AN IMPROVED METHOD OF PHYSICAL INTERACTION AND SIGNAL FLOW MODELING FOR SYSTEMS ENGINEERING

Mehdi Dadfarnia[a]
Conrad Bock[a]
Raphael Barbau[b]

[a]National Institute of Standards & Technology
[b]Engisis, LLC

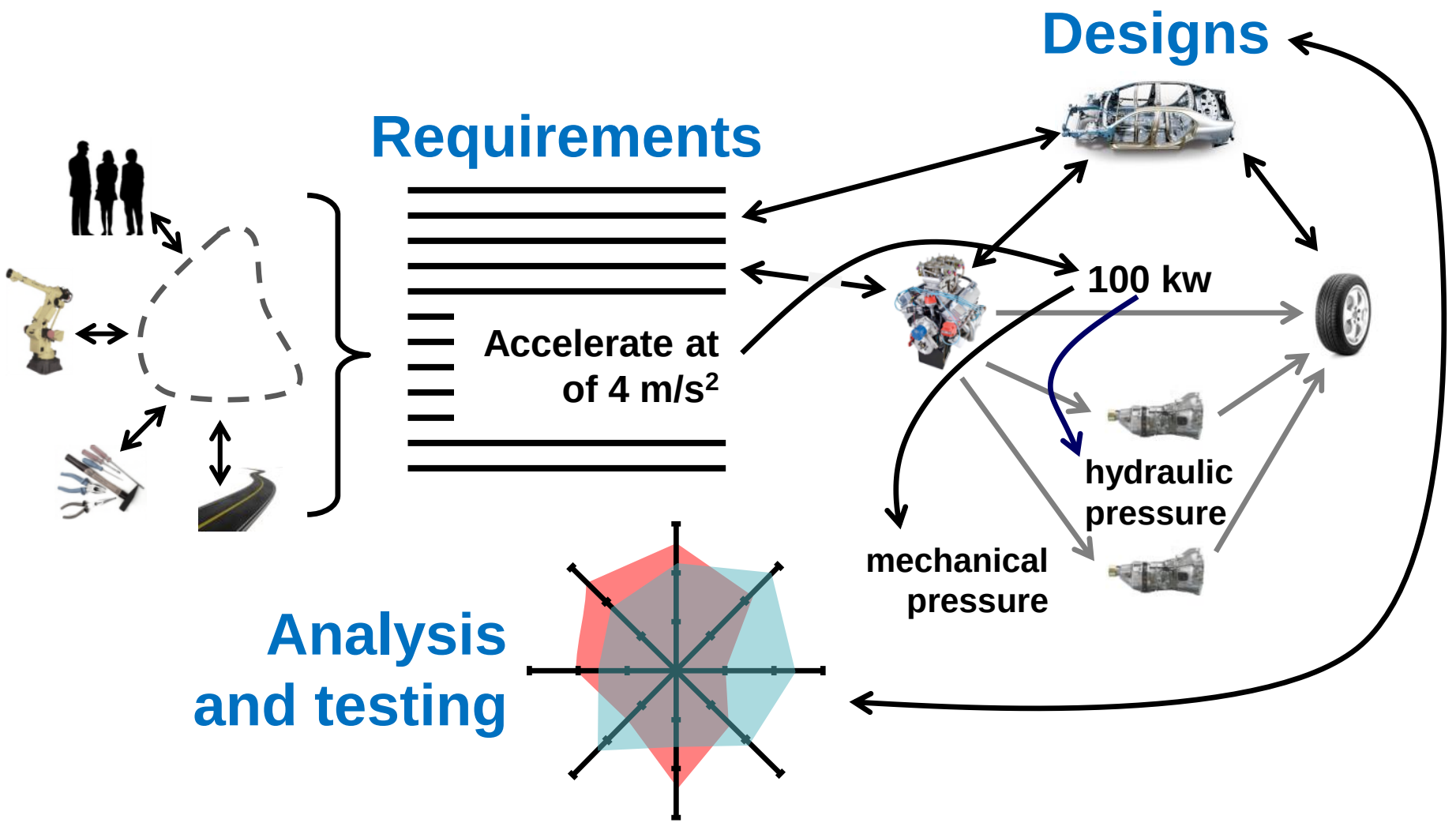
2/7/2017

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

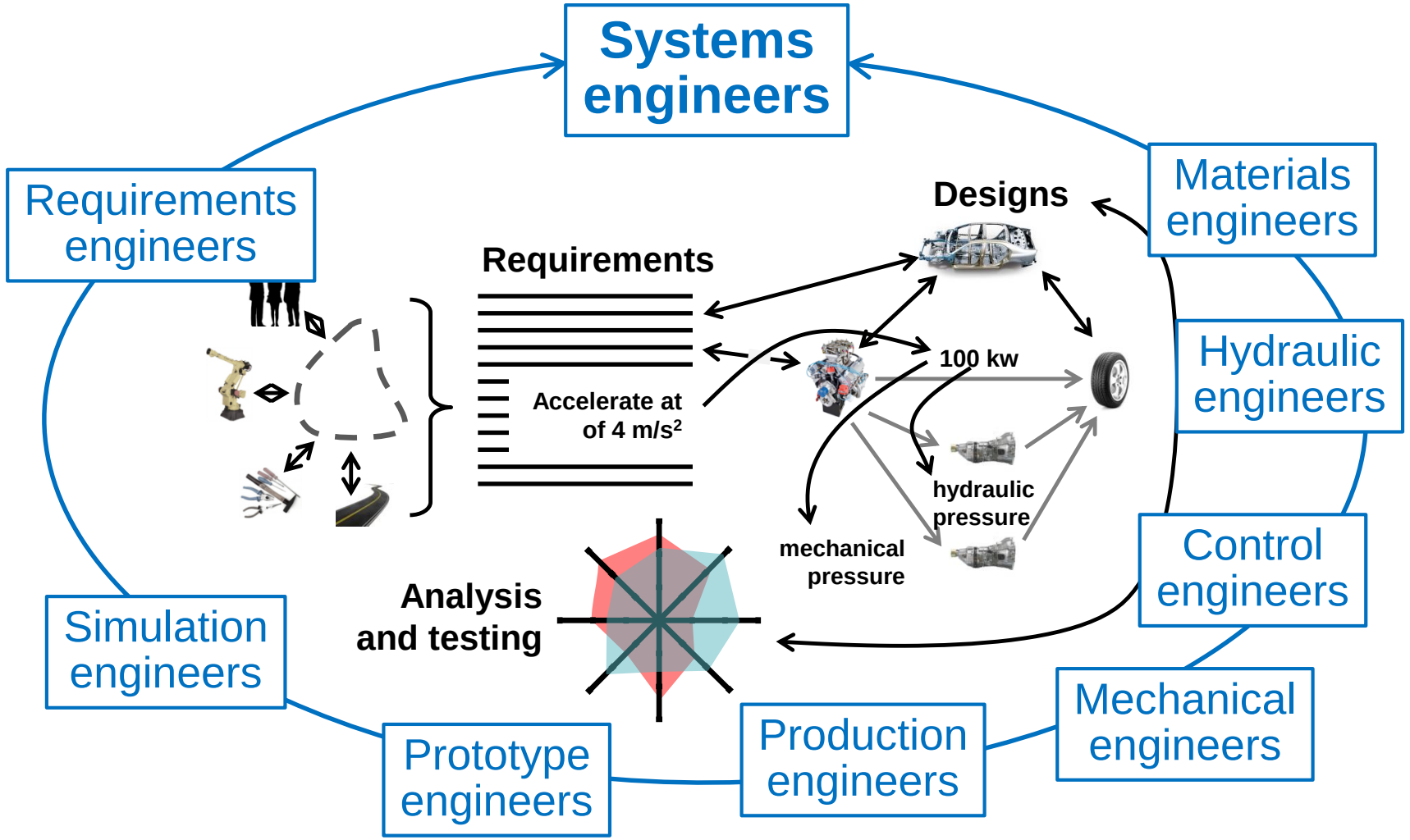
TABLE OF CONTENTS

TABLE OF CONTENTS

SYSTEMS ENGINEERING

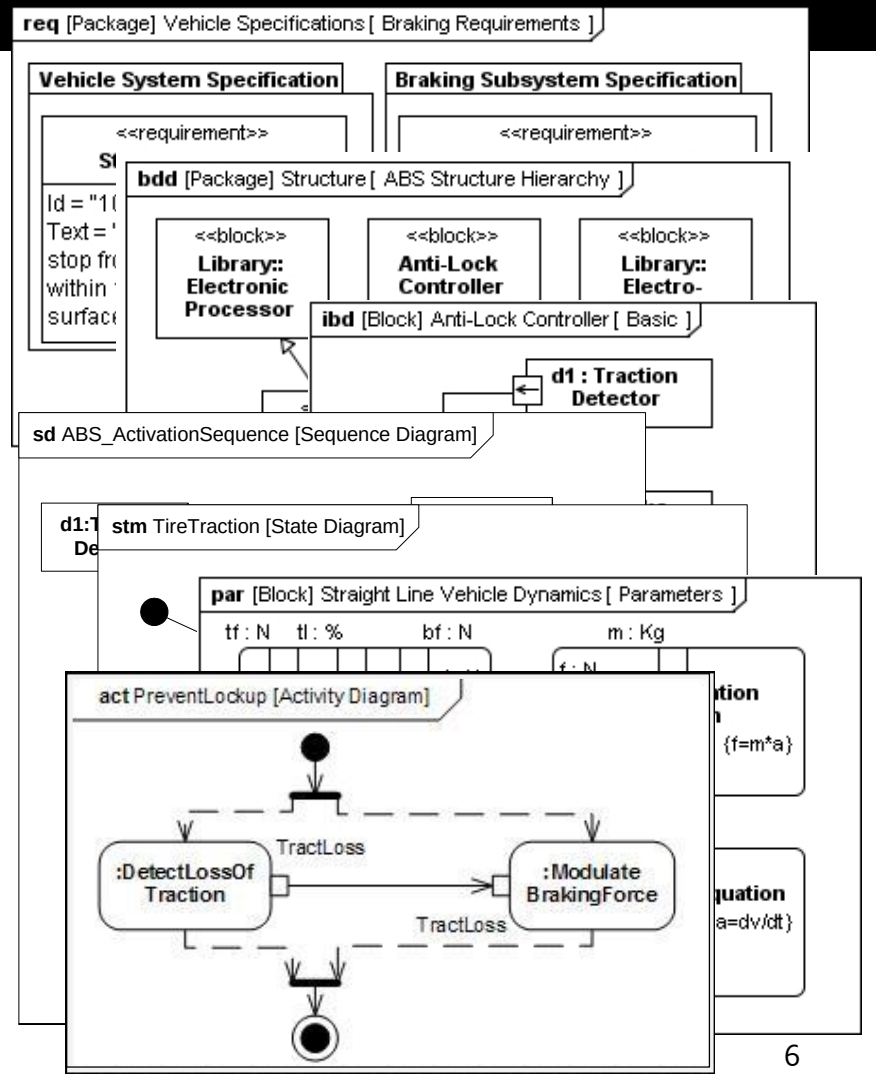


SYSTEMS ENGINEERS



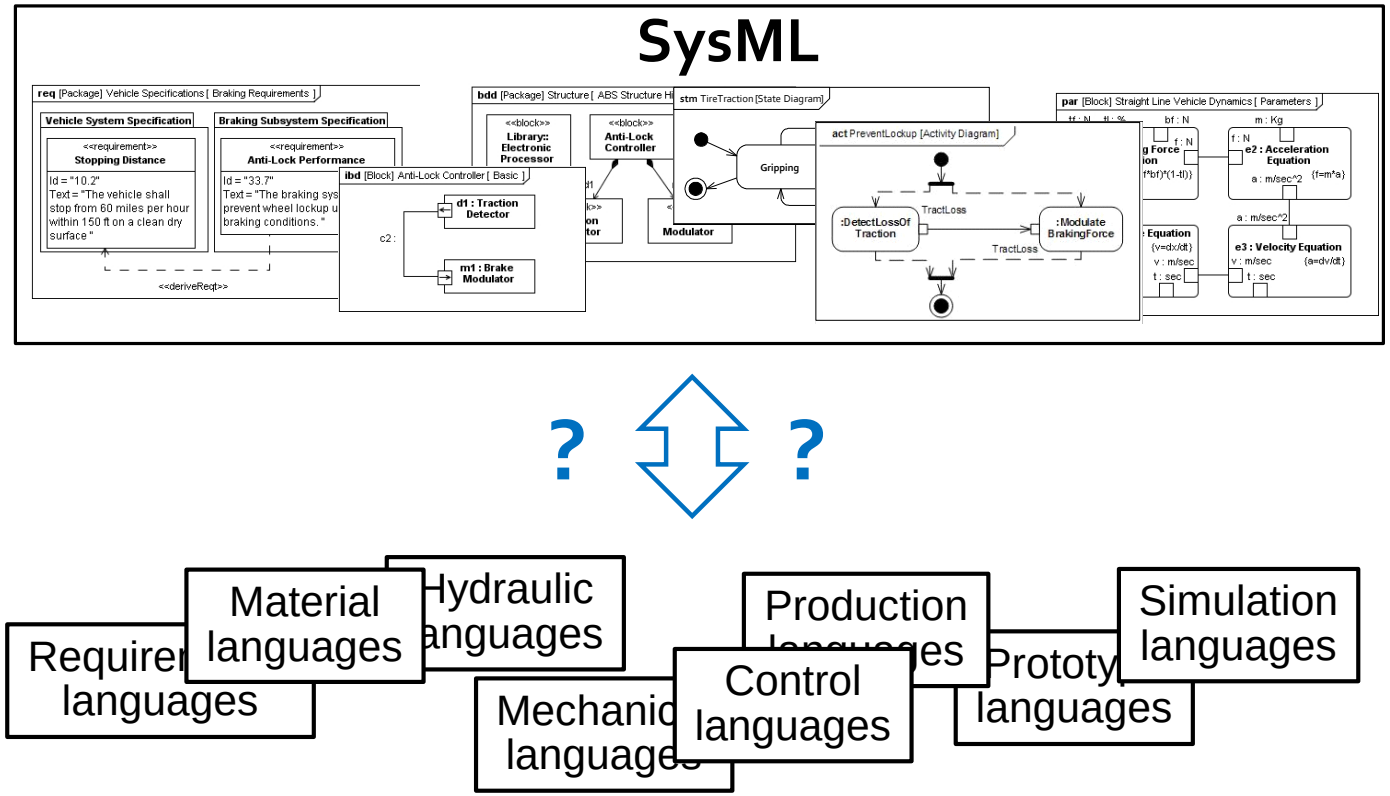
SYSTEMS MODELING LANGUAGE (SYSML)

- Most widely used graphical modeling language for systems engineering
  - International standard since 2007
  - Currently at revision 1.5
- Diagrams for:
  - Requirements, component breakdown and interconnection, behavior, parametrics



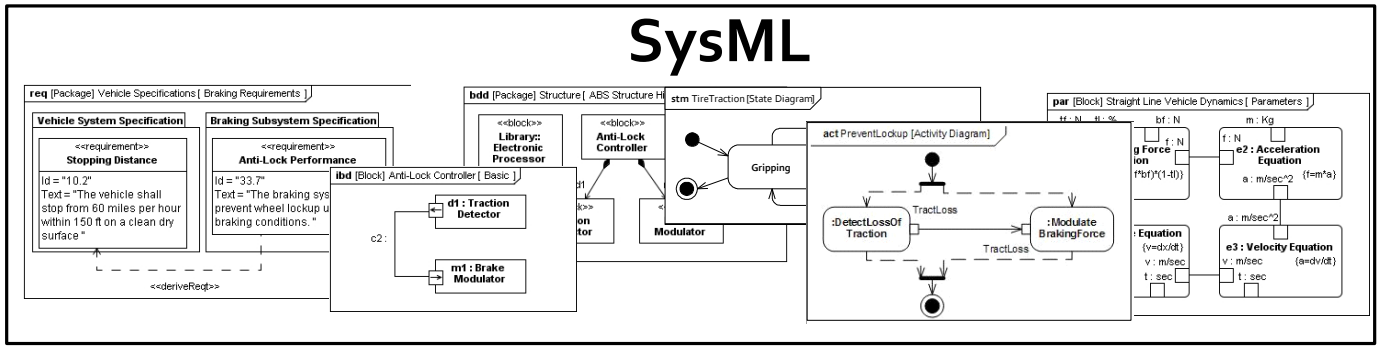
A PROBLEM

- Systems engineers and domain engineers:
  - Build (separate) models of physical and informational behaviors
  - Exchange models with each other
- Overlapping and inconsistent system specifications in multiple languages



A MORE SPECIFIC PROBLEM

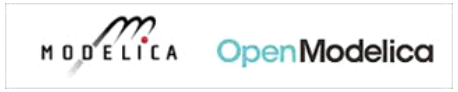
- Domain engineers often use physical interaction and signal flow analysis
  - Aka lumped parameter, 1-dimensional, or network analysis

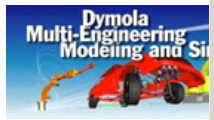


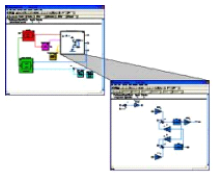
?

?

PI & SF simulation languages & tools

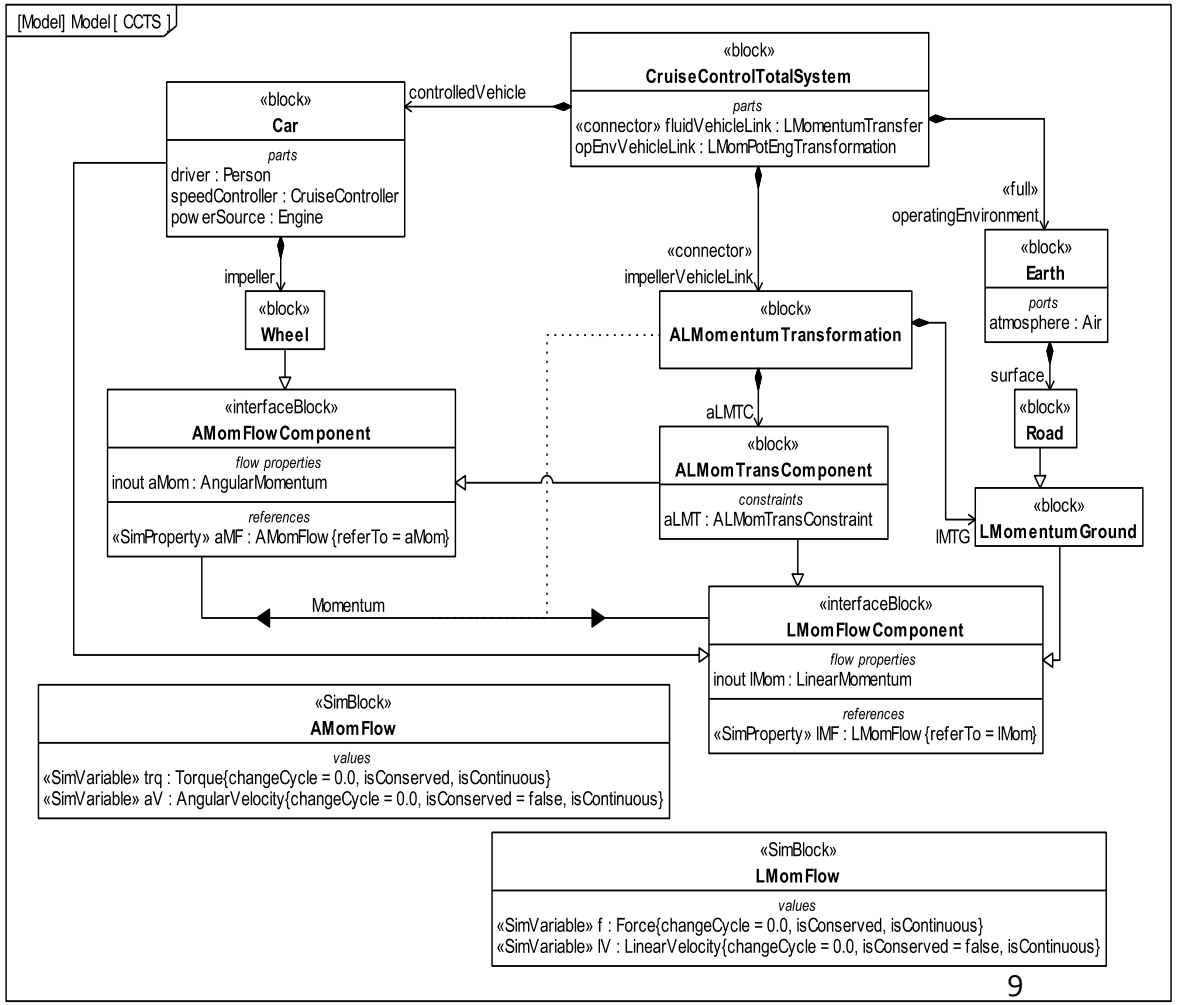






MODELING IN SYSTEMS ENGINEERING

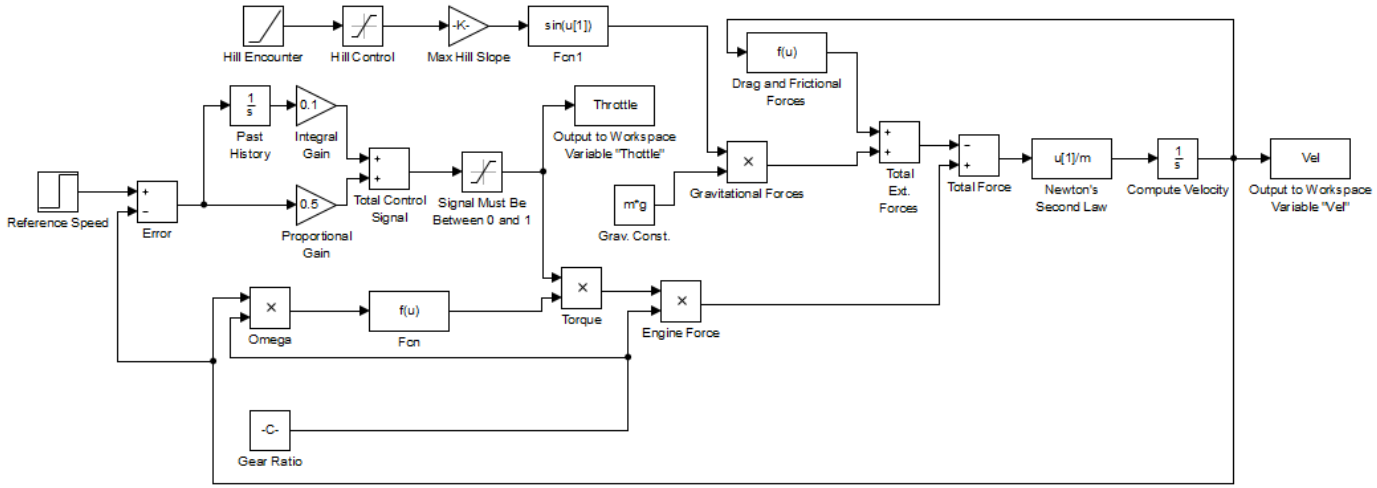
- Specifies breakdown of components into subcomponents
- Physical substances and information exchanged across component interconnections
- Lacks support for more detailed physical modeling



MODELING WITH EQUATIONS : UNIDIRECTIONALLY (SEQUENTIALLY)

- Sequential equation-based modeling
  - Lacks any structure
  - Physics obscured
- Unidirectional component modeling:
  - Lacks system structure
  - Physics still obscured

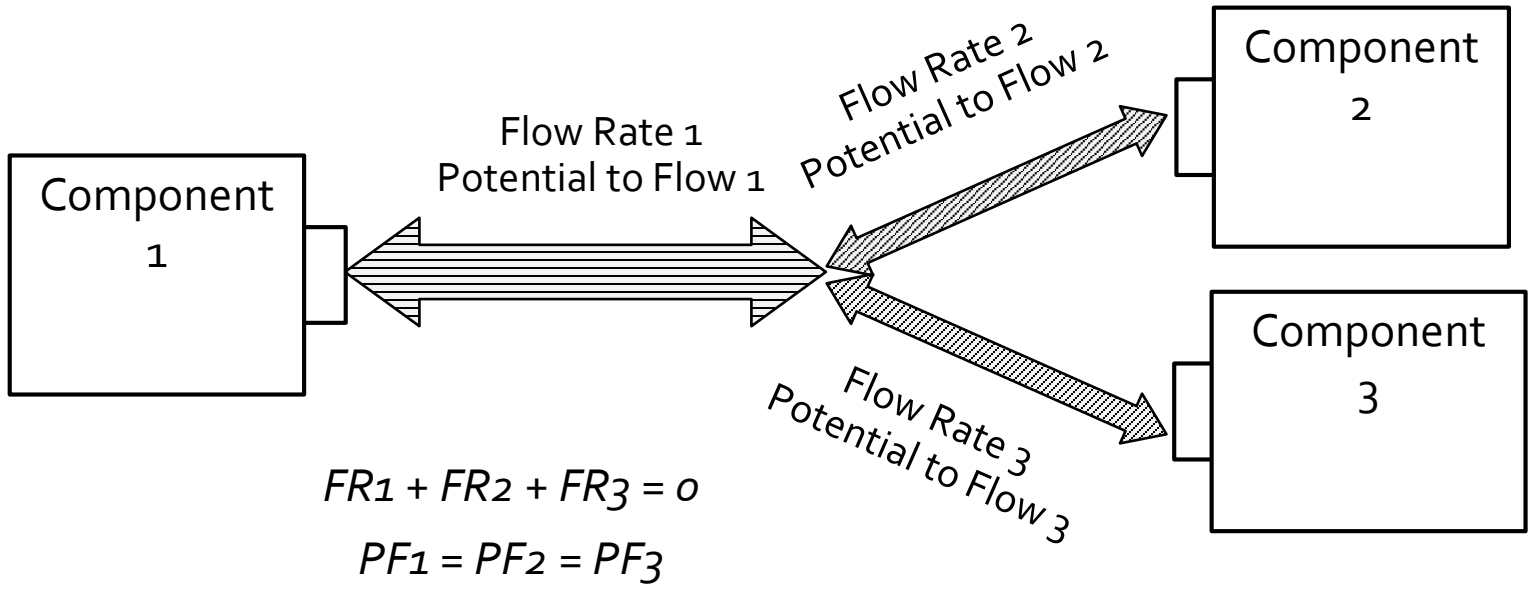
```
omega = alpha(gear) * v;
torque = u * Tm * ( 1 - beta * (omega/wm - 1)^2 );
F = alpha(gear) * torque;
Fr = m * g * Cr;
Fa = 0.5 * rho * Cd * A * v^2;
Fg = m * g * sin(theta);
Fd = Fr + Fa + Fg;
dv = (F - Fd) / m;
```



From Åström, K., Murray, R., "From Feedback Systems: An Introduction for Scientists and Engineers," 2nd edition, Princeton University Press, 2008. Models available at http://www.cds.caltech.edu/~murray/amwiki/index.php/Introduction.

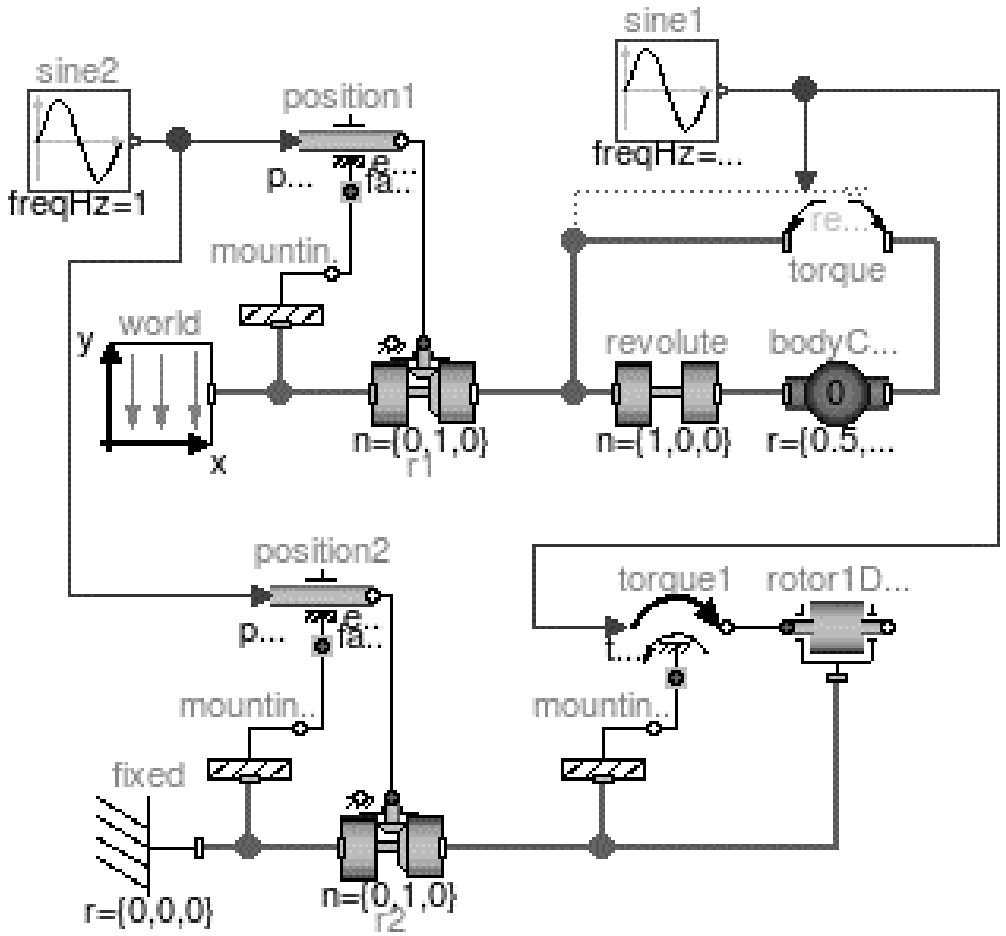
MODELING WITH EQUATIONS : BIDIRECTIONALLY

- Flow rate of substance is conserved (bidirectional)
- Substance's potential to flow is unchanged
- Flow rate x potential to flow = energy rate



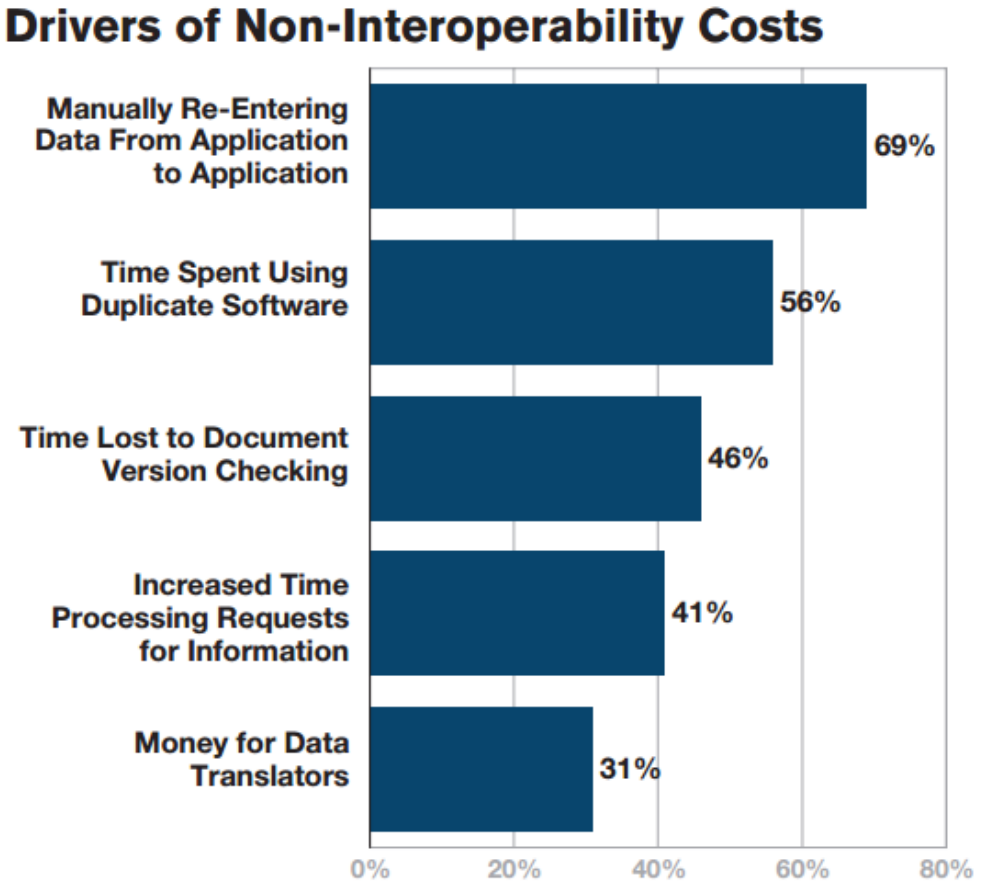
MODELING WITH EQUATIONS : BIDIRECTIONAL METHODS

- Lacks methods for aligning with systems engineering
  - Omit physical substances
  - Components interact by sharing variable values for energy exchange
  - Processes within components not addressed



EXCHANGING SYSTEMS ENGINEERING AND SIMULATION MODELS

- Result of differences between modeling systems engineering structures and equation-based behaviors:
  - Produces conflicting or erroneous models from misaligned structure/behavior
  - Discourages communication between engineers
  - Requires additional work to resolve differences



Source: McGraw-Hill Construction Research and Analytics, 2007

BRIDGING THE DIFFERENCE : FIND OVERLAPS AND DIFFERENCES

Domain	What is flowing
Electrical	Charge
Mechanics, translational	Linear momentum
Mechanics, angular	Angular momentum
Hydraulics	Volume
Thermal	Entropy

Direction of flow

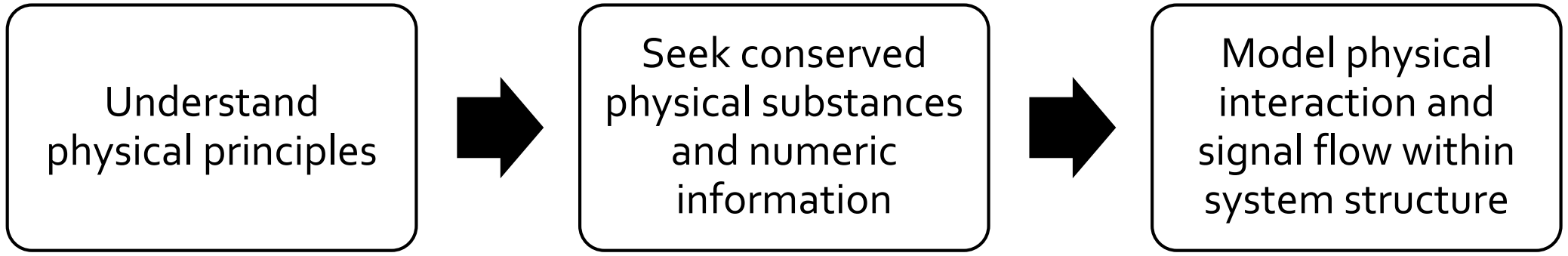
Flow rate	Potential to flow
Current	Voltage
Force	Linear velocity
Torque	Angular velocity
Volumetric rate	Pressure
Entropy flow rate	Temperature

Systems Engineering **Physical Simulation**

SOLUTION OVERVIEW

1. Interchange between systems tools and physical interaction and signal flow simulation tools
   a. Extension of SysML.
   b. Translation between extended SysML and simulation models.

2. Improved method for developing models:



SYSML FOR SIMULATION: REUSE EQUIVALENT CONCEPTS

Simulation Concepts		SysML
Models		Blocks with internal block diagram, but not a component of other blocks
Components	Atomic	Blocks without internal block diagram
	Subsystems	Blocks with internal block diagram
Links		Connectors
Ports		Ports with flow properties
Equations		Constraint blocks

Almost all concepts in simulation are available in SysML

SYSML FOR SIMULATION: EXTEND FOR MISSING CONCEPTS

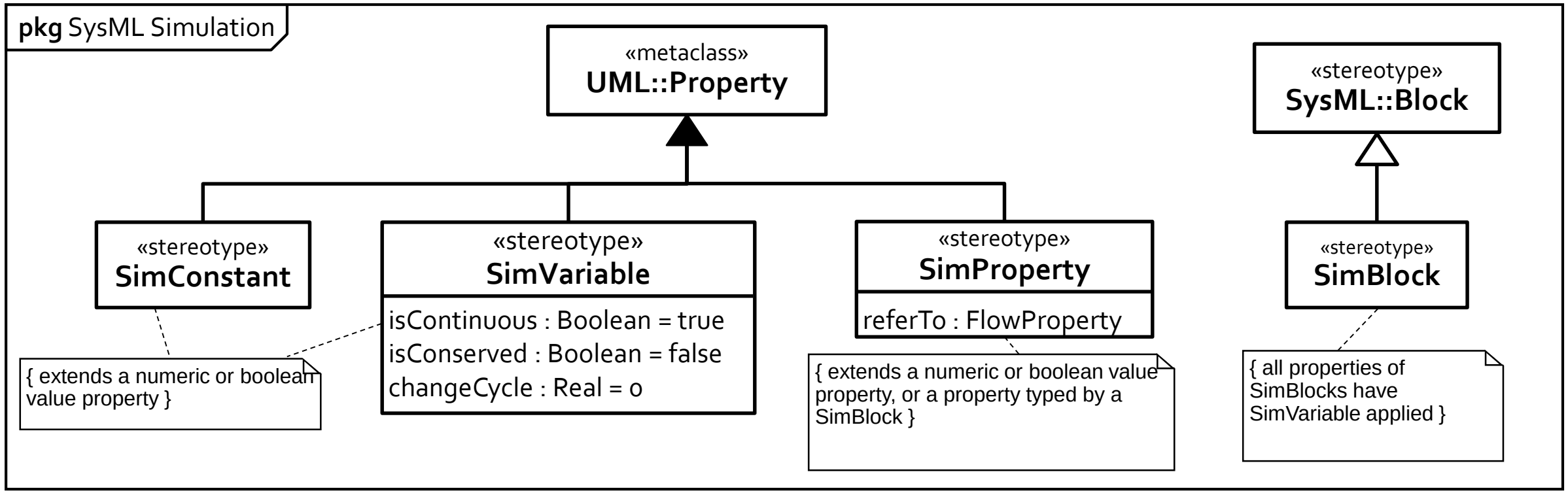
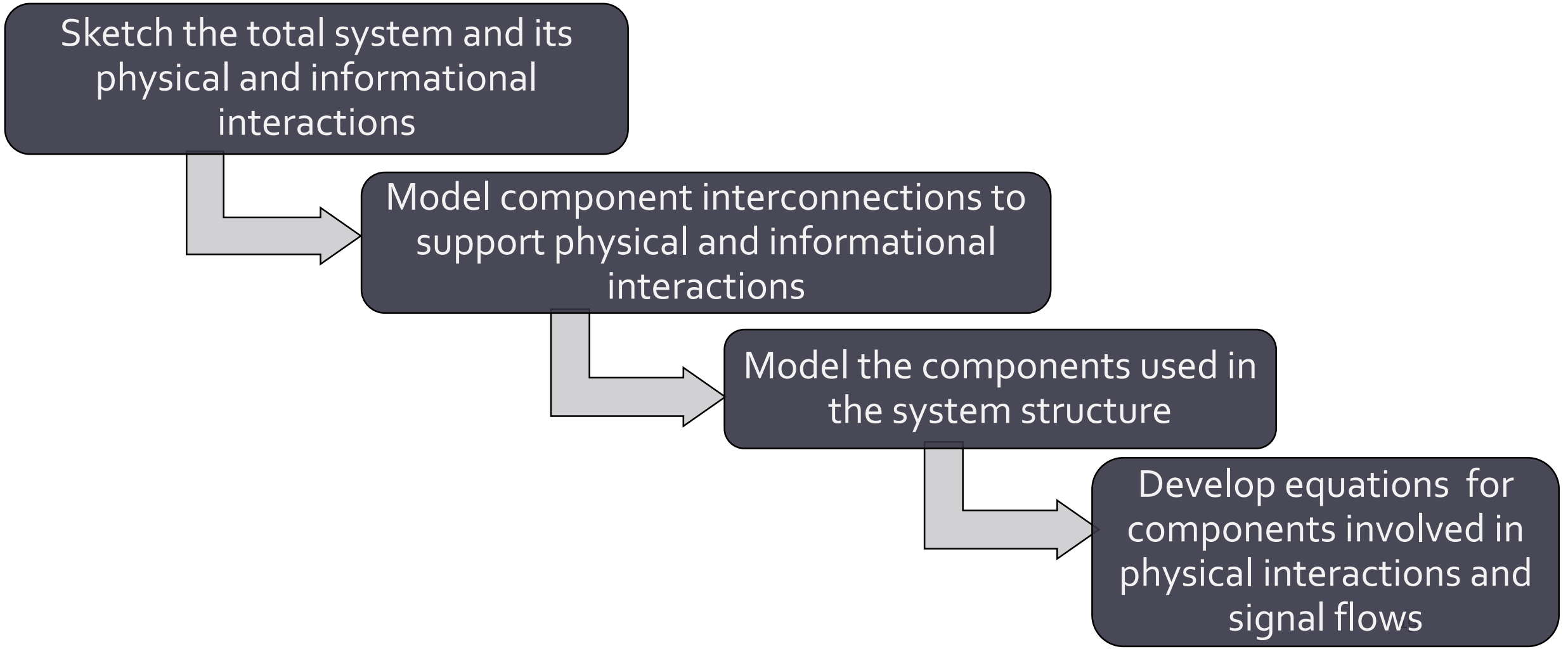


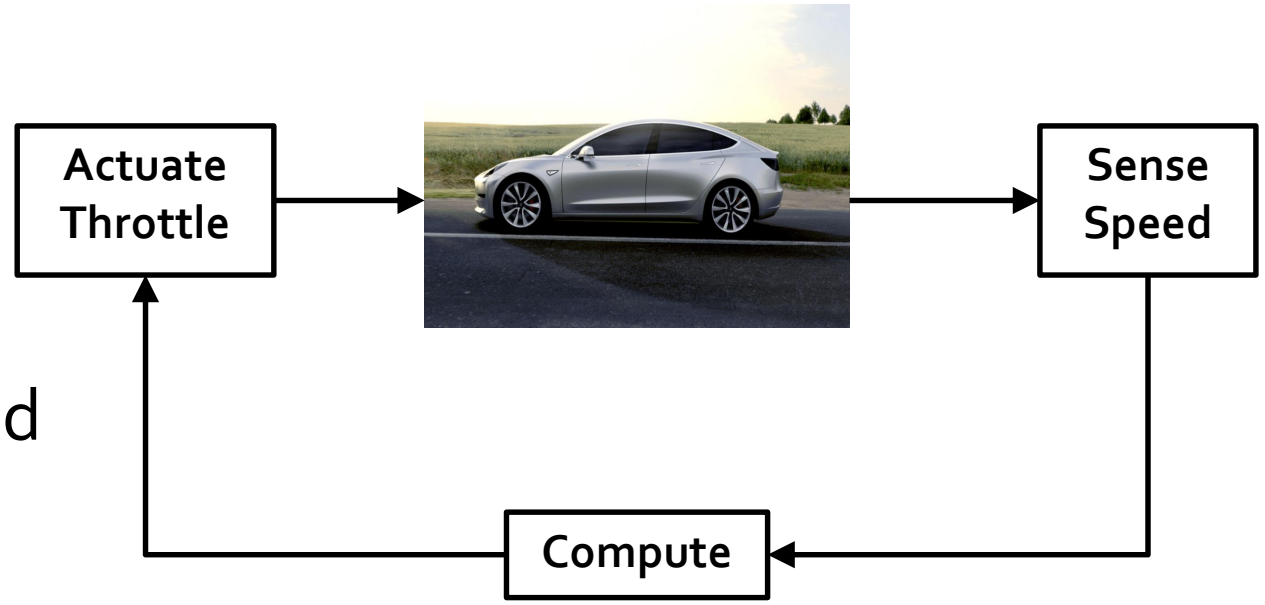
TABLE OF CONTENTS

METHOD STEPS



AN EXAMPLE

- Example: Automobile cruise control system
  - Commonly used in simulation modeling methods
  - Physical interactions and signal flows
- Attempt to keep constant vehicle speed (throttle) despite disturbances
  - Gravity (road slope)
  - Tire rolling resistance
  - Air effects

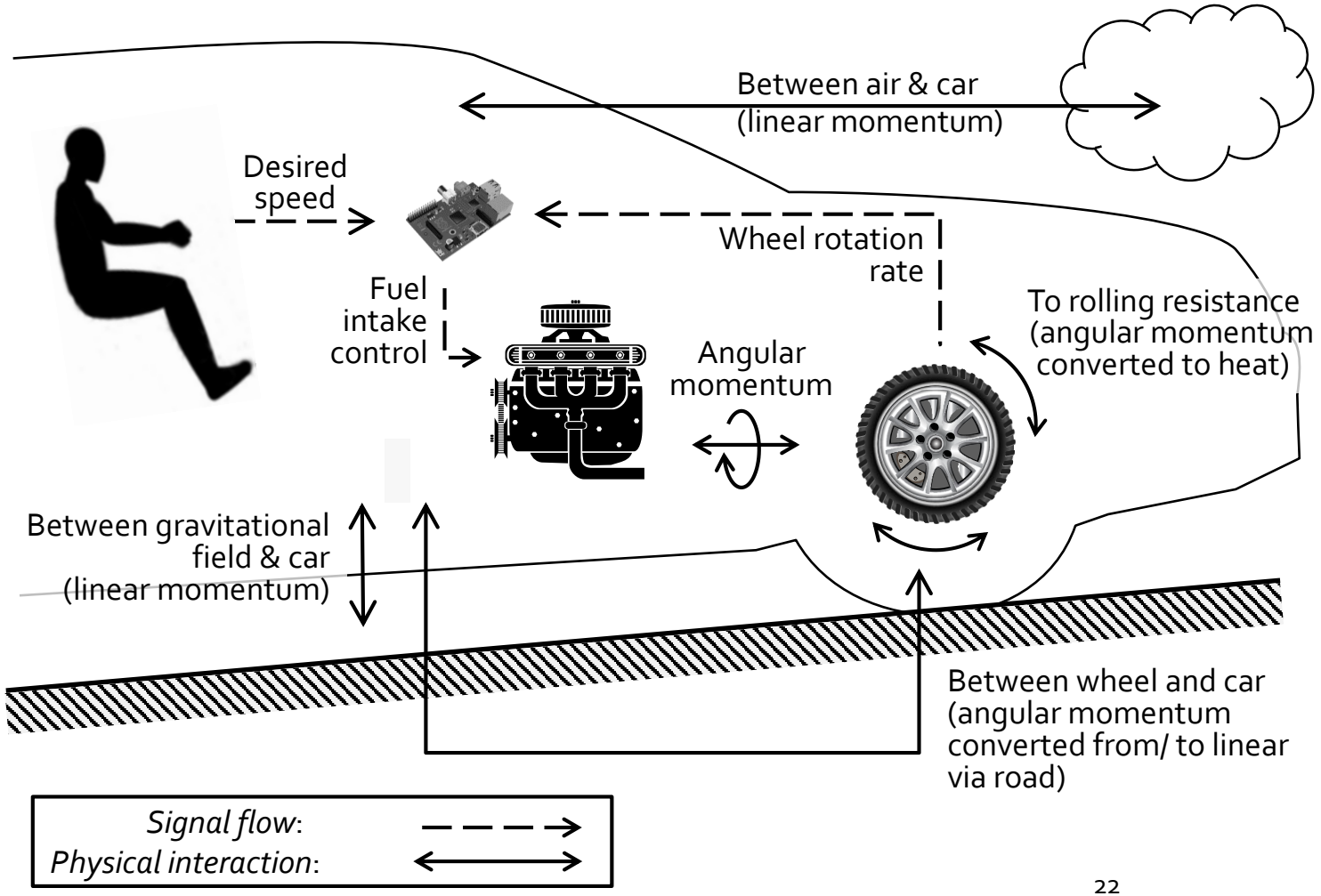


STEP 1: DEVELOP AN INFORMAL SKETCH OF THE TOTAL SYSTEM COMPONENTS

- Physical interactions between and within components
  - Transmission/transformation of conserved substances in system
  - Angular momentum, linear momentum, electric charge, etc
- Information sent between components
  - Communication of information from one part of system to another
  - Numeric information, such as control signals, etc

STEP 1: INFORMAL SKETCH EXAMPLE

- Total system for a cruise controller
- Physical interactions: Solid, bidirectional arrows
- Signal flows: dashed, unidirectional arrows

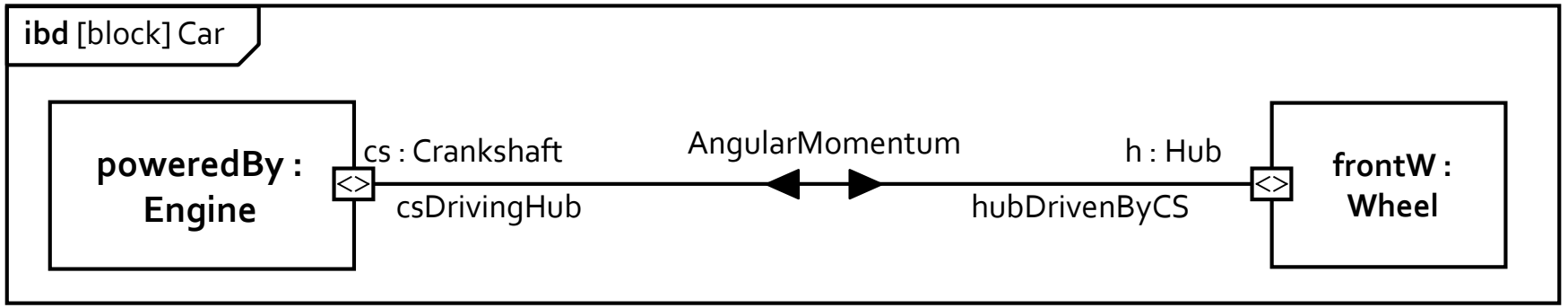


STEP 2.1: IDENTIFY PHYSICAL INTERACTIONS AND SIGNAL FLOWS

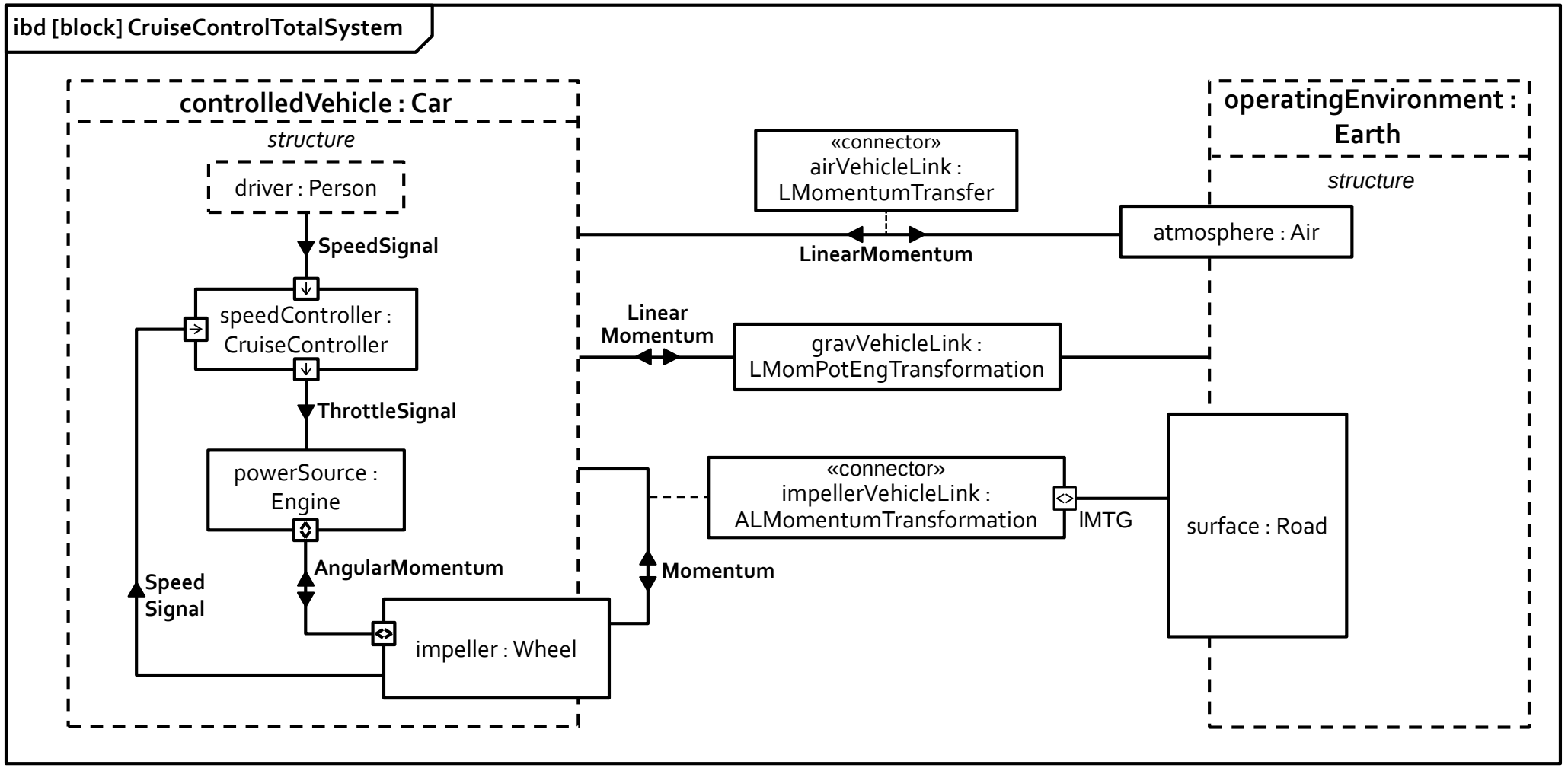
- Physical interactions:
  - Vehicular driving force: Transformation between angular momentum of wheels and linear momentum of vehicle
  - Air effects: Transfer of linear momentum between vehicle and air
  - Slope of road/gravity: Transformation between linear momentum and gravitational potential energy
- Signal flows:
  - Driver instructions: Sends desired speed to controller.
  - Controller commands: Sends throttle setting to engine.
  - Wheel sensor readings: Sends angular velocity to controller

STEP 2.2: DEFINE COMPONENT INTERCONNECTIONS

- Include all components, subcomponents, and interconnections between them
  - Interconnections reflect the sketch's physical interactions / signal flows
- Use SysML Internal Block Diagram
  - Roles (parts, ports): Played by kinds of things (blocks)
  - Connectors between roles

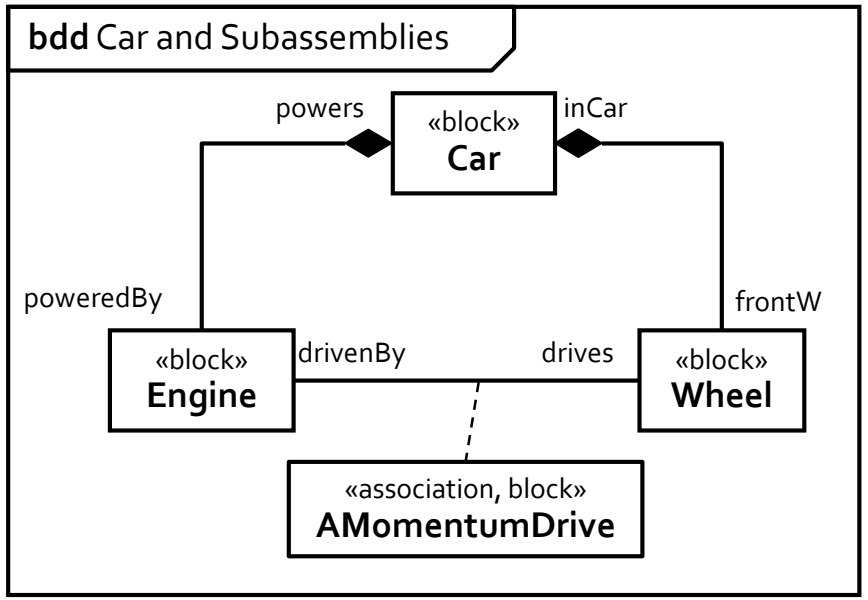


STEP 2: INTERNAL BLOCK DIAGRAM EXAMPLE

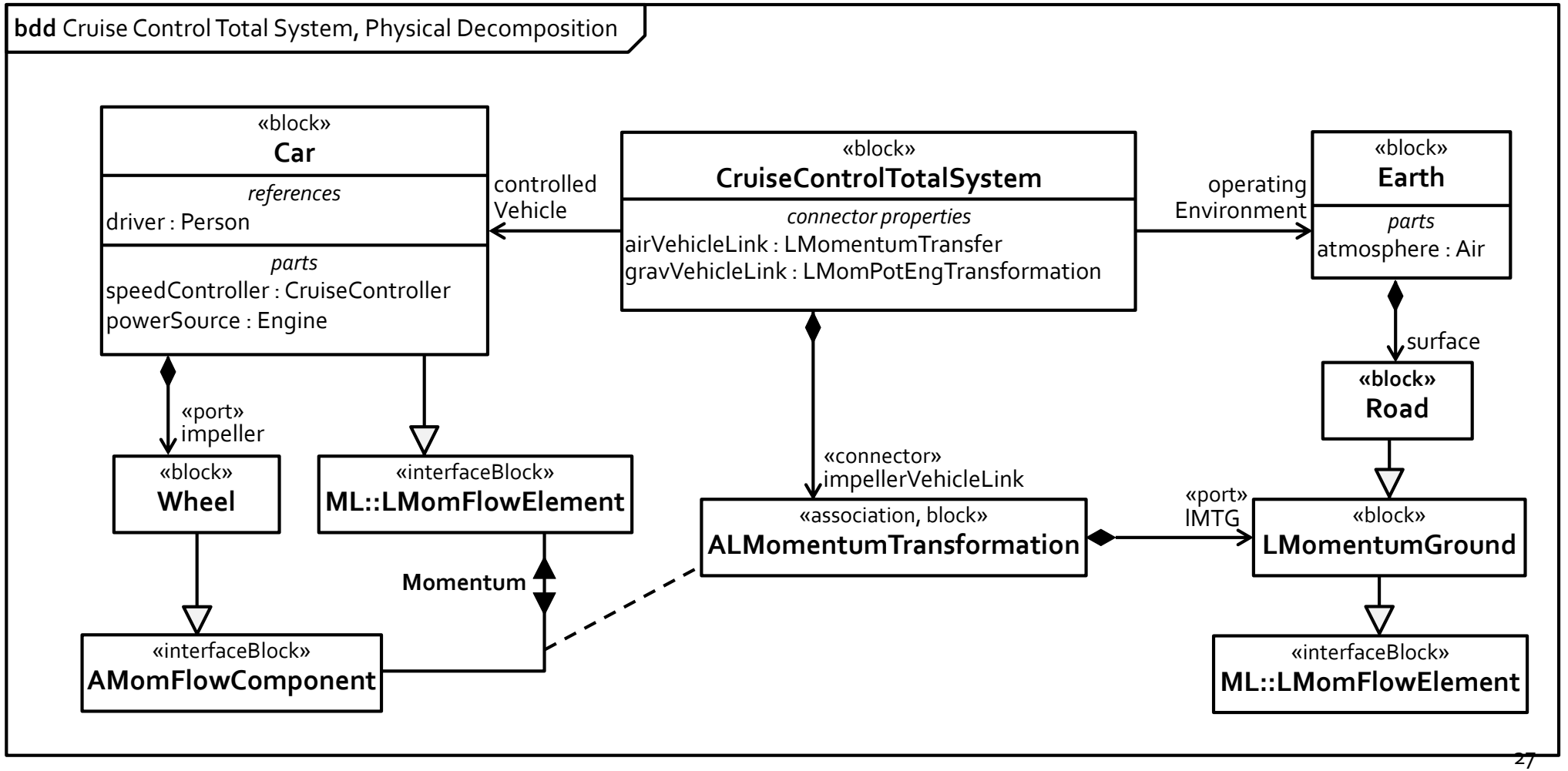


STEP 3: DEFINE COMPONENTS USED IN SYSTEM STRUCTURE

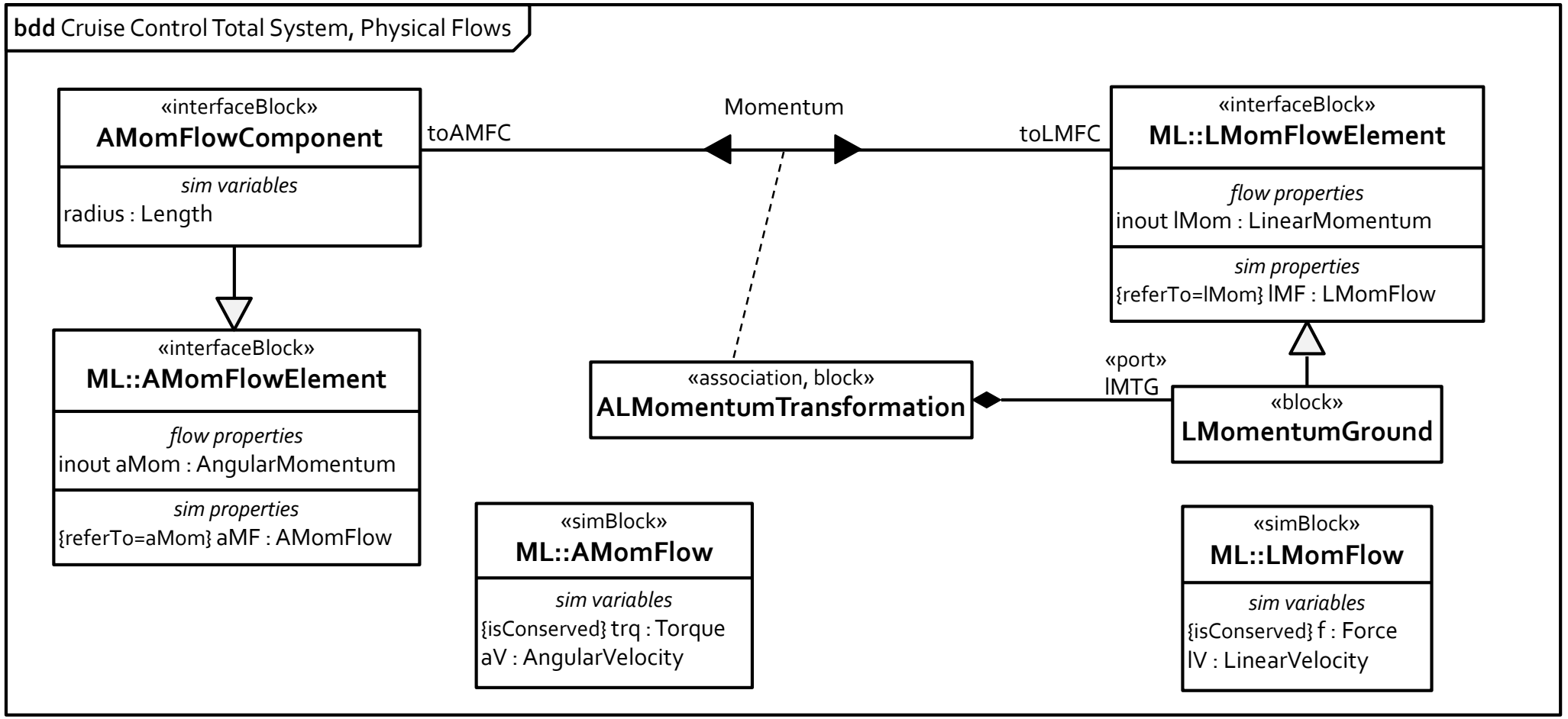
- Specifies the kinds of things playing parts and ports in Step 2
- Shows relationships between components that can be used to connect components in the system.
- Use SysML Block Definition Diagram



STEP 3: BLOCK DEFINITION DIAGRAM EXAMPLE, SYSTEM DECOMPOSITION

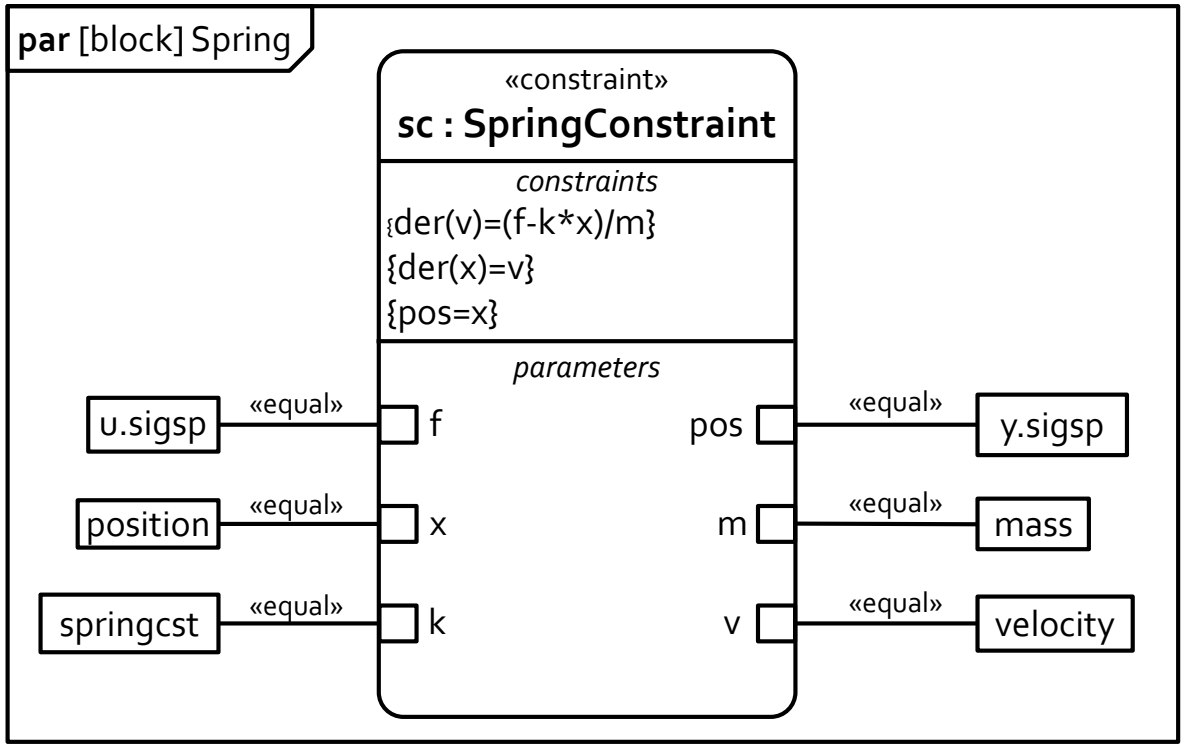


STEP 3: BLOCK DEFINITION EXAMPLE, PHYSICAL FLOWS



STEP 4: DEVELOP EQUATIONS FOR COMPONENTS

- Develop equations for components modeled in Step 3, based on interactions in Step 2
- Provides equations to generate code for simulators
- SysML Parametric Diagram
  - Bind component/association block properties to equations
  - One for every component / association block



STEP 4: COMPONENT EQUATION EXAMPLE

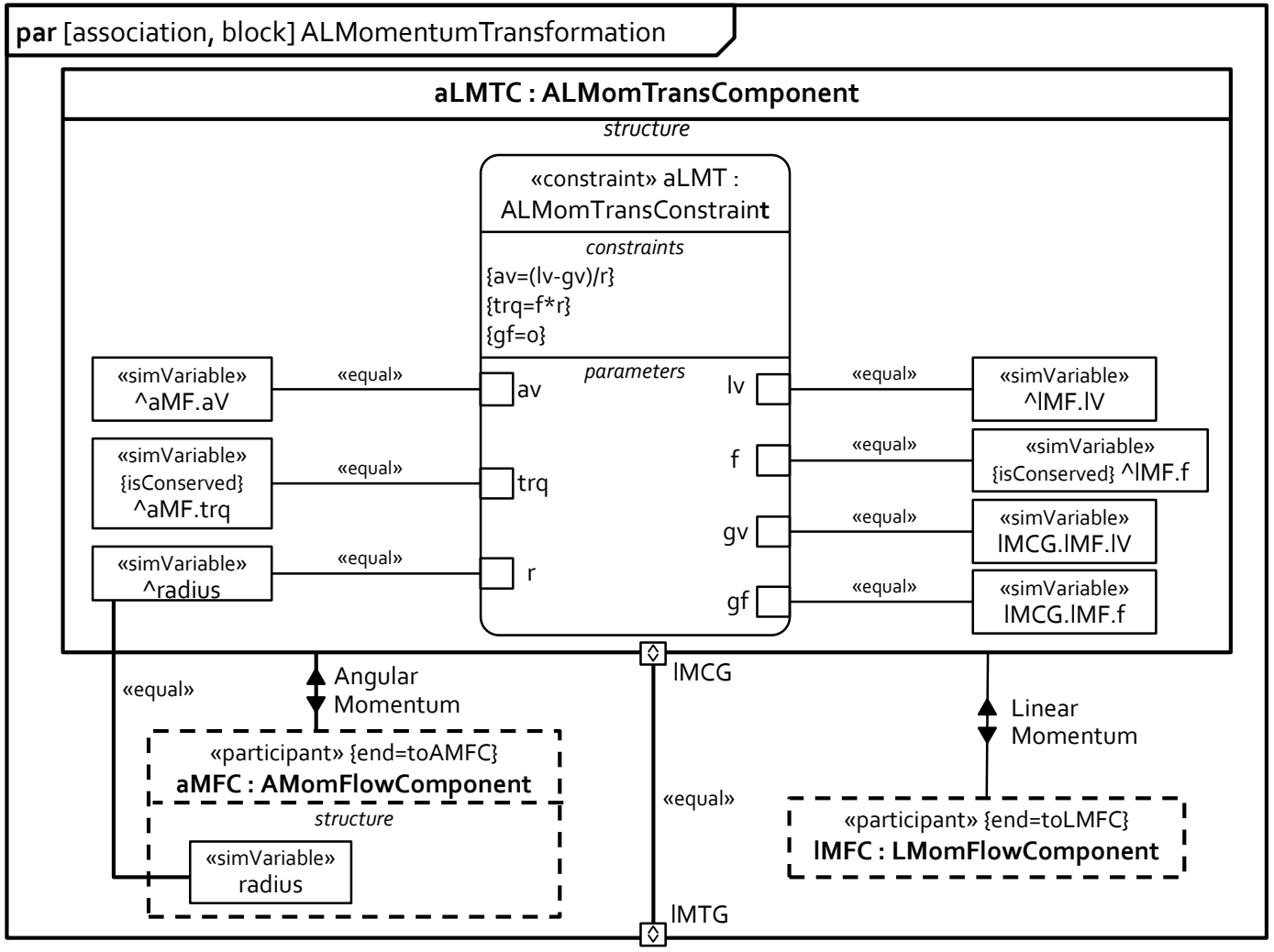
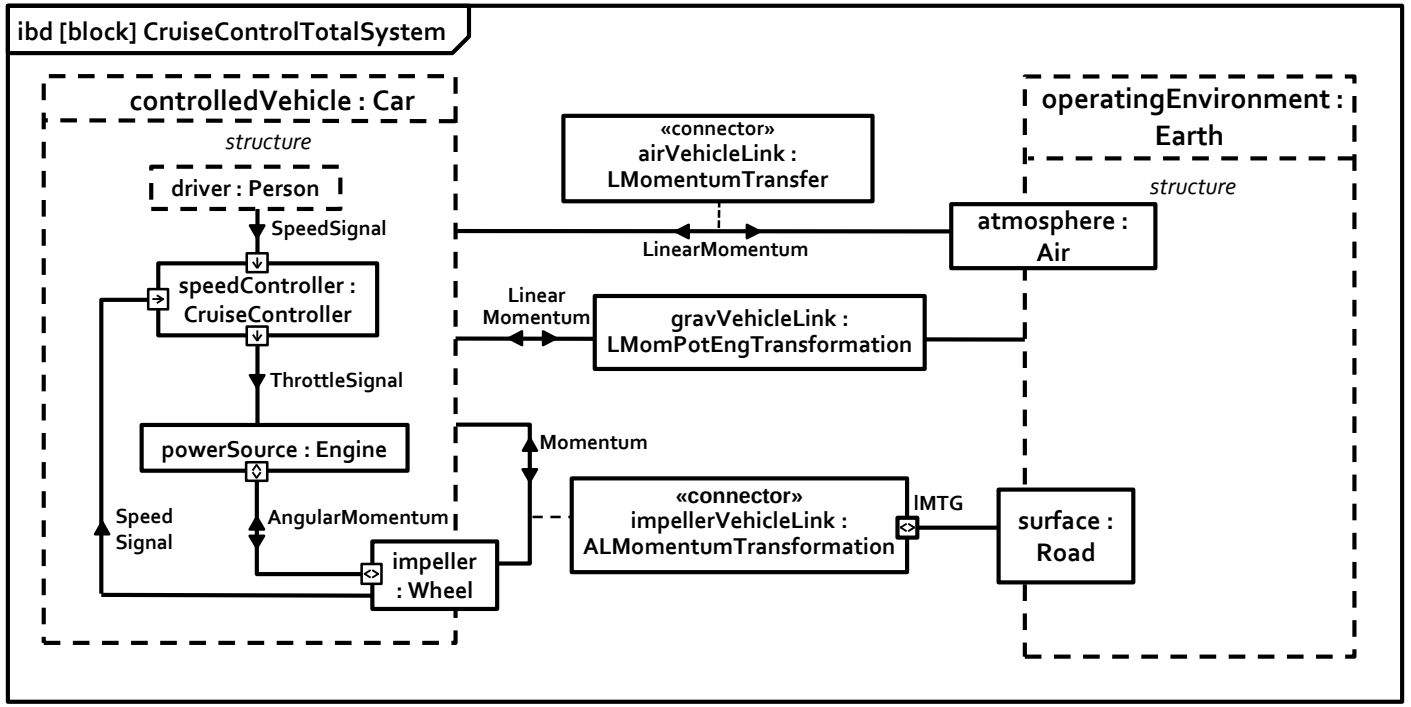


TABLE OF CONTENTS

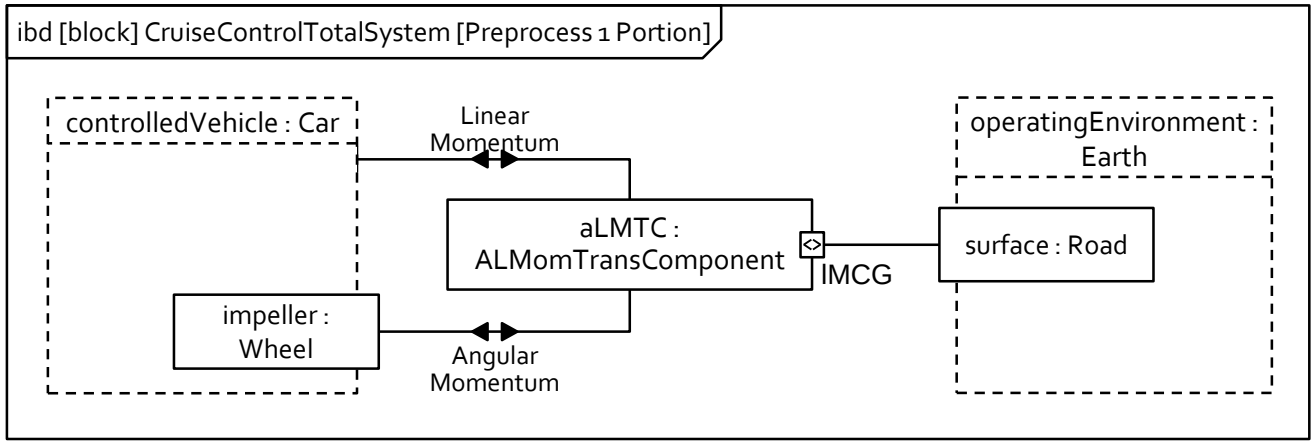
TRANSLATION: PRE-PROCESSING

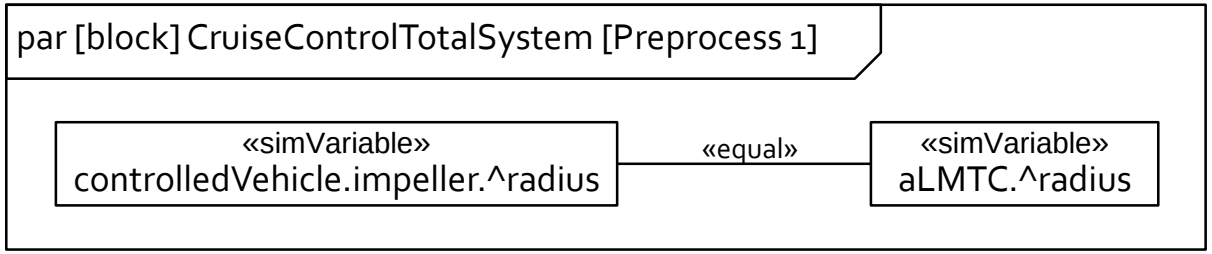
- SysML structures that simulation doesn't support
  - Association blocks
  - Flow / simulation properties on components / part
  - Nested ports
- Addressed by processing before translation.



PRE-PROCESSING: CHANGE ASSOCIATION BLOCKS INTO COMPONENTS

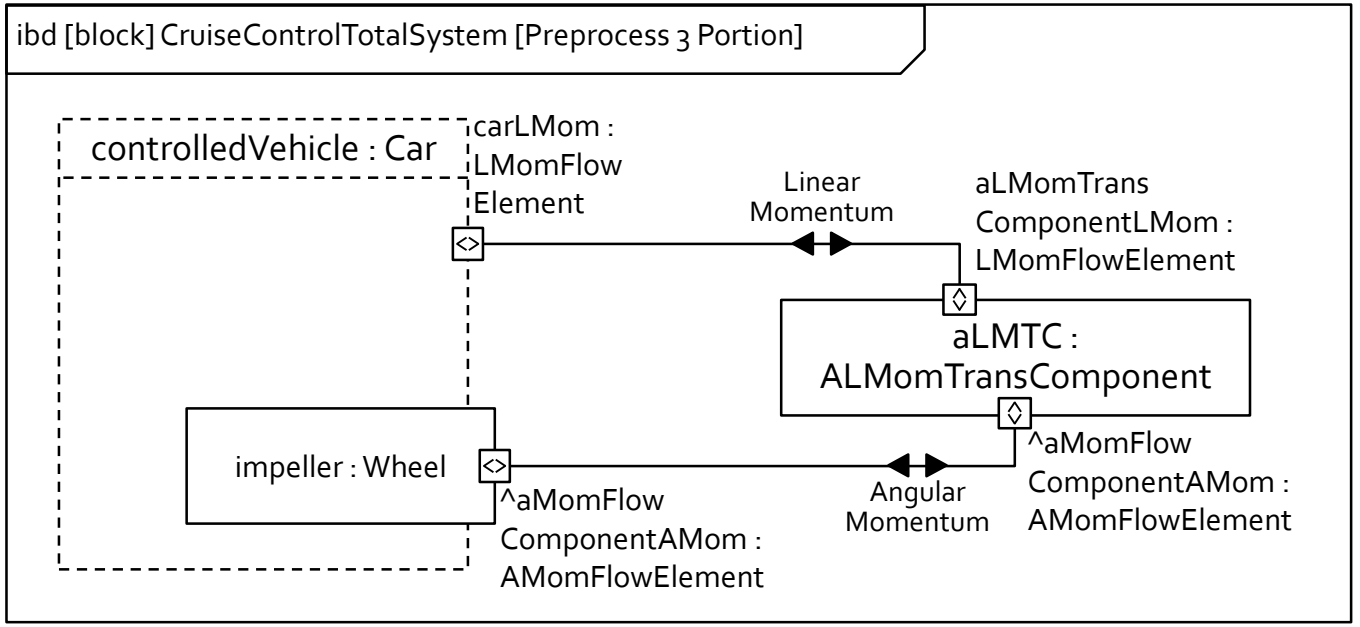
- Association blocks have their own interconnected parts
- Replace association blocks with their contents
- Move connectors in association block to block owning connector property.
  - Links to participant properties directly linked to participants
- Move binding connectors in association blocks to a parametric diagram





PRE-PROCESSING: MOVE SIMULATION / FLOW PROPERTIES FROM COMPONENTS TO PORTS

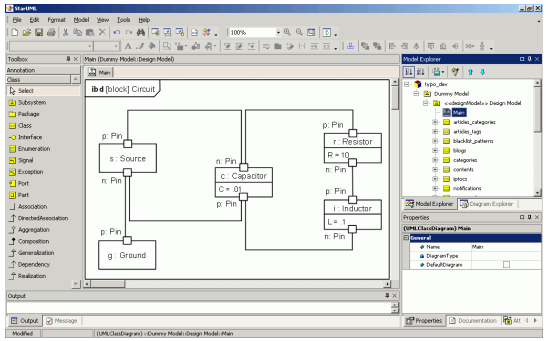
- SysML blocks can have flow / simulation properties
  - Car inherits linear momentum property and simulation property referring to it.
- Move flow / simulation properties to port.



TRANSLATION: BETWEEN EXTENDED SYSML AND SIMULATION PLATFORMS

Extended SysML	Modelica	Simulink / Simscape
Blocks without internal block diagrams	Models without connections	Block types / Components
Blocks with internal block diagrams	Models with connections	Systems / Components
Connectors	Connections	Lines / Connections
SimConstants	Parameters	Parameters
SimVariables (conserved)	Flow variables	NA / Balancing variables
SimVariables (not conserved)	Variables	NA / Variables
SimBlocks	Connectors	NA / Domains
Ports (physical interaction)	Ports (informal)	Ports / Nodes
Ports (signal flow)	Input, output variables	Input, output ports
ConstraintBlock usages	Equations	S-functions / Equations

AUTOMATED TRANSLATION



SysML Tool

```
<xmi:XMI xmi:uml=http://www.omg.org/spec/UML/20110701>
<packagedElement xmi:type="uml:Class" name="Circuit">
  <ownedAttribute xmi:type="uml:Property" name="r"
   type="_17_0_2_1_8340219_1386362605823_957929_2536"/>
  <ownedAttribute xmi:type-"uml:Property" name="c"
   type="_17_0_2_1_8340219_1386362584651_224589_2534"/>
  <owned.Attr-bute xml:type="uml:Property" name="i"
   type="_17_0_2_1_8340219_1386362591582_80360_2535"/>
  <ownedConnector xmi:type="uml:Connector"/>
  <ownedConnector xmi:type="uml:Connector"/>
  <ownedConnector xmi:type="uml:Connector"/>
  <ownedConnector xmi:type="uml:Connector"/>
</packagedElement>
<packagedElement xmi:type="uml:Class" name="Source"/>
<packagedElement xmi:type="uml:Class" name="Ground"/>
<packagedElement xmi:type="uml:Class" name="Capacitor"/>
<packagedElement xmi:type="uml:Association"/>
<packagedElement xmi:type="uml:Association"/>
</uml:Model>
</xmi>
```

SysML Model File

UML Repository

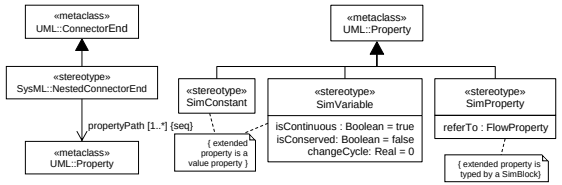
```
model circuit
  Resistor  R1(R=10);
  Capacitor C(C=0.01);
  Resistor  R2(R=100);
  Inductor  L(L=0.1);
  VsourceAC AC;
  Ground    G;
  equation
    connect (AC.p, R1.p);
    connect (R1.n, C.p);
    connect (C.n, AC.n);
    connect (R1.p, R2.p);
    connect (R2.n, L.p);
    connect (L.n,  C.n);
    connect (AC.n, G.p);
end circuit;
```

Simulator Input File

Modelica and Mathworks

loaded into

developed in

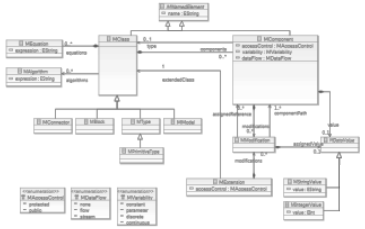


SysML and Simulator Extensions

```
public void generate(Class urootblock, Resource r) {
  sysmlutil = new SysMLUtil(r.getResourceSet());
  slibrary = SimulinkFactory.eINSTANCE.createSLib();
  slibrary.setName(urootblock.getName() + "Library");
  while(toProcess.size()!=0)
  { ReferenceKey rk = toProcess.pop();
    SElement selement = refs.get(rk);
    if (selement instanceof SSystem)
    { SSystem ssystem2 = processSys (rk.getKey()[0]);
      slibrary.getSystem().add(ssystem2.getSubsys());}
    else if (selement instanceof SFunction)
    { processSFunc(rk.getKey()[0], rk.getKey()[1];}
  }}
```

Translator Program

instantiates



Simulation Language Metamodels

MODEL TRANSLATION: MODELICA

- Load SysML model files as instances in UML metamodeling repository
- Apply pre-processing to instances
- Translate & emit simulation files (different platforms)
  - Components
  - Value of simulation constants
  - Conserved variables
  - Port properties & port types
  - Inputs/Outputs

```
model Car
 Person driver;
 CruiseController speedController(kI.start=30.0,kI.fixed=true,
                    kP.start=200.0,kP.fixed=true,
                    throttleAccRatio.start=1.0,
                    throttleAccRatio.fixed=true);
 Engine powerSource;
 Wheel impeller;
 LMomFlowComponent simCarLMom;
 AMomFlowSim simCarAMom;
 equation
  connect(speedController.speedDriverJack,
driver.simPersonSpeed);
  connect(speedController.throttleActuatorJack,
       powerSource.simEngineThrottleSetting);
  connect(speedController.speedSensorJack,
impeller.simWheelSpeed);
  connect(impeller.WheelAMom, powerSource.engineAMom);
  connect(impeller.simWheelAMom, simCarAMom);
end Car;
```

MODEL TRANSLATION: SIMSCAPE / SIMULINK

- Load SysML model files as instances in UML metamodeling repository
- Apply pre-processing to instances
- Translate & emit simulation files (different platforms)
  - Components
  - Value of simulation constants
  - Conserved variables
  - Port properties & port types
  - Inputs/Outputs

```
<Block BlockType="SubSystem" Name="Person" SID="2">
 <P Name="Ports">[0,1]</P>
 <System>
  <Block BlockType="Outport" Name="simPersonSpeed"
SID="3">
   <P Name="Port">1</P> </Block>
  <Block BlockType="M-S-Function" Name="pc" SID="4">
   <P Name="FunctionName">PersonConstraint</P>
   <P Name="Ports">[0,1]</P> </Block>
  <Line>
   <P Name="Src">4#out:1</P>
   <P Name="Dst">3#in:1</P> </Line>
 </System>
</Block>
```

```
model Person
 output Velocity simPersonSpeed;
equation
 simPersonSpeed=f(t);
end Person;
```

MODEL TRANSLATION: DEMONSTRATION

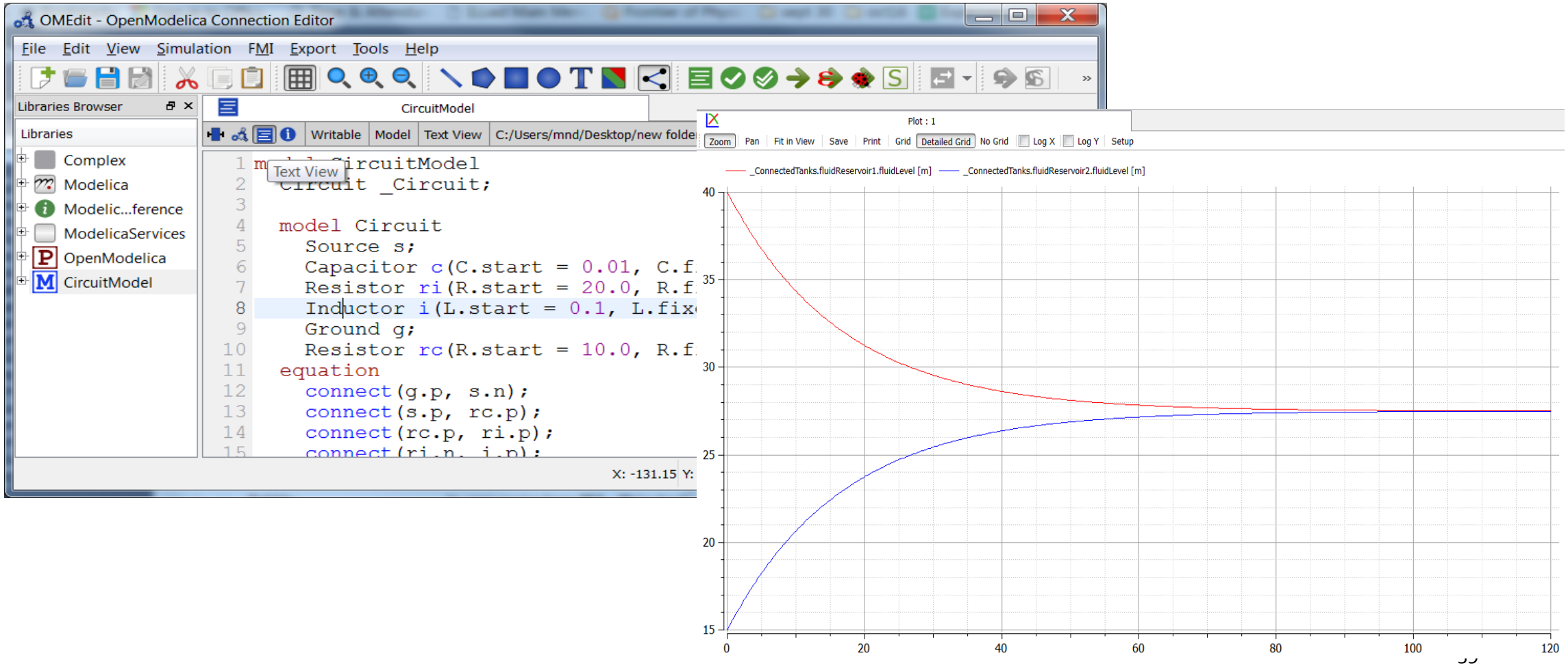


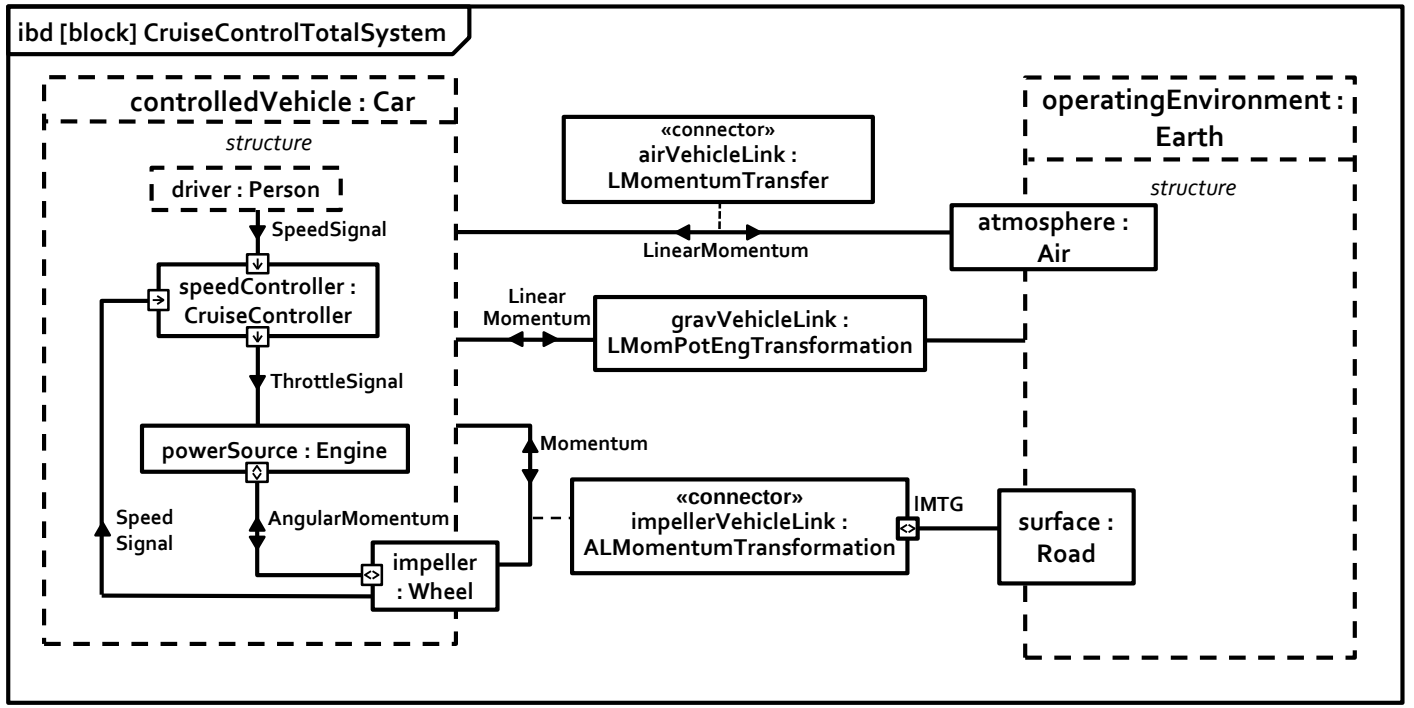
TABLE OF CONTENTS

DISCUSSION OF RESULTS

- Importance of physical principles
  - Flow of conserved physical substances within components and between components
- Models emphasize transformation and transmission of conserved substances, supported by equations and variable values → better model representation of system
- Include objects in operating environment of system
- More systematic modeling method for capturing system physics

METHOD BENEFITS: MODELING

- Physical processes modeled within system structure
- Equation development guided by flow and transformation of conserved substances
- Compare to piecing together equations defined elsewhere that are not tailored to the system



METHOD BENEFITS: COMPATIBILITY

- Better alignment of systems engineering and simulation models
  - Physical processes only occur within components and between neighboring components
  - Equations developed within graphical representation of system structure
- Better communication between engineers
  - Encourages systems and simulation analysts to discuss physical processes around similar, shared models

CONCLUSION

- Modeling method to facilitate integration of systems engineering models with physical interaction and signal flow simulations
  - Focusing on underlying physics (flow and transformation of conserved substances).
- More guidance in developing equations than other methods
- Encourages collaborative development of systems engineering and simulation models

QUESTIONS?