Extending SysML for Integration with Solver-based Simulation Tools

Ion Matei
Conrad Bock
Overview

- Motivation and approach
- Dynamic simulation overview
- SysML extension
- Detailed example
- Transforming to simulation formats
- Summary
Overview

- Motivation and approach
- Dynamic simulation overview
- SysML extension
- Detailed example
- Transforming to simulation formats
- Summary
Model-based Systems Engineering

Enabled by integrated models of:

- Requirements
- Accelerate at a rate of 4 m/s²
- Analysis and testing
- Designs
- 100 kw
- Hydraulic pressure
- Mechanical pressure
Modeling Languages

Needed for people / computers to share models.

Diagrams and/or text

- Graphics:
  - Circles,
  - Rectangles
  - Lines

- Domain terms:
  - Lathes, Feeders
  - Drying, Shaping

- What happens:
  - Geometry changed.
  - Pieces mounted onto machines.
  - Water removed.

Text:
- Reserved words
Systems Modeling Language (SysML)

- Most widely used graphical modeling language for systems engineering.
- Open standard published by the Object Management Group (OMG).
- Initiated by the International Council on Systems Engineering (INCOSE).
- First published in 2007, most recent update in 2012.
- Adopted by practically all commercial and open source SE modeling tools.
SysML Diagrams

**Requirements**

 req [Package] Vehicle Specifications [ Braking Requirements ]

Vehicle System 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."

Braking Subsystem Specification

<<requirement>>

Anti-Lock Performance

Id = "33.7"
Text = "The braking system shall prevent wheel lockup under all braking conditions."

<<deriveReq>>

**Components**

 bdd [Package] Structure [ ABS Structure Hierarchy ]

<<block>>

Library:
Electronic Processor

<<block>>

Anti-Lock Controller

ibd [Block] Anti-Lock Controller [ Basic ]

<<block>>

Traction Detector

d1: Traction Detector

c2: Traction Modulator

m1: Brake Modulator

**Behavior**

sd ABS_ActivationSequence [Sequence Diagram]

<<block>>

ABS_ActivationSequence [Sequence Diagram]

stm TireTraction [State Diagram]

<<block>>

TireTraction [State Diagram]

act PreventLockup [Activity Diagram]

:t: DetectLossOfTract

:: ModulateBrakingForce

par [Block] Straight Line Vehicle Dynamics [ Parameters ]

e1: Braking Force Equation

\[ f = (f^2 + 1 - f) \]

\[ m = Kg \]

e2: Acceleration Equation

\[ a = m/sec^2 \]

\[ f = m^a \]

e3: Velocity Equation

\[ v = m/sec \]

\[ a = dv/dt \]

\[ t = sec \]
SysML extends the Unified Modeling Language (UML)

- UML is the most widely used graphical modeling language for software (also published by OMG).

- INCOSE chose to extend UML (and approach OMG) because
  - Modern systems/products usually have significant amounts of software in them.
  - Extending UML is a path to integrating engineering and software development.
  - Software modeling in UML has many commonalities with systems engineering modeling.
SysML/UML Diagrams

SysML (structure)
- Block Definition Diagram
- Requirements Diagram
- Internal Block Diagram
- Parametric Diagram
- Package Diagram

UML
- Class Diagram
- Internal Structure Diagram
- Package Diagram

SysML/UML (behavior)
- Activity Diagram
- State Machine Diagram
- Use Case Diagram
- Interaction Diagram

As-is from UML
Extension of UML
Focus of this talk is integration with solver-based simulation.
Solver-based Simulators

Solver-based simulators have user interfaces similar to modeling tools.

But the tools treat these as equations rather than physical things.
Solver-based Simulators

Generate differential equations from diagrams and incrementally solve them to give values of variables over time.
Integration supported by different profiles for each simulator.
Reduce Specialized Profiles

Extend SysML with a general simulation profile.
Overview

Motivation and approach

Dynamic simulation overview

SysML extension

Detailed example

Transforming to simulation formats

Summary
Multiple Engineering Disciplines

- Generally, solvers use the same numerical algorithms for all the engineering disciplines.
Multiple Engineering Disciplines

Possible because of commonality of underlying physics.

<table>
<thead>
<tr>
<th>Domain</th>
<th>What is flowing</th>
<th>Flow rate</th>
<th>Potential to flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical</td>
<td>Charge</td>
<td>Current</td>
<td>Voltage</td>
</tr>
<tr>
<td>Mechanics, translational</td>
<td>Momentum</td>
<td>Force</td>
<td>Velocity</td>
</tr>
<tr>
<td>Mechanics, angular</td>
<td>Angular momentum</td>
<td>Torque</td>
<td>Angular velocity</td>
</tr>
<tr>
<td>Hydraulics</td>
<td>Volume (uncompressable fluid)</td>
<td>Volumetric rate</td>
<td>Pressure</td>
</tr>
<tr>
<td>Thermal</td>
<td>Heat energy</td>
<td>Heat flow rate</td>
<td>Temperature</td>
</tr>
</tbody>
</table>
Conservation Laws

Rates of flow follow conservation laws, potentials to flow do not.

\[ \text{FR out 1} + \text{FR out 2} = \text{FR in} \]

Potential to flow is the same on all ends.
Conservation Laws

- Flow rates can be in either direction (positive or negative).

\[ \text{Flow rate, out/in 1} + \text{Flow rate, out/in 2} = \text{Flow rate, in/out} \]

- Potential to flow is the same on all ends.
Simulating Information Flow

Information flow does not follow conservation laws

- Information can be copied.
- Simulated as potential to flow (signals).

Information is the same on all ends.
Simulator Constraints

β Rates of flow cannot be simulated on unidirectional flows.

β Flow rate, out 1 + Flow rate, out 2 = Flow rate, in

β Potential to flow is the same on all ends.
Simulator Constraints

- Unidirectional flows cannot be merged.

- They can be split (reverse of above).

- Bidirectional flows can be merged and split.
Overview

- Motivation and approach
- Dynamic simulation overview
- SysML extension
- Detailed example
- Transforming to simulation formats
- Summary
Systems engineering models and simulators are concerned with overlapping aspects of flow.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Kind of item flowing</th>
<th>Flow Rate</th>
<th>Potential to flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical</td>
<td>Charge</td>
<td>Current</td>
<td>Voltage</td>
</tr>
<tr>
<td>Mechanics, translational</td>
<td>Momentum</td>
<td>Force</td>
<td>Velocity</td>
</tr>
<tr>
<td>Mechanics, angular</td>
<td>Angular momentum</td>
<td>Torque</td>
<td>Angular velocity</td>
</tr>
<tr>
<td>Fluid</td>
<td>Volume (uncompressable fluid)</td>
<td>Volumetric rate</td>
<td>Pressure</td>
</tr>
<tr>
<td>Thermal</td>
<td>Heat energy</td>
<td>Heat flow rate</td>
<td>Temperature</td>
</tr>
</tbody>
</table>
Specify what is flowing and in which direction.

- **Crankshaft**
  - Out produces: Energy

- **Hub**
  - In accepts: Energy

**Direction of flow**: Engine → Crankshaft

**Kind of item flowing**: Hub → Wheel
Bring flows and potentials into SysML for generating simulator input.

```
out produces : Energy
  «simProperty» {referTo: produces}
  simProduces : EnergyFlow

in accepts : Energy
  «simProperty» {referTo: accepts}
  simAccepts : EnergyFlow
```

```
«simBlock»

EnergyFlow

«simVariable» {isConserved} energyFlowRate : Power
«simVariable» energyPotential : PotentialEnergy
```
Stereotypes

{ property is typed by Class with SimBlock applied }

isContinuous : Boolean = true
isConserved: Boolean = false
changeCycle: Real = 0
Conservation and Directionality

- SimBlocks for unidirectional flow properties \((in\ or\ out)\) can only have non-conserved variables (isConserved = false).

- Simblocks for bidirectional flow properties can have both conserved and non-conserved variables.
Connection Constraints

- SimBlocks of matching flow properties must either be the same or match exactly.
- In flow properties can be connected to no more than out flow property.
- Out flow properties can be connected to any number of in flow properties.
- Inout flow properties aren’t constrained in linkage number.
Overview

- Motivation and approach
- Dynamic simulation overview
- SysML extension
- Detailed example
- Transforming to simulation formats
- Summary
N-ary electrical connections broken into binary SysML connectors.
Example (Extensions)

```
start here

simBlock

sim variables

{isConserved} i  : Current
v : Voltage

flow properties

inout electricity: Charge

sim properties

{referTo=electricity} sb: ElectricityFlow

sim constants

R : Resistance= 10
C : Capacitance= 0.01
L : Inductance= 0.1

sim Block

ElectricityFlow

sim variables

{isConserved} i : Current
v : Voltage

sim constants

R : Resistance= 10
C : Capacitance= 0.01
L : Inductance= 0.1

Start here
```
Example (Constraint Blocks)

```
«block»
TwoPinElectricalComponent

ports
p : Pin
n : Pin

sim variables
«isConserved» iThru : Current
vDrop : Voltage

«block»
Resistor

sim constants
R : Resistance= 10

constraints
rc : ResistorConstraint
```
Specifying mathematical equations.
Overview

- Motivation and approach
- Dynamic simulation overview
- SysML extension
- Detailed example
- Transforming to simulation formats
- Summary
## Terminology Mapping

<table>
<thead>
<tr>
<th>SysML+</th>
<th>Modelica</th>
<th>Simulink / Simscape</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block without internal structure</td>
<td>Model without connections</td>
<td>BlockType / Component</td>
</tr>
<tr>
<td>Block with internal structure</td>
<td>Model with connections</td>
<td>System / Component</td>
</tr>
<tr>
<td><strong>SimBlock</strong></td>
<td><strong>Connector</strong></td>
<td>Library elements</td>
</tr>
<tr>
<td>(referring to a flow prop)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variables</td>
<td>Variables</td>
<td>Ports / Variables</td>
</tr>
<tr>
<td>On SimBlocks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td>Connection/Equation</td>
<td>Line/Connection</td>
</tr>
<tr>
<td>Constraint block</td>
<td>Equation</td>
<td>S-Function / Equation</td>
</tr>
</tbody>
</table>
Mapping Internal Structure to Modelica

model circuit
  Resistor r(R=10);
  Capacitor c(C=0.01);
  Inductor i(L=0.1);
  Source s;
  Ground g;

equation
  connect (s.p, c.n);
  connect (c.n, r.p);
  connect (r.n, i.p);
  connect (i.n, c.p);
  connect (c.p, s.n);
  connect (s.n, g.p);
end circuit;

SysML  Modelica
Mapping Internal Structure to Simscape

component circuit
components
  r = Resistor(R=10);
  c = Capacitor(C=0.01);
  i = Inductor(L=0.1);
  S = Source;
  G = Ground;
end
connections
  connect (s.p, c.n);
  connect (c.n, r.p);
  connect (r.n, i.p);
  connect (i.n, c.p);
  connect (c.p, s.n);
  connect (s.n, g.p);
end
end

SysML

Simscape
Mapping Constraints to Modelica

model Resistor "Electrical resistor"
  Pin p,n;
  flow Current iThru;
  Voltage vDrop;
  parameter Real R(unit="Ohm")
    "Resistance";
  equation
    vDrop = p.v - n.v;
    0 = p.i + n.i;
    iThru = p.i;
    R*iThru = vDrop;
end Resistor
component "Electrical Resistor"

nodes
  p = foundation.electrical.electrical;
  n = foundation.electrical.electrical;
end

variables
  iThru = { 0, 'A' };  // Current
  vDrop = { 0, 'V' };  // Voltage
end

parameters
  R = { 1, 'Ohm' };  // Resistance
end

function setup
  across( vDrop, p.v, n.v );
  through( iThru, p.i, n.i );
end

equations
  R*iThru == vDrop;
end
Pin in simulator has properties of SimBlocks

- Flow properties used only to determine direction ("causality") in usages of SimBlocks.

connector Pin
  flow Current i;
  Voltage v;
end EF;
Mapping SimBlocks to Simscape

We can’t find how these elements are specified, but they are referred to in user models, see later slides.

Pin in simulator is only the SimBlock
- Flow properties used only to determine direction (“causality”) in usage of SimBlocks.
Overview

- Motivation and approach
- Dynamic simulation overview
- SysML extension
- Detailed example
- Transforming to simulation formats
- Summary
Summary

- Goal is to reduce size and complexity of simulator-specific profiles.
  - by reusing and extending SysML.

- SysML concerned with flow direction (input/output) and kind of things flowing.

- Simulators are concerned with flow direction, potential, and rate.

- Extend SysML with rate, potential, and other aspects of simulated flow.

- Use extended SysML to generate simulator-specific files.
More Information

An Analysis of Solver-Based Simulation Tools

- Survey of solver-based simulators
- (http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909924)

Modeling Methodologies and Simulation for Dynamical Systems

- Describes two ways simulators are used.
- (http://nvlpubs.nist.gov/nistpubs/ir/2012/NIST.IR.7875.pdf)

SysML Extension for Dynamical System Simulation Tools

- Covers a simulator-independent extension of SysML.
- (http://nvlpubs.nist.gov/nistpubs/ir/2012/NIST.IR.7888.pdf)