SysML – a modeling language for Systems Engineering

IDA - Dansk Selskab for Datateknik

Ingeniørhuset i København, 15. Marts 2010

Finn Overgaard Hansen

foh@iha.dk

Ingeniørhøjskolen i Århus

Agenda

- ➢ Systems Engineering and SysML
  - ➢ SE processes
- ➢ What is SysML?
- ➢ New SysML concepts and diagrams
  1. SysML Requirements
  2. SysML Structure
  3. SysML Behaviour
  4. SysML Parametric
- ➢ Perspectives for SysML

1

Systems Engineering

According to INCOSE:
*"Systems Engineering is an **engineering discipline** whose responsibility is creating and executing an **interdisciplinary process** to **ensure** that the **customer and stakeholder's needs are satisfied** in a high quality, trustworthy, cost efficient and schedule compliant manner **throughout a system's entire life cycle**"*

INCOSE: The International Council on Systems Engineering
Founded in 1990, 6720 members in dec. 2008
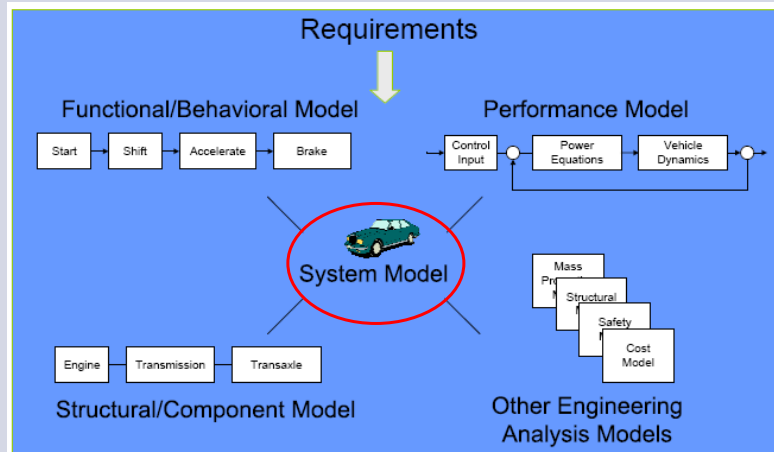
Model Based Systems Engineering (MBSE)

- From **document-based** to **model-based** approach
- A model-based approach requires modeling concepts and tools
- MBSE: producing and controlling a coherent **System Model**
- SysML is created to realize an MBSE approach based on a System model of the wanted system
- **SysML** is **a modeling language** not a **System Engineering (SE) process**

Systems Modeling

Requirements

Functional/Behavioral Model

| Start | Shift | Accelerate | Brake |

Performance Model

| Control Input | Power Equations | Vehicle Dynamics |

System Model

Structural/Component Model

| Engine | Transmission | Transaxle |

Other Engineering Analysis Models

Mass, Structural, Safety, Cost Model

The **SIMILAR** (SE) Process

Customer Needs

| **S**tate the Problem | **I**nvestigate Alternatives | **M**odel the System | **I**ntegrate | **L**aunch the System | **A**ccess Performance |

| **R**e-evaluate | **R**e-evaluate | **R**e-evaluate | **R**e-evaluate | **R**e-evaluate | **R**e-evaluate |

Note: All functions are performed in a parallel and iterative manner.

Outputs
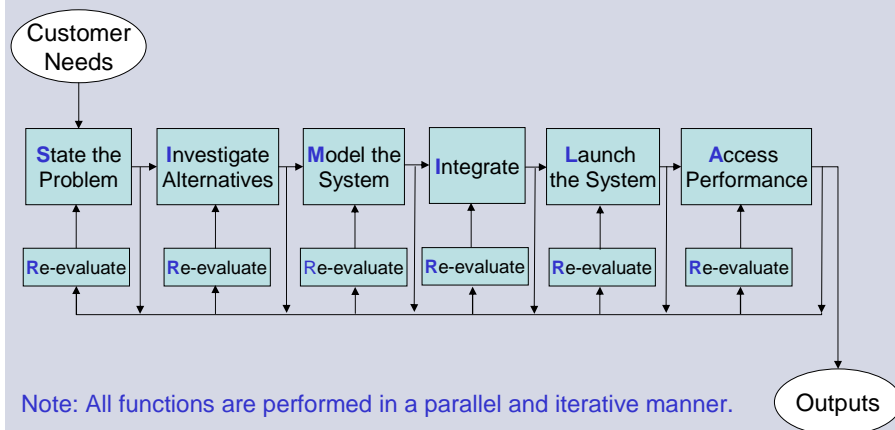
Ref. A.T. Bahill and B.Gissing, 1998

3

The Harmony (SE) Process (IBM)

iha.dk

- The Harmony process facilitates a seamless transition from **Systems Engineering** to **Software Engineering**
  - It uses SysML exclusively for system representation and specification.
- Harmony process characteristics:
  - a scenario-driven and iterative development process
  - promotes reuse of test scenarios throughout system development

Harmony Process for Systems Engineering

iha.dk
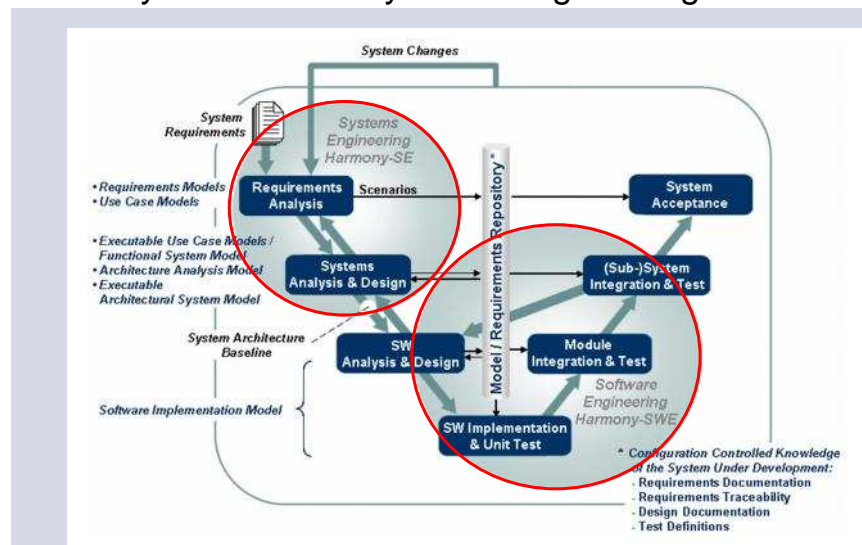
The Harmony process benefits

- The Harmony process models allow systems engineers to **find design errors** early in the development
- **Customer requests** can be more efficiently assessed, incorporated, and given timely feedback
- However, the **greatest benefit** of a model-driven process is **improved communication**
  - between engineering disciplines
  - between technical and non-technical parties
  - using different levels of abstraction
  - avoids information overload

What is SysML?

- A graphical modeling language created in response to the UML for Systems Engineering RFP developed by the **OMG** and **INCOSE**.
  - a UML Profile that represents a subset of UML 2 with important extensions
- Supports the specification, analysis, design, verification and validation of systems that include **hardware, software, data, personnel, procedures, and facilities**
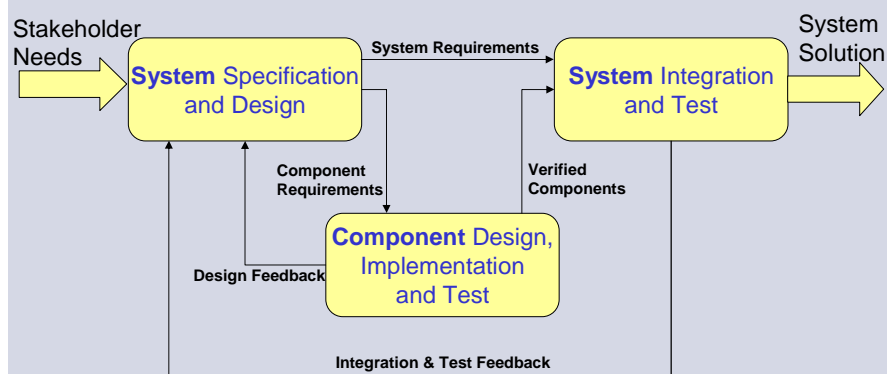- Supports model and data interchange via **XMI**

> **SysML is a Critical Enabler for Model Driven or Model Based Systems Engineering**
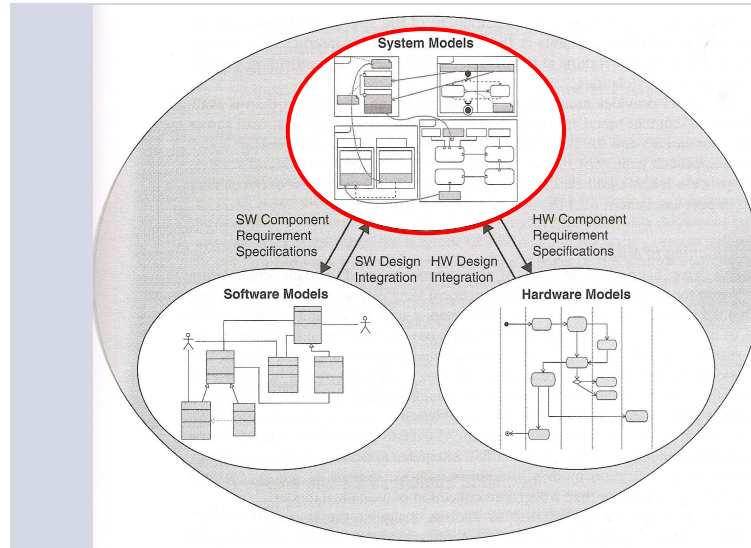
SysML Specification History and Status

iha.dk

- **Nov. 1997: UML V1.1 launched by OMG**
- **March 2003:** The UML for Systems Engineering RFP (Request for Proposal) was developed jointly by **OMG** and **INCOSE**
  - The SysML specification was developed in response to these requirements by the diverse group of **tool vendors, end users, academia, and government representatives**
- **Sept. 2007:** OMG SysML v.1.0
- **Nov. 2008:** OMG SysML v1.1
  - (doc.id: formal/2008-11-02, 256 pages)

System Engineering Technical Processes

iha.dk

Stakeholder Needs →

System Specification and Design

→ System Requirements →

System Integration and Test

→ System Solution

Component Requirements

Verified Components

Component Design, Implementation and Test

Design Feedback

Integration & Test Feedback

System Model and SW/HW Components

Comparison of SysML and UML
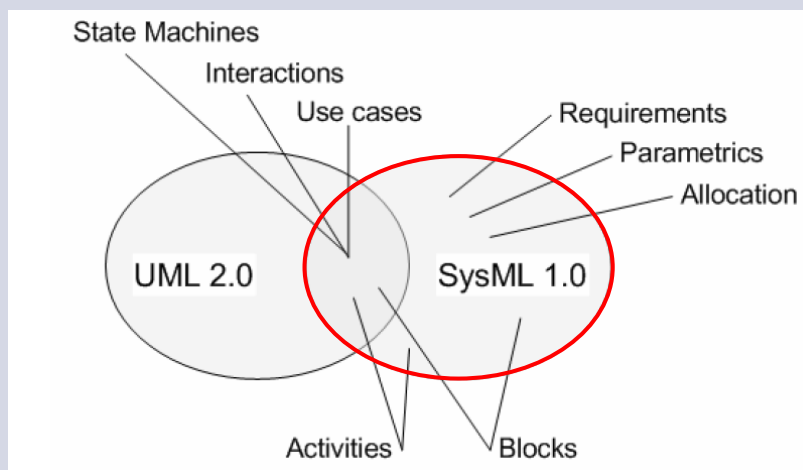
7

SysML Diagram Taxonomy
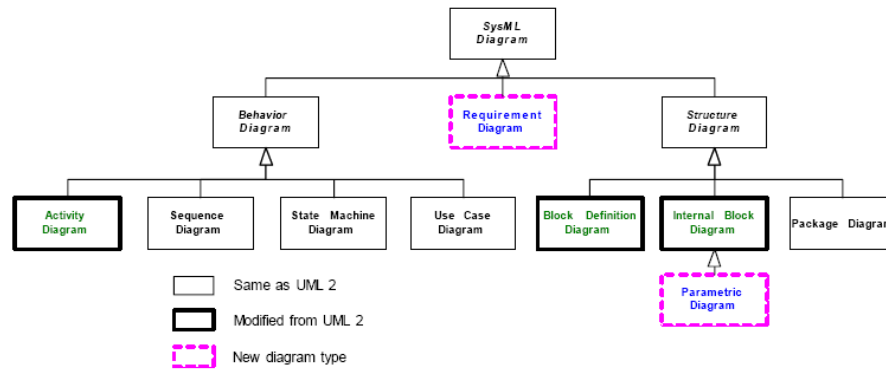
Major Extensions to UML 2.x

- New Diagram Types
  - Requirement Diagram (req)
  - Parametric Diagram (par)
- Structure Diagrams
  - Block Definition Diagram (bdd)
  - Internal Block Diagrams (ibd)
- Activity Diagrams
  - extensions for continuous flow modeling
  - extensions to support control operators

The 4 Pillars of SysML
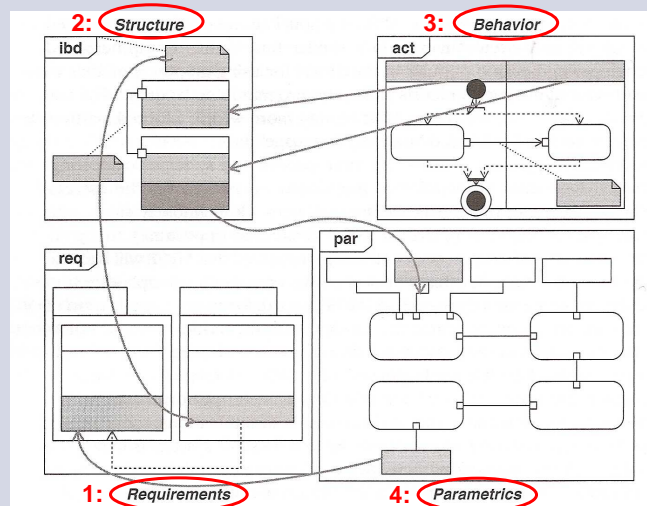
Project activities using SysML

1. Capture and analyze black box system requirements
   - System Context & System Use Cases, Requirement diagrams
2. Develop one ore more candidate system architectures
   - Block Definition & Internal Block diagrams
3. Perform engineering trade-off analysis to evaluate and select the optimal architecture
   - Parametric Diagrams
4. Specify component requirements and their traceability to system requirements
   - Requirement diagram
5. Verify the system design by executing system-level test cases

1. SysML Requirements

- Requirement Diagram – a NEW diagram type
- Graphical visualization of requirements
  - Functional
  - Non-functional
- Requirements can graphical be related to:
  - Other requirements
  - Design elements
  - Test Cases
- Standard stereotypes:
  - **derive, satisfy, verify, refine, trace and copy**
  - Used for requirement **traceability**

Requirement Diagram Example
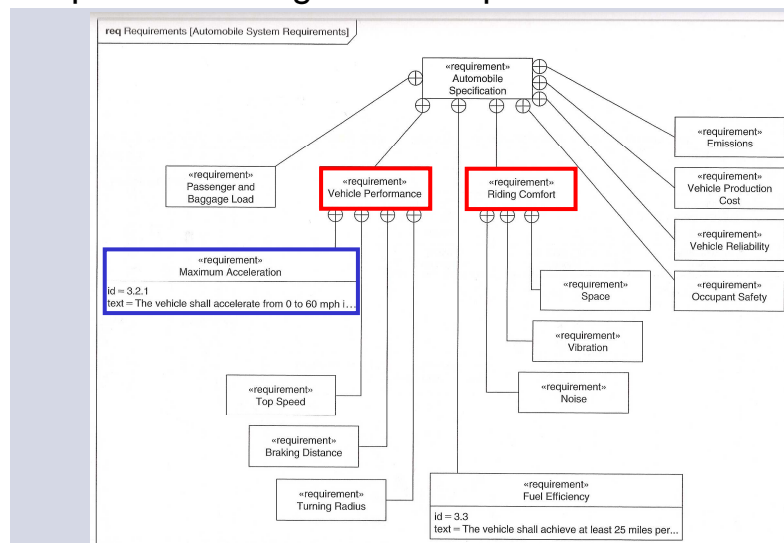
Requirement Traceability Example

---
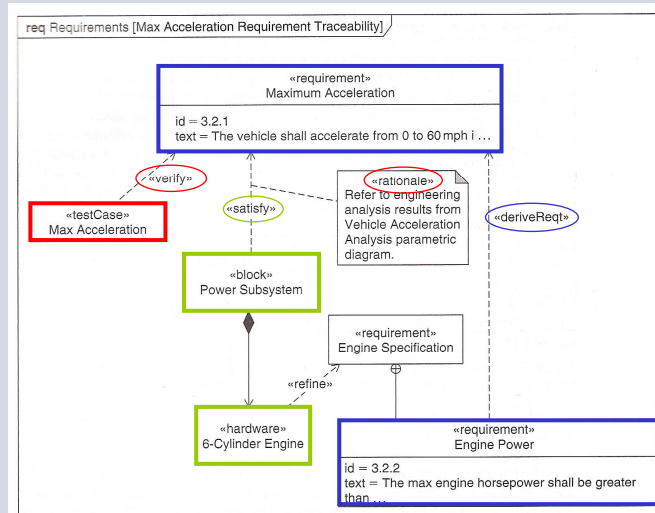
2. SysML Structure

- UMLs class concept is replaced with the **Block** concept
- A Block connect to other blocks via **Ports**
- Class diagrams are replaced with **Block Definition Diagrams (bdd)**
- Each Block has an **Internal Block Diagram (ibd)** where the internal parts are connected via **ports**
  - a replacement for class composite diagrams
- **Ports** can connect **discrete** as well as **continuous flows of material or information**

Blocks are Basic Structural Elements

- Provides a unifying concept to describe the structure of an element or system
  - Hardware
  - Software
  - Data
  - Procedure
  - Facility
  - Person

«block»
BrakeModulator

allocatedFrom
«activity»Modulate
BrakingForce

values
DutyCycle: Percentage
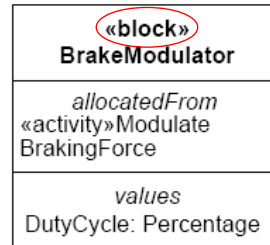
- Multiple compartments can describe the block characteristics
  - Properties (parts, references, values)
  - Operations
  - Constraints
  - Allocations to the block (e.g. activities)
  - Requirements the block satisfies

Blocks and Atomic Flow Ports

- A **flow port** describes an interaction point for items flowing in or out of a block

- An **atomic flow port** specifies only a single type of input or output

Imaging Assembly

optical image: Light → → electrical image: Image

Optical Assembly

external light: Light → → Optical image: Image
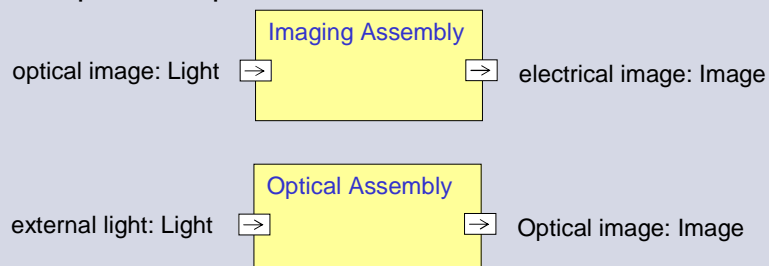
Blocks and Nonatomic Flow Ports

iha.dk

An interaction point with a complex interface
is modeled as a **Nonatomic Port**

:Light

«block»
Camera

camera i/o:
Camera Interface

station i/o:
Camera Interface

«block»
Monitoring
Station

A Conjugate flow port

«flow specification»
Camera Interface

flowProperties

out digital video: MPEG4
out analog video: Composite
in control: Control Data
in startup sig: Start Up

Connector and Ports

iha.dk

:Light

cameras:
Camera [4]

camera i/o:
Camera Interface

station i/o:
Camera Interface

:Monitoring
Station

4

1

Connector

:Camera Module

:Image

:Electronics
Assembly

:Image

Delegation Ports

ibd [Block] **Camera** [Nested flow]

:Camera Module

:Optical Assembly

:Light

:Imaging Assembly

:Light

:Image

:Electronic Assembly

:MPEG Converter

:MPEG4

:Video

:Image Processor

camera i/o:
Camera
Interface

:Light

Delegation port

Standard (service based) ports

**Camera
Control**

camera requests

Monitoring Station

Required interface

Provided
interface

«interface»
Camera Control

operations
getCameraStatus(in cameraId: Integer, in cameraStatus: String)
testCameras()
panCamera(in strength: Integer)
tiltCamera(in strength: Integer)

Block Definition Diagram Example

bdd Structure [Automobile Domain]

«block»
Automobile
Domain

0..4   bag   v   pe

«block»   «system of interest»   «block»
Baggage   Vehicle   Physical
Environment

Driver   Passenger

r   ee 0..*   atm

Vehicle Occupant

«block»   «block»   «block»
Road   External Entity   Atmosphere

values   values

friction : Real   temperature : Deg C
incline : Radians   air density : Mass/Volume

Internal Block Diagram for Automobile Domain
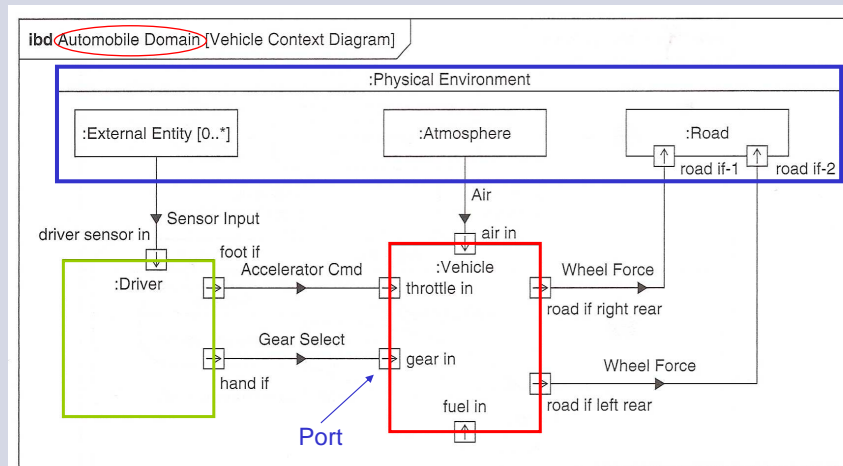
ibd Automobile Domain [Vehicle Context Diagram]

:Physical Environment

:External Entity [0..*]   :Atmosphere   :Road

road if-1   road if-2

driver sensor in   Sensor Input   Air

foot if   air in
:Driver   Accelerator Cmd   :Vehicle
throttle in   Wheel Force

road if right rear

Gear Select
hand if   gear in   Wheel Force

road if left rear
Port   fuel in

15

Block Definition Diagram Example

bdd Vehicle Structure [Vehicle Hierarchy]
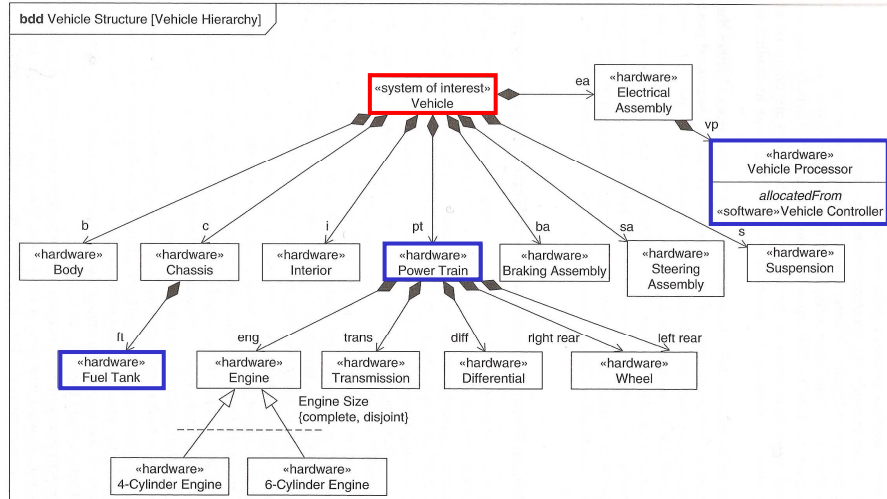
SysML - a modeling language for Systems Engineering   31

Internal Block Diagram Example

ibd Vehicle [Power Subsystem]
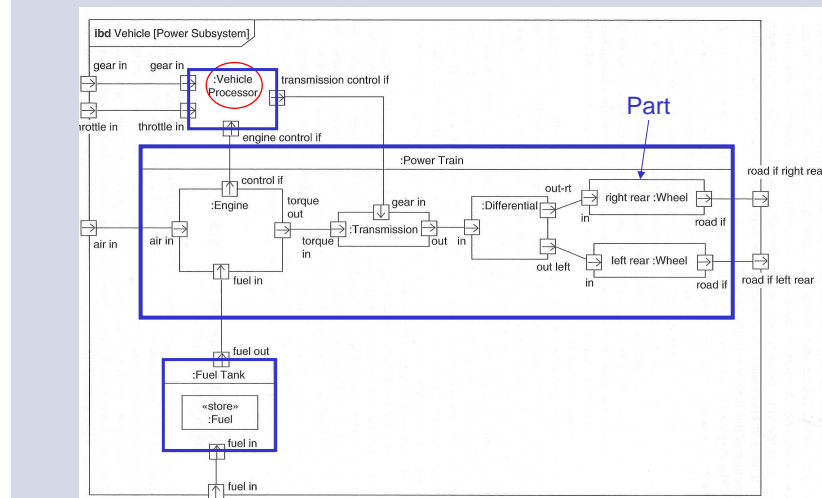
SysML - a modeling language for Systems Engineering   32

3. SysML - Behavior

- Activity diagrams are enhanced with new concepts
- Flows can be **continuous** and model **information** as well as **material flow**
- Control flows are introduced
- Activities can have pins

Input control flow

Requried input [1]   **Action a1**   Requried output [1]

Optional input [0..1]   Optional output [0..1]

Output control flow
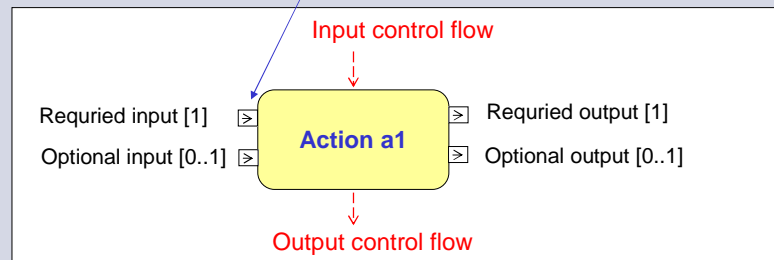
Activity Diagram with parameter nodes

act Operate Camera [Activity Frame]

«optional»
currentImage : Light[0..1]
{stream, direction = in }

«optional»
MPEG output : MPEG4[0..1]
{stream, direction = out }

config :ConfigurationData
{direction = in }

«optional»
composite out : Composite[0..1]
{stream, direction = out }
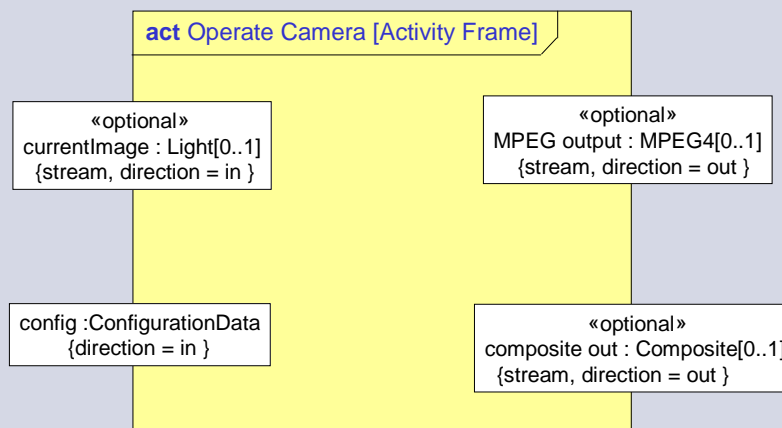
Activity Diagram - decomposed

act Operate Camera [Object Flow]

«optional»
current image
{stream}

:CollectImages

Object flow →

captured image
{stream}

video out
{stream}

:CaptureVideo

:GenerateVideo
Outputs

«optional»
MPEG output
{stream}

input signal
{stream}

«optional»
Composite out
{stream}

Subdiagram
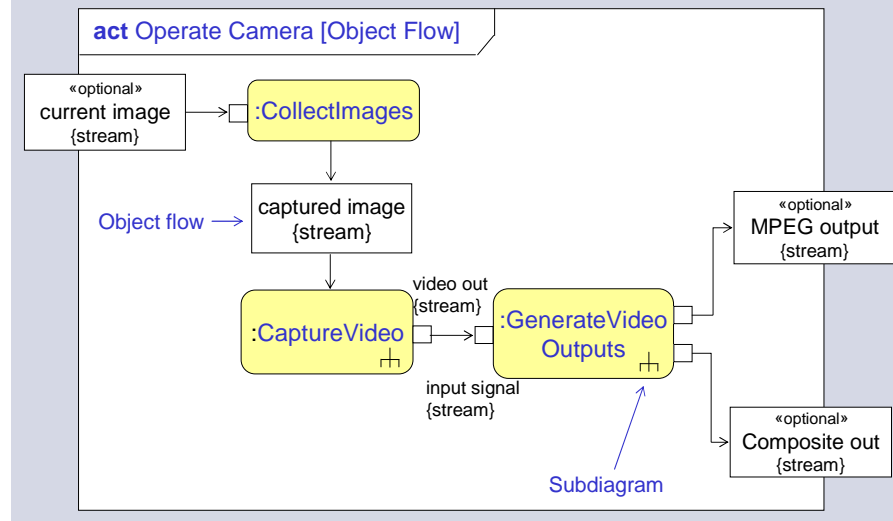
Activity Diagram Notation

act MonitorTraction

Activity

WheelRevs

Action

Initial
Node

Calculate
Wheel
Velocity

Decision

Fork

AngularVelocity
Calculate
Traction
Speed

[loss
of traction]

Calculate
Modulation
Frequency

Modulation
Frequency

Control
Flow

Calculate Car
Velocity

[else]

SpeedoInput

Object
Flow

Flow
Final
Node

Activity
Final
Node

Activity
Parameter
Node

Pin
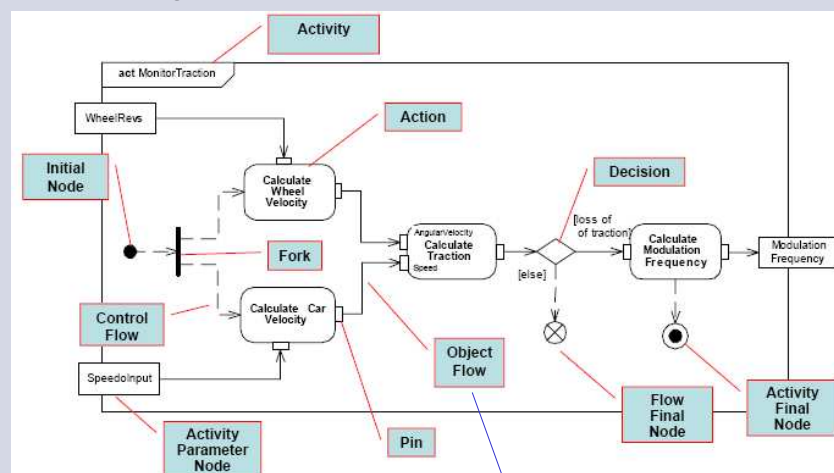
Flows can be discrete, streaming or control

18

Activity Diagram Example

---
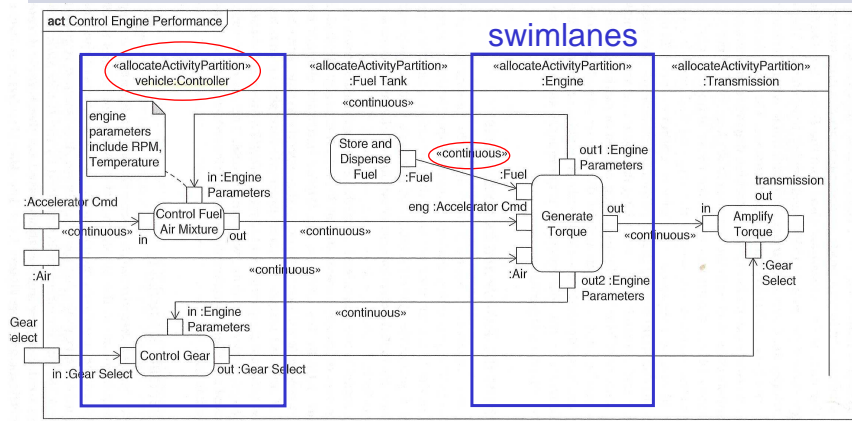
4. SysML Parametric

- Parametric Diagram – a NEW diagram type
- Used to express constraints (equations) between value properties
  - Provides support for engineering analysis (e.g., performance, reliability)
- Constraint block captures equations shown on a bdd
  - Expression language can be **formal** (e.g., MathML, OCL) or **informal**
  - Computational engine is defined by applicable analysis tool and not by SysML
- Parametric diagram represents the usage of the constraints in an analysis context
  - Binding of constraint usage to value properties of blocks (e.g., vehicle mass bound to F= m × a)
- Parametric enable integration of engineering analysis with design models

BDD Parametric Constraint Blocks

iha.dk

bdd Parametrics [Analysis Context]

«block»
Analysis
Context

a «block»
Automobile
Domain

«constraintBlock»
Gravitational Force
parameters
w: Pounds
fg: Pounds
theta: Radians
constraints
{fg = w/g * sin(theta)}

«constraintBlock»
Drag Force
parameters
fd: Pounds
rho: Mass/Volume
v: Ft/sec
Cd: Real
constraints
{fd = 0.5*rho*v*v*Cd}

«constraintBlock»
Power Train Force
parameters
fp: Pounds
constraints
{fp(Teng, Ttrans, Tdiff, fw)}

«constraintBlock»
Total Force
parameters
fi: Pounds
fk: Pounds
ft: Pounds
fj: Pounds
constraints
{ft = fi+fj+fk}

«constraintBlock»
Acceleration
parameters
f: Pounds
a: Ft/sec2
w: Pounds
constraints
{f = (w/g) *a}

«constraintBlock»
Integrator
parameters
y: Ft/sec
y0: Ft/sec
x: Ft/sec2
constraints
{y = integral{xdt}+y0}

«constraintBlock»
Engine Torque
Teng: Foot pounds
fuel flow rate: Mass/Volume
Ncyl: Integer
displ: Cubic Inches
eff eng: Percent
rpm: Real
constraints
{Teng (fuel flow rate, Ncyl, displ, eff eng, rpm)}

«constraintBlock»
Transmission Torque
parameters
torque in: Foot pounds
gear ratio: Integer
Ttrans: Foot pounds
eff tm: Percent
constraints
{Ttrans (torque in, gear ratio, eff tm)}

«constraintBlock»
Differential Torque
parameters
Tdiff: Foot pounds
eff ta: Percent
torque in: Foot pounds
k: Integer
constraints
{Tdiff (torque in, k, eff ta)}

«constraintBlock»
Wheel Force
parameters
fw: Pounds
torque in: Foot pounds
wheel dia: Inches
tire friction: Real
constraints
{fw (torque in, wheel dia, tire friction)}

SysML - a modeling language for Systems Engineering

39

Parametric Diagram - Example

iha.dk

par [block]Analysis Context [Vehicle Acceleration Analysis]

a.pe.r.incline

a.v.wt

theta   w
:Gravitational Force
fg

:Power Train Force
fp

fk
:Total Force
fj   fi   ft

w
:Acceleration
f   a

:Integrator
x(t)   y0   y

a.v.speed

fd
:Drag Force
rho   cd   v

a.v.b.drag
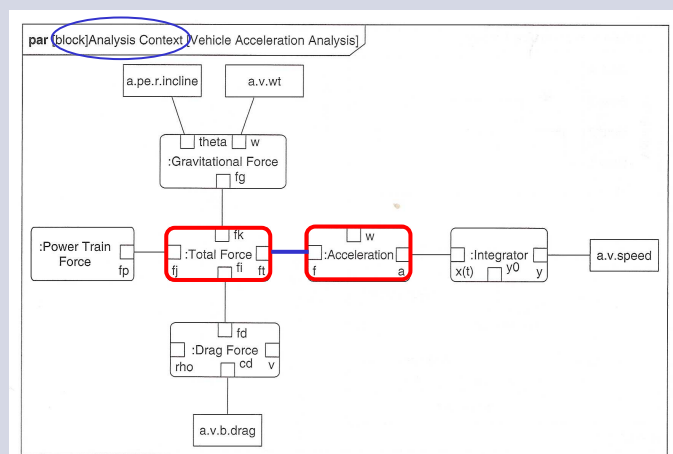
SysML - a modeling language for Systems Engineering

40

20

Combining Model-Driven (MDD) and Model Based Design (MBD) in Industrial Machine Control

iha.dk

MDD: Model Driven Development in Rhapsody (IBM)   **MBD: Model Based Design in Simulink (Mathworks)**

MDD versus MBD Feature Comparison

iha.dk

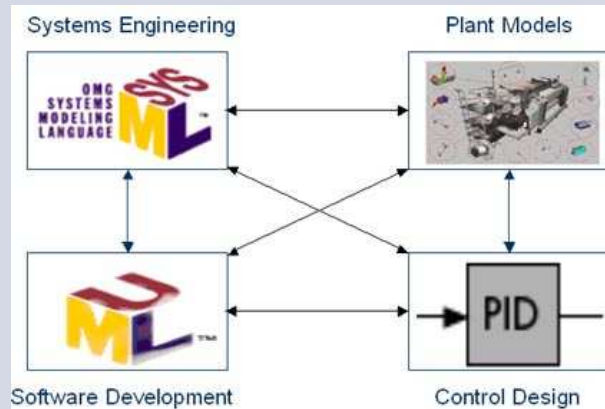| Tool Environment Capabilities	Rhapsody	Simulink
Systems and Software Development Environment	X	
Mathematical Block Diagram Environment		X
UML 2.1, SysML 2.0 Based Code Generation	X	
Dynamic Controls and Signal Processing Engineering		X
Logical Algorithm Development		X
Requirements Traceability & Documentation	X	
Define Physical, Functional & Software Architecture	X	
Mathematical Algorithm Development		X
SysML/UML Analysis, Simulation & Test	X	
Physical Plant Behavior Modeling		X

Table 1. MDD versus MBD feature comparison

Cross Connecting Model Elements

Structure

Behavior

Requirements

Parametrics

ibd [block] Anti-LockController [Internal Block Diagram]

satisfies «requirement» Anti-Lock Performance

d1:TractionDetector

c1:modulator Interface

allocatedFrom «activity»DetectLoss Of Traction

m1:BrakeModulator

allocatedFrom «ObjectNode» TractionLoss:

allocatedFrom «activity»Modulate BrakingForce

values DutyCycle: Percentage

satisfy

act PreventLockup [Swimlane Diagram]

«allocate» :TractionDetector

«allocate» :BrakeModulator

DetectLossOf Traction

TractionLoss:

Modulate BrakingForce

allocatedTo «connector»c1:modulatorInterface

allocate

value binding

req [package] VehicleSpecifications [Requirements Diagram - Braking Requirements]

Vehicle System Specification

Braking Subsystem Specification

«requirement» StoppingDistance

id="102"
text="The vehicle shall stop from 60 mph within 150 ft on a clean dry surface."

VerifiedBy «interaction»MinimumStopp ingDistance

«requirement» Anti-LockPerformance

id="337"
text="Braking subsystem shall prevent wheel lockup under all braking conditions."

SatisfiedBy «block»Anti-LockController

«deriveReqt»

verify

par [constraintBlock] StraightLineVehicleDynamics [Parametric Diagram]

v.chassis.tire. Friction:

v.brake.abs.m1. DutyCycle:

v.brake.rotor. BrakingForce:

v.Weight:

tf: tl: bf:

m:

:BrakingForce Equation
[f = (tf*bf)*(1-tl)]

f:

:Accelleration Equation
[F = ma]

F:

:DistanceEquation
[v = dx/dt]

v:

:VelocityEquation
[a = dv/dt]

a:

v:

x:

v.Position:

Vendors of SysML tools

- ARTiSAN Software Tools
- EmbeddedPlus Engineering (Third party for IBM Rational)
- IBM
  - Rhapsody
  - Tau
- InterCAX
- No Magic
- Papyrus for SysML (open source eclipse modeling tool)
- Software Stencils - Microsoft Visio SysML and UML templates
- Sparx Systems

Perspectives for SysML

iha.dk

- Enable a common modeling language and model across engineering disciplines
- Enable traceability between disciplines
- Enable different kinds of system analysis
- Enable integration of discrete and continuous based modeling tools
- Critical enabler for Model Based System Engineering with tool support

Summary

iha.dk

- SysML a common modeling language for different disciplines e.g. Hardware, Software and Mechanics
- New and important concepts for cross disciplinary analysis of system properties (e.g. parametric)
- Blocks and ports as general modeling elements
- Important enhancement to activity diagrams
- Lot of support for traceability between models and model elements
- Must be supported by an appropriate SE process

References

- OMGs SysML homepage: www.omgsysml.org
- INCOSE organization: www.incose.org
  - *"Re-evaluating systems engineering concepts using systems thinking"*, A.T. Bahill and B. Gissing, IEEE Transaction on Systems, Man and Cybernetics, Part C, 28 (4), 516-527, 1988.
- IBM Rational Harmony:
  - http://www-01.ibm.com/software/rational/services/harmony/
- *Books:*
  - *"A Practical Guide to SysML – The System Modeling Language"*, Sanford Friedenthal, Allan Moore, Rick Steiner, Elsevier, 2008.
  - *"Systems Engineering with SysML/UML – Modeling, Analysis, Design"*, Tim Weilkiens, Elsevier, 2007.