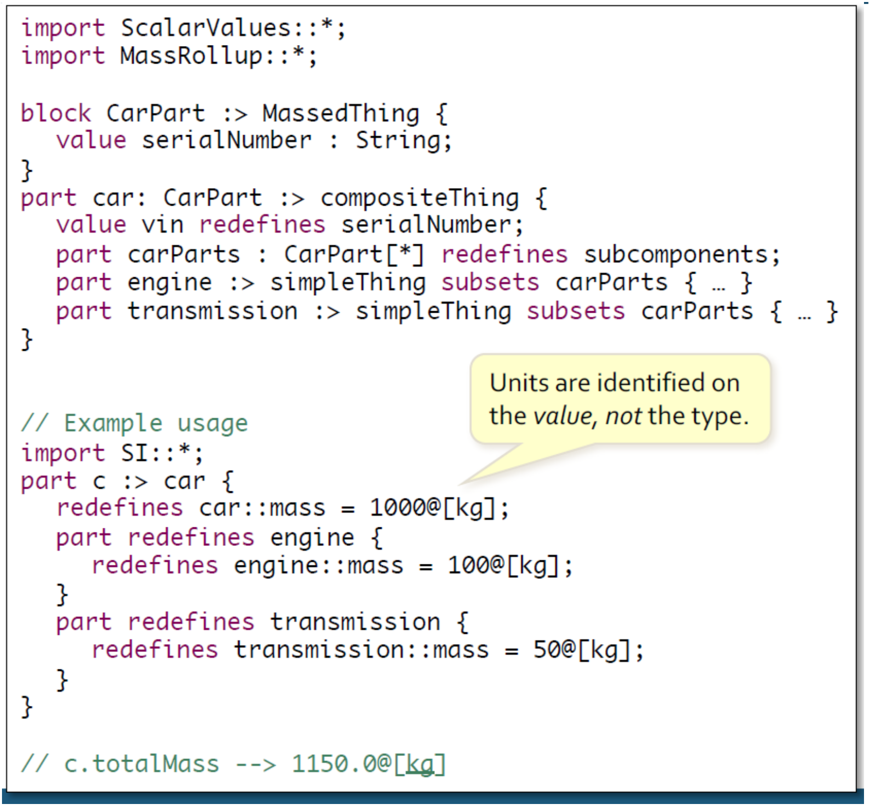
A *succession* asserts that the first action must complete before the second can begin.

With the succession mod explicitly, a stream item can be used here.

Courtesy Ed Seidewitz,
Model Driven Solutions

Textual Language Examples Quantities, Units and Scales

SST



```
import ScalarValues::*;
import MassRollup::*;

block CarPart :> MassedThing {
   value serialNumber : String;
}
part car: CarPart :> compositeThing {
   value vin redefines serialNumber;
   part carParts : CarPart[*] redefines subcomponents;
   part engine :> simpleThing subsets carParts { … }
   part transmission :> simpleThing subsets carParts { … }
}


// Example usage
import SI::*;
part c :> car {
   redefines car::mass = 1000@[kg];
   part redefines engine {
      redefines engine::mass = 100@[kg];
   }
   part redefines transmission {
      redefines transmission::mass = 50@[kg];
   }
}

// c.totalMass --> 1150.0@[kg]
```

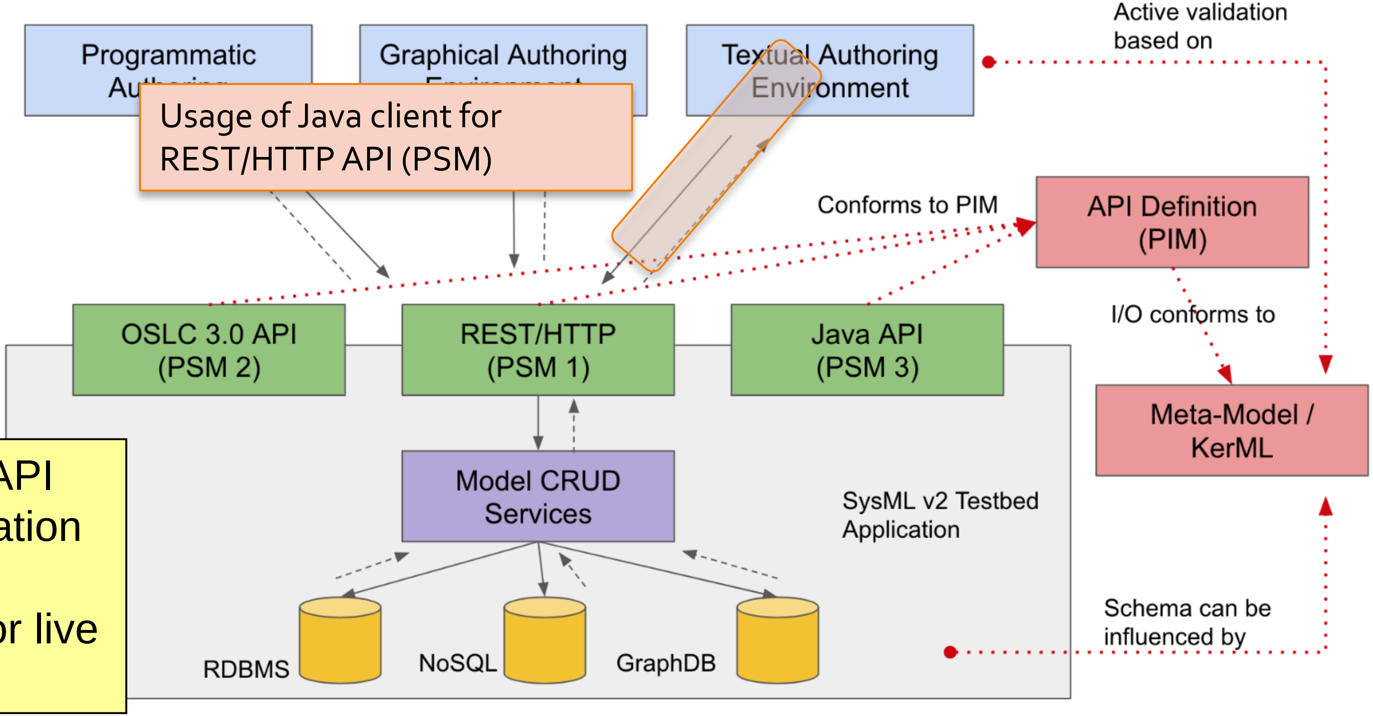
Units are identified on the *value*, *not* the type.

- Full ISO/IEC 80000 semantic model
- US Customary Units as well
- Implemented as extensible model library
- Automated conversion between any compatible units/scales
- Build on SysML v1 QUDV
- Compatible data model with ECSS E-TM-10-23 & E-TM-10-25, EGS-CC
- Quantity value definition will be extended in 2020-Q1 to probabilistic values with probability distributions, uncertainties, etc.

Pilot Implementation

SST

High-Level Architecture of SysML v2 Testbed



Usage of Java client for REST/HTTP API (PSM)

- Prototype API implementation
- REST API available for live testing

Summary

- SysML v2 on a very promising track
- First SysML v2 Public Incremental Release 2019-09 made 11 Oct 2019
  - See http://openmbee.org/sysml-v2-release/2019-09
  - Running textual prototype on Jupyter Notebook with REST API
    and Tom Sawyer auto-layout graphical language visualisation in web-browser
- Many serious improvements
  - Properly based on formal semantics – including mapping to OWL/RDF
  - Usage-focused modelling & unification of structure and behavior: much more SE-friendly
  - Standardized technology-independent API – Can also be used by non-SysML tools
  - Substantial European influence (Experience from ECSS, RangeDB, Capella)
- Will take another ~1.5 years before becoming available in tools