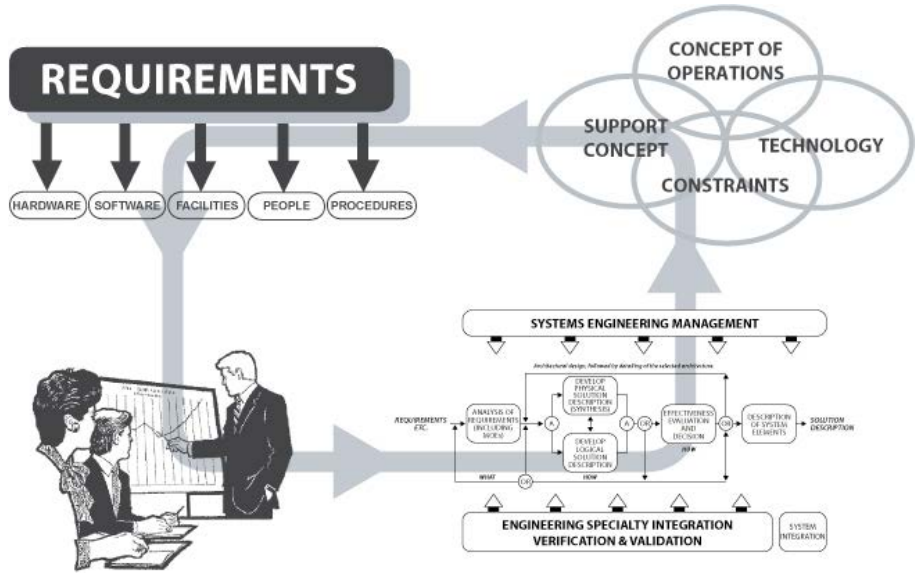


Short Introduction about Systems Engineering and SysML

Benoit Combemale

CNRS IRIT (SM@RT team)

University of Toulouse - Jean Jaurès

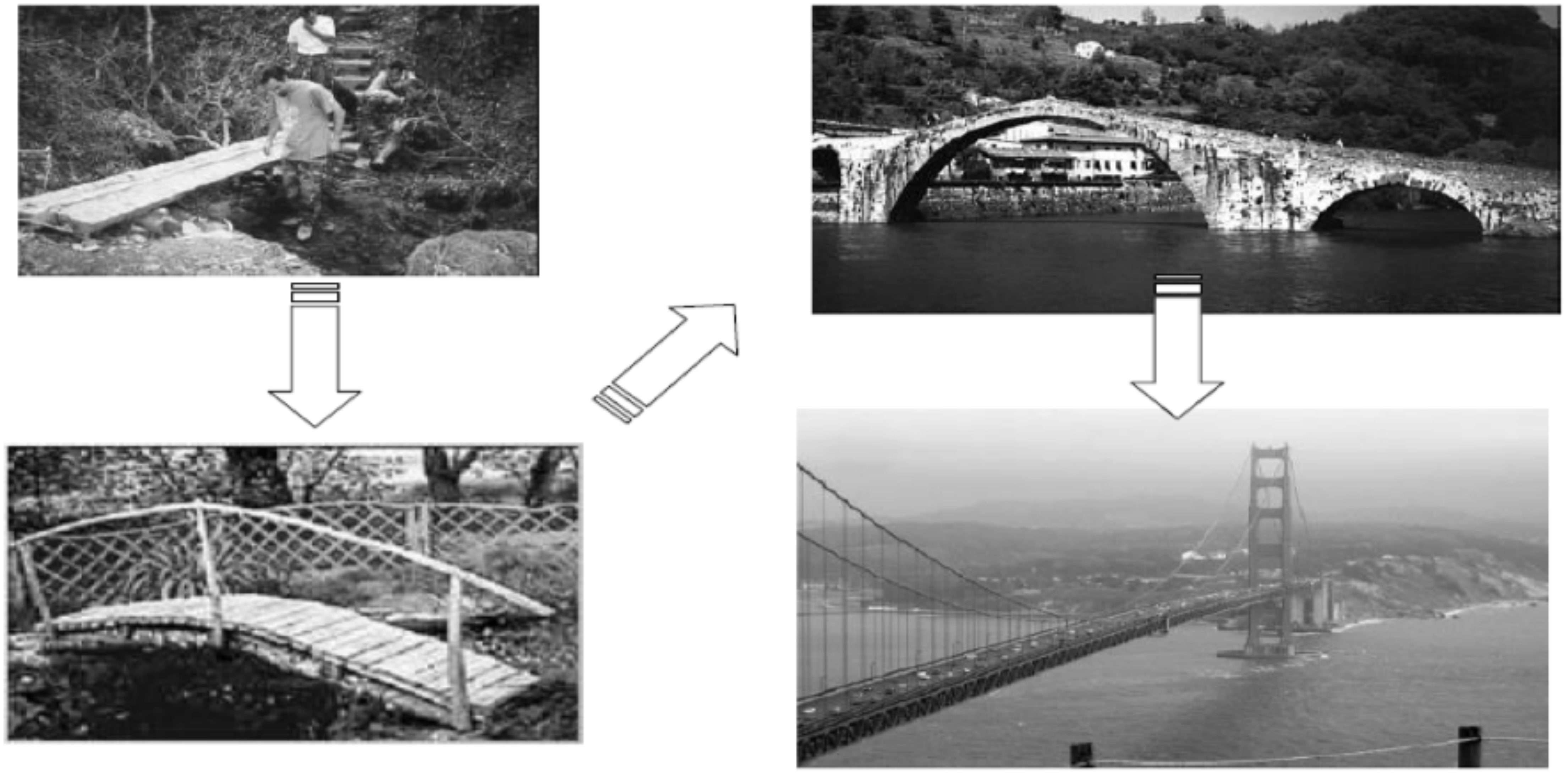
benoit.combemale@irit.fr

Inspired from the OMG specification SysML v1.2, from the OMG/INCOSE tutorial, from the Prof. J.-M. Bruel lecture, and the G. Finance's article (Object Direct).

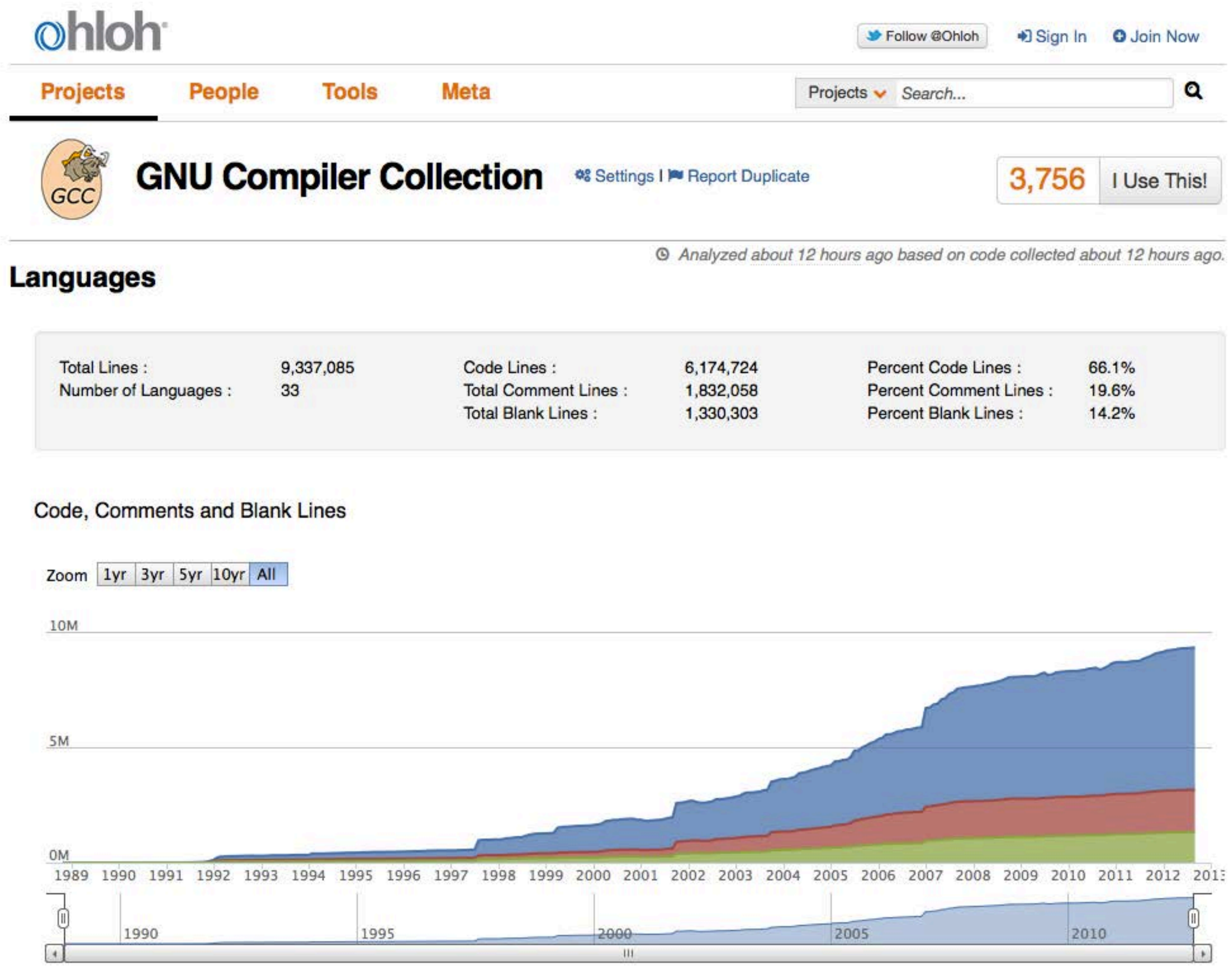
Version Oct., 2017.

Materials available on : https://combemale.github.io/

System Complexity



System Complexity



GNU Compiler Collection

Analyzed about 12 hours ago based on code collected about 12 hours ago.

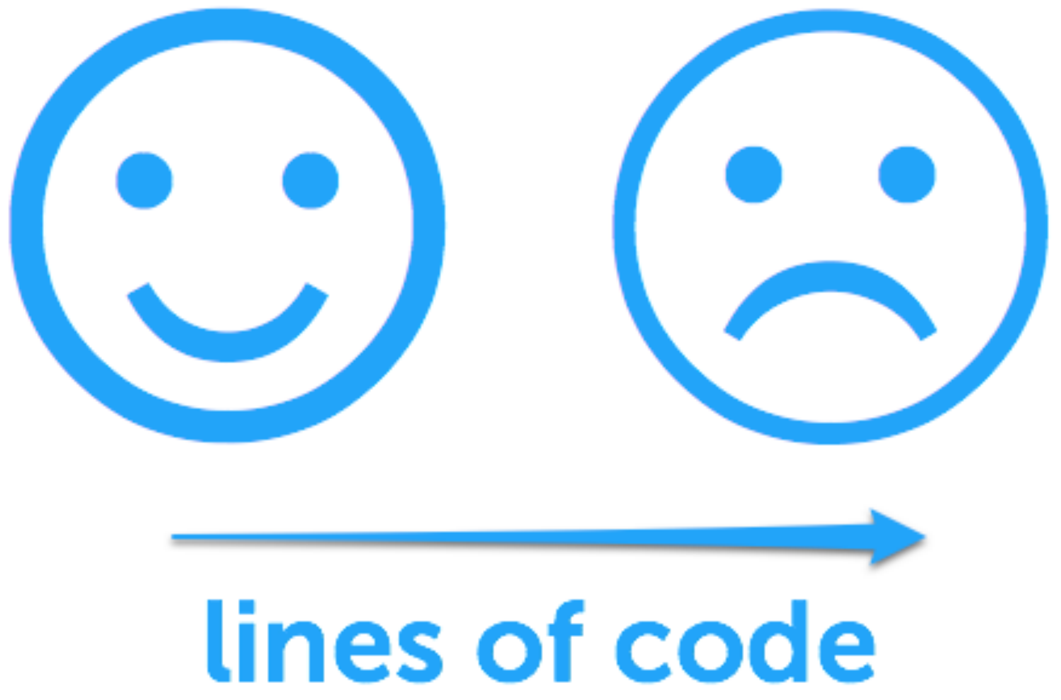
Languages

Total Lines :	9,337,085	Code Lines :	6,174,724	Percent Code Lines :	66.1%
Number of Languages :	33	Total Comment Lines :	1,832,058	Percent Comment Lines :	19.6%
		Total Blank Lines :	1,330,303	Percent Blank Lines :	14.2%

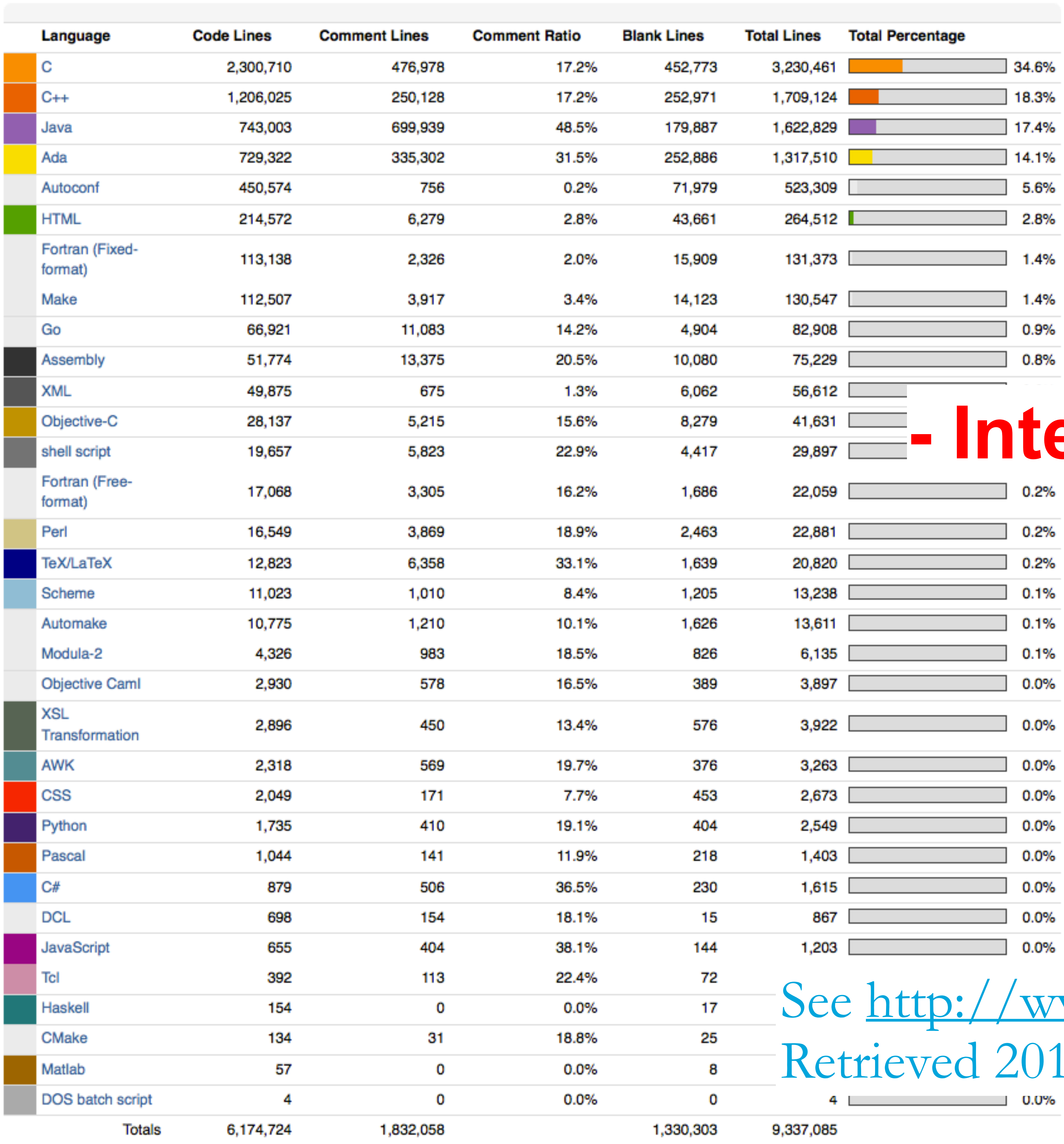
Code, Comments and Blank Lines

See http://www.ohloh.net/p/gcc. Retrieved 2012-09-16.

System Complexity



But also...



Language	Code Lines	Comment Lines	Comment Ratio	Blank Lines	Total Lines	Total Percentage
C	2,300,710	476,978	17.2%	452,773	3,230,461	34.6%
C++	1,206,025	250,128	17.2%	252,971	1,709,124	18.3%
Java	743,003	699,939	48.5%	179,887	1,622,829	17.4%
Ada	729,322	335,302	31.5%	252,886	1,317,510	14.1%
Autoconf	450,574	756	0.2%	71,979	523,309	5.6%
HTML	214,572	6,279	2.8%	43,661	264,512	2.8%
Fortran (Fixed-format)	113,138	2,326	2.0%	15,909	131,373	1.4%
Make	112,507	3,917	3.4%	14,123	130,547	1.4%
Go	66,921	11,083	14.2%	4,904	82,908	0.9%
Assembly	51,774	13,375	20.5%	10,080	75,229	0.8%
XML	49,875	675	1.3%	6,062	56,612	
Objective-C	28,137	5,215	15.6%	8,279	41,631	
shell script	19,657	5,823	22.9%	4,417	29,897	
Fortran (Free-format)	17,068	3,305	16.2%	1,686	22,059	0.2%
Perl	16,549	3,869	18.9%	2,463	22,881	0.2%
TeX/LaTeX	12,823	6,358	33.1%	1,639	20,820	0.2%
Scheme	11,023	1,010	8.4%	1,205	13,238	0.1%
Automake	10,775	1,210	10.1%	1,626	13,611	0.1%
Modula-2	4,326	983	18.5%	826	6,135	0.1%
Objective Caml	2,930	578	16.5%	389	3,897	0.0%
XSL Transformation	2,896	450	13.4%	576	3,922	0.0%
AWK	2,318	569	19.7%	376	3,263	0.0%
CSS	2,049	171	7.7%	453	2,673	0.0%
Python	1,735	410	19.1%	404	2,549	0.0%
Pascal	1,044	141	11.9%	218	1,403	0.0%
C#	879	506	36.5%	230	1,615	0.0%
DCL	698	154	18.1%	15	867	0.0%
JavaScript	655	404	38.1%	144	1,203	0.0%
Tcl	392	113	22.4%	72		
Haskell	154	0	0.0%	17		
CMake	134	31	18.8%	25		
Matlab	57	0	0.0%	8		
DOS batch script	4	0	0.0%	0	4	0.0%
Totals	6,174,724	1,832,058		1,330,303	9,337,085	

- Interoperability

See http://www.ohloh.net/p/gcc. Retrieved 2012-09-16.

System Complexity

- Reusability
- Durability

=> Time To Market!

- Variability

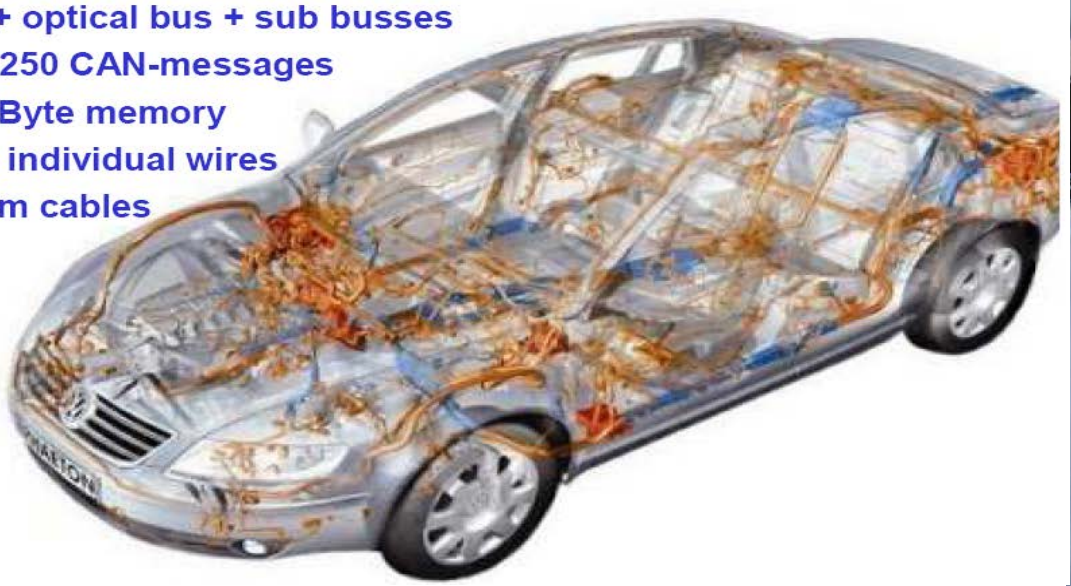
SM@RT

System Complexity

- Embedded
- Critical
- Real time

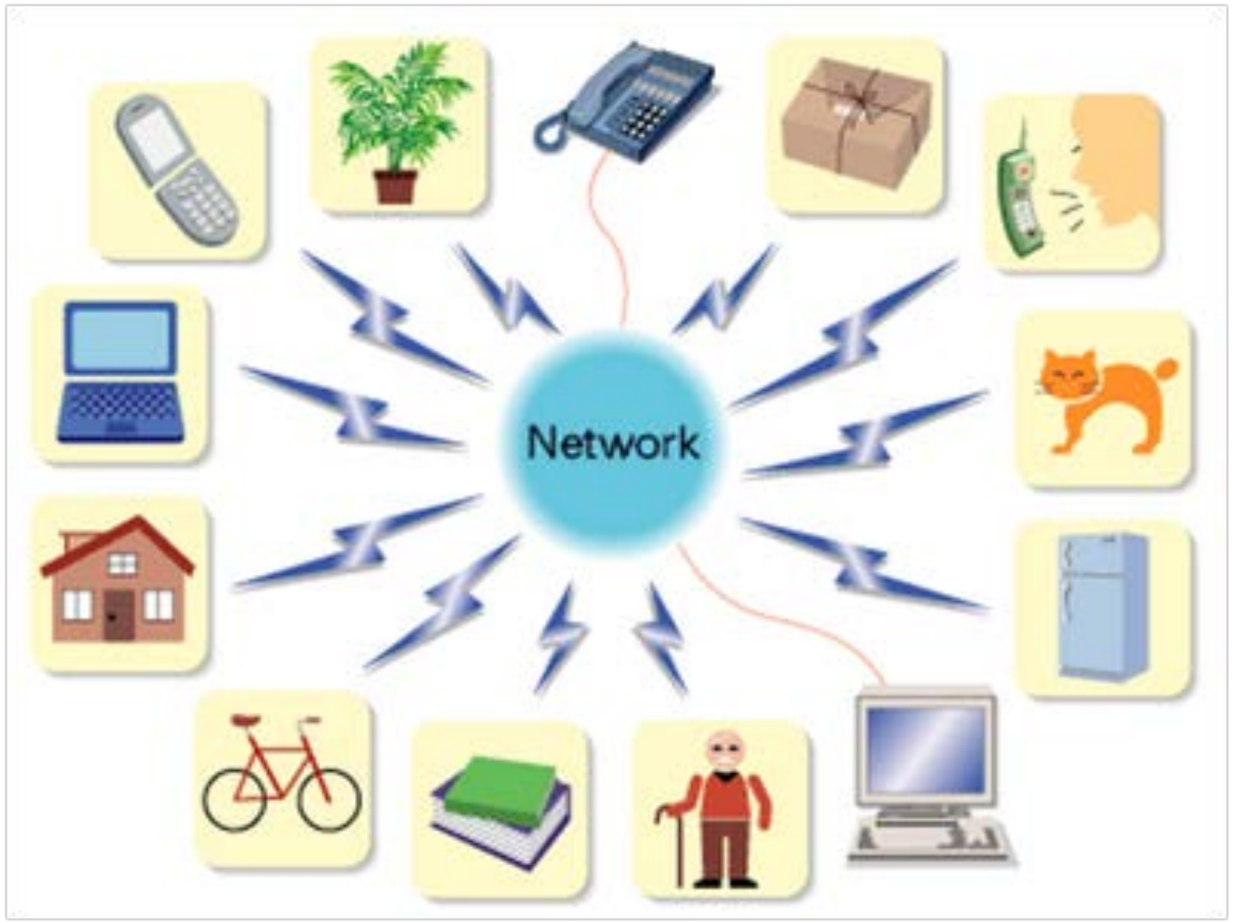
Phaeton

- 61 networked ECUs
- 3 bus systems + optical bus + sub busses
- 2500 signals in 250 CAN-messages
- more than 50 MByte memory
- more than 2000 individual wires
- more than 3800m cables

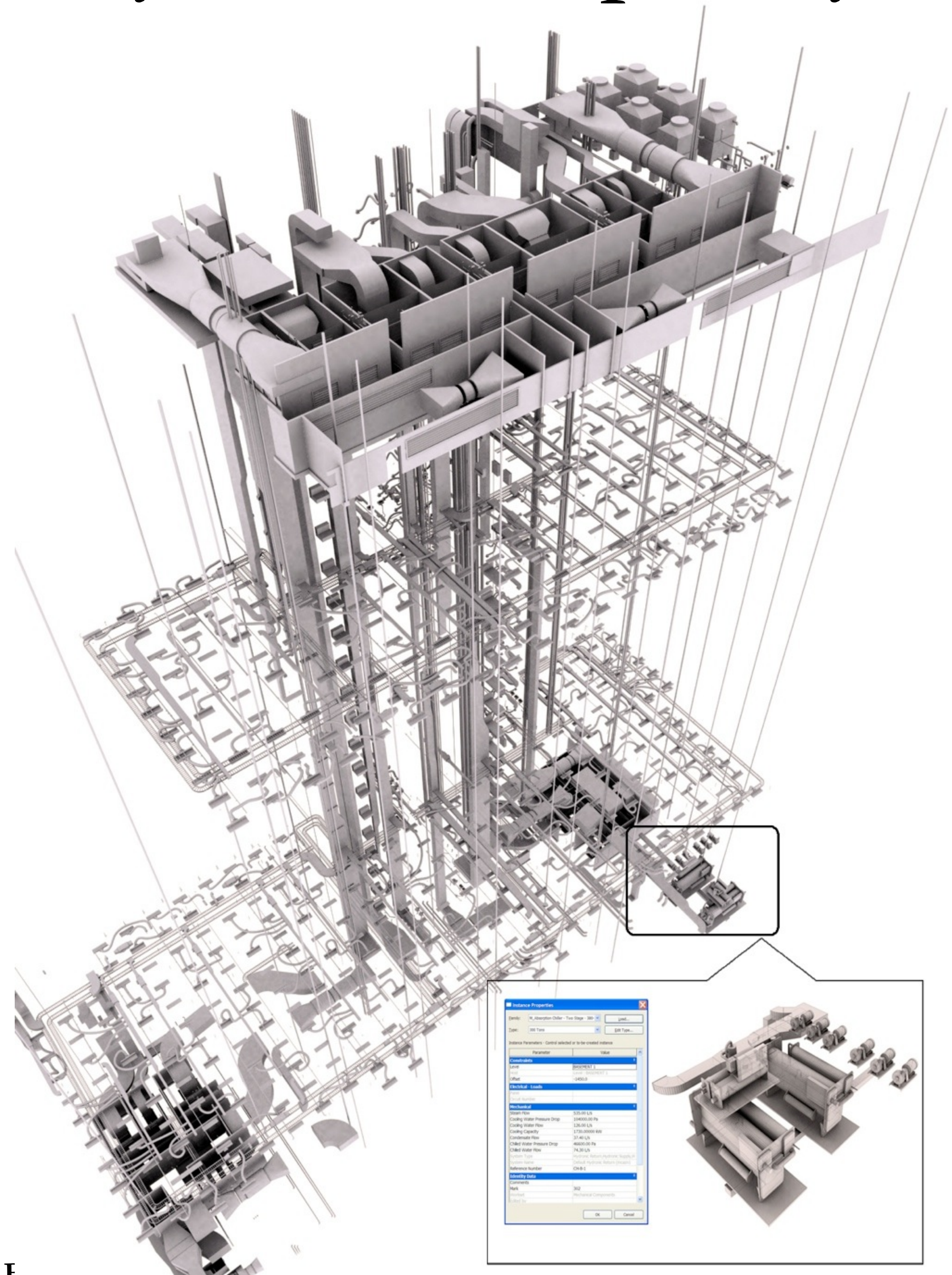


System Complexity

- Autonomic Computing
- Cloud Computing
- SaaS, IoS, IoT
- System of Systems



System Complexity



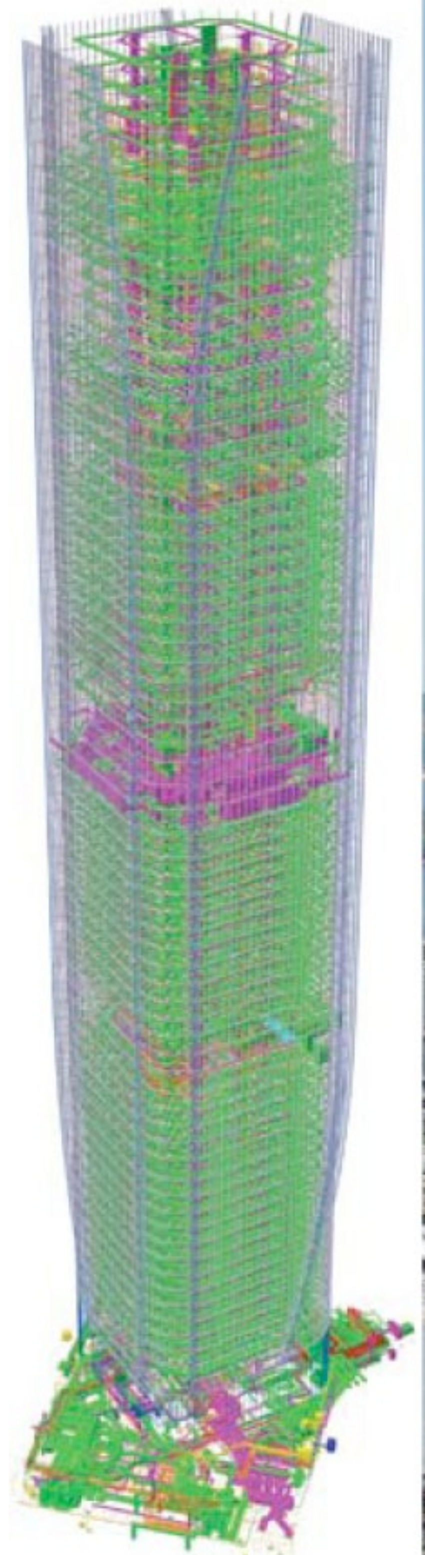
Instance Properties

Family: M_Absorption Chiller - Two Stage - 380

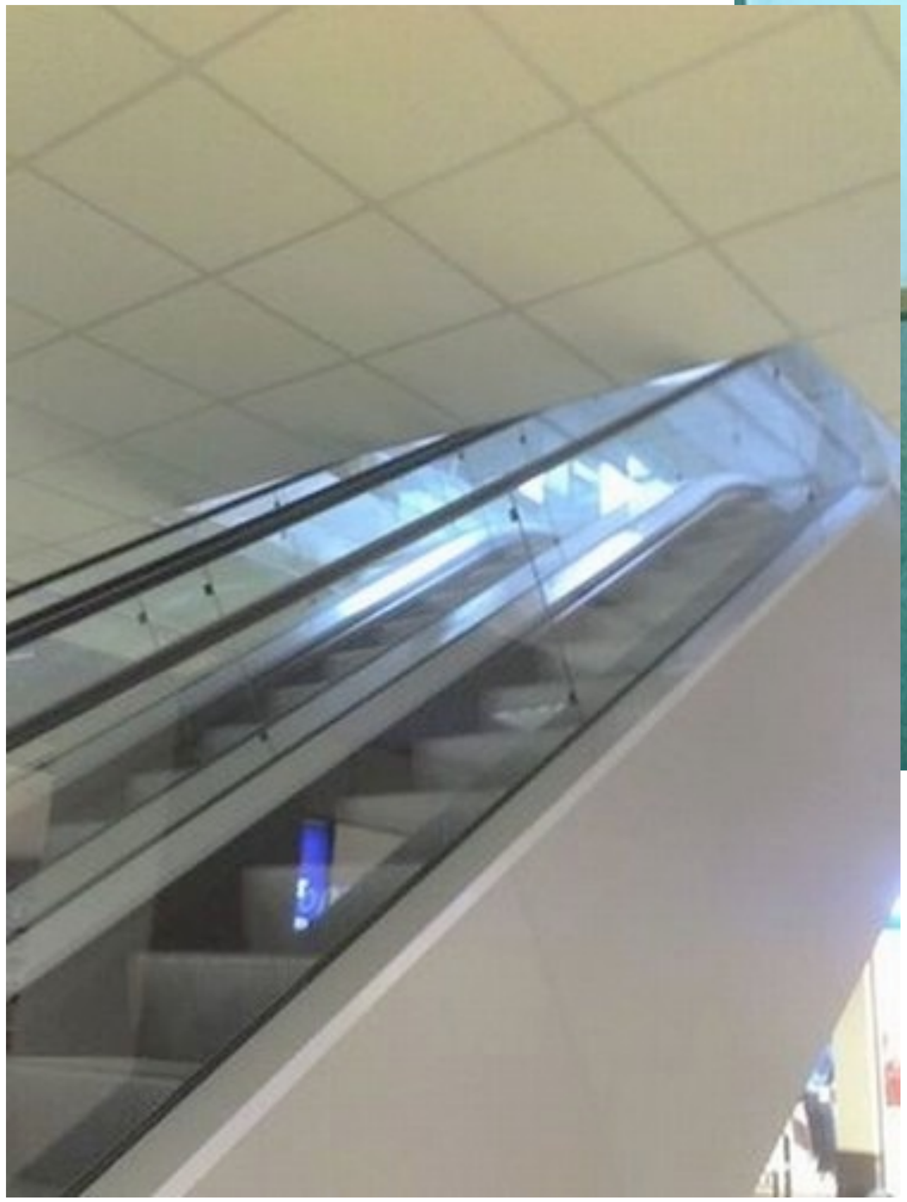
Type: 300 Tons

Instance Parameters - Control selected or to-be-created instance

Parameter	Value
Constraints	
Level	BASEMENT 1
Host	Level : BASEMENT 1
Offset	-1450.0
Electrical - Loads	
Panel	
Circuit Number	
Mechanical	
Steam Flow	535.00 L/s
Cooling Water Pressure Drop	104000.00 Pa
Cooling Water Flow	126.00 L/s
Cooling Capacity	1730.00000 kW
Condensate Flow	37.40 L/s
Chilled Water Pressure Drop	46600.00 Pa
Chilled Water Flow	74.30 L/s
System Type	Hydronic Return,Hydronic Supply,H
System Name	Default Hydronic Return (Hidden)
Reference Number	CH-B-1
Identity Data	
Comments	
Mark	302
Workset	Mechanical Components
Edited by	

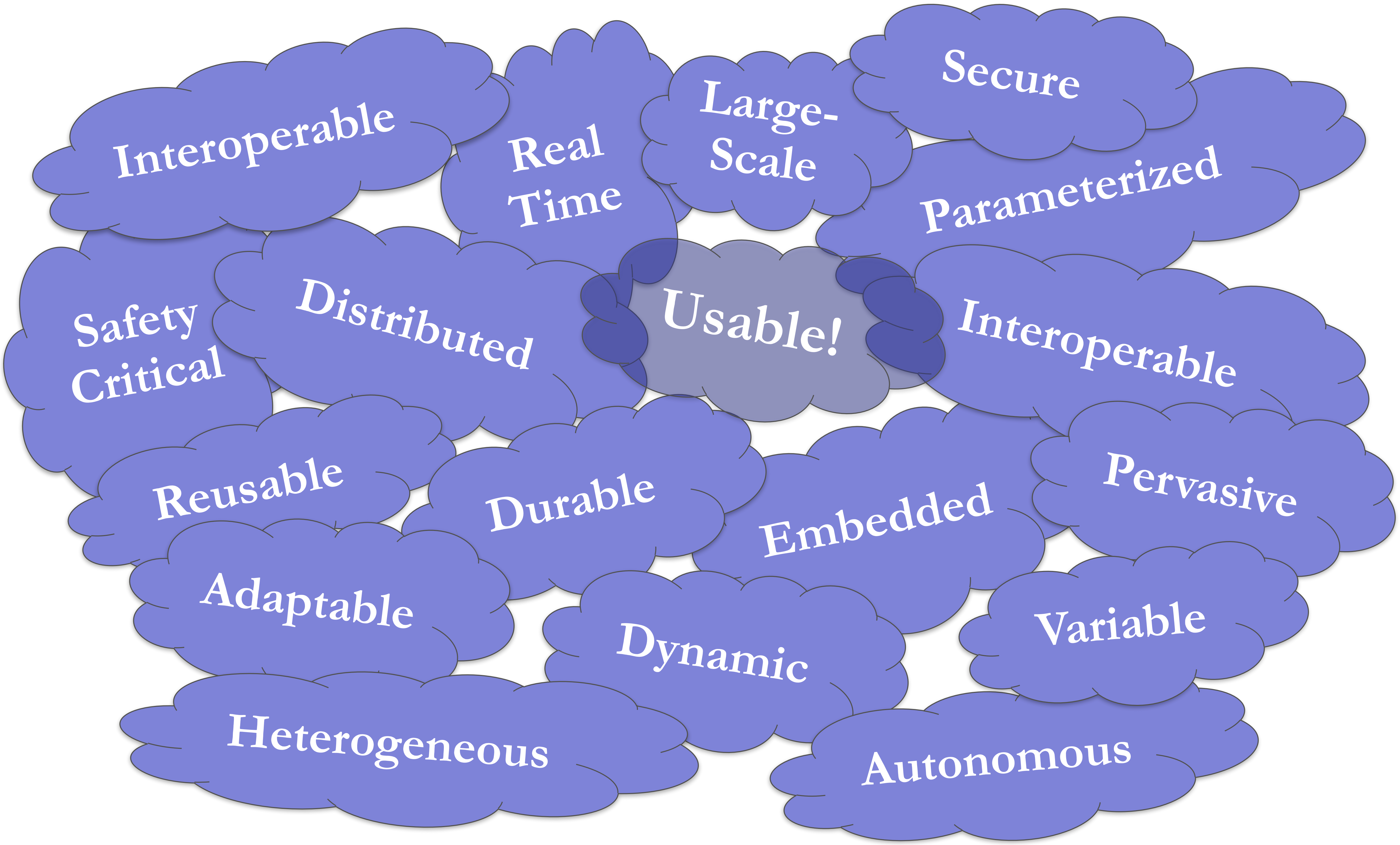


Failures in Civil Engineering!



Failures in Civil Engineering!

Systems Complexity: Some Dimensions



Outline

- From **Software** Engineering to **Systems** Engineering
- SysML: Overview
- SysML Structure Diagrams
- SysML Behavioral Diagrams
- SysML Extensions: Requirement Diagram & Allocation
- Conclusion

Systems Engineering (SE) …

- … is an approach and discipline to deal with complex systems realised through software and hardware solutions.

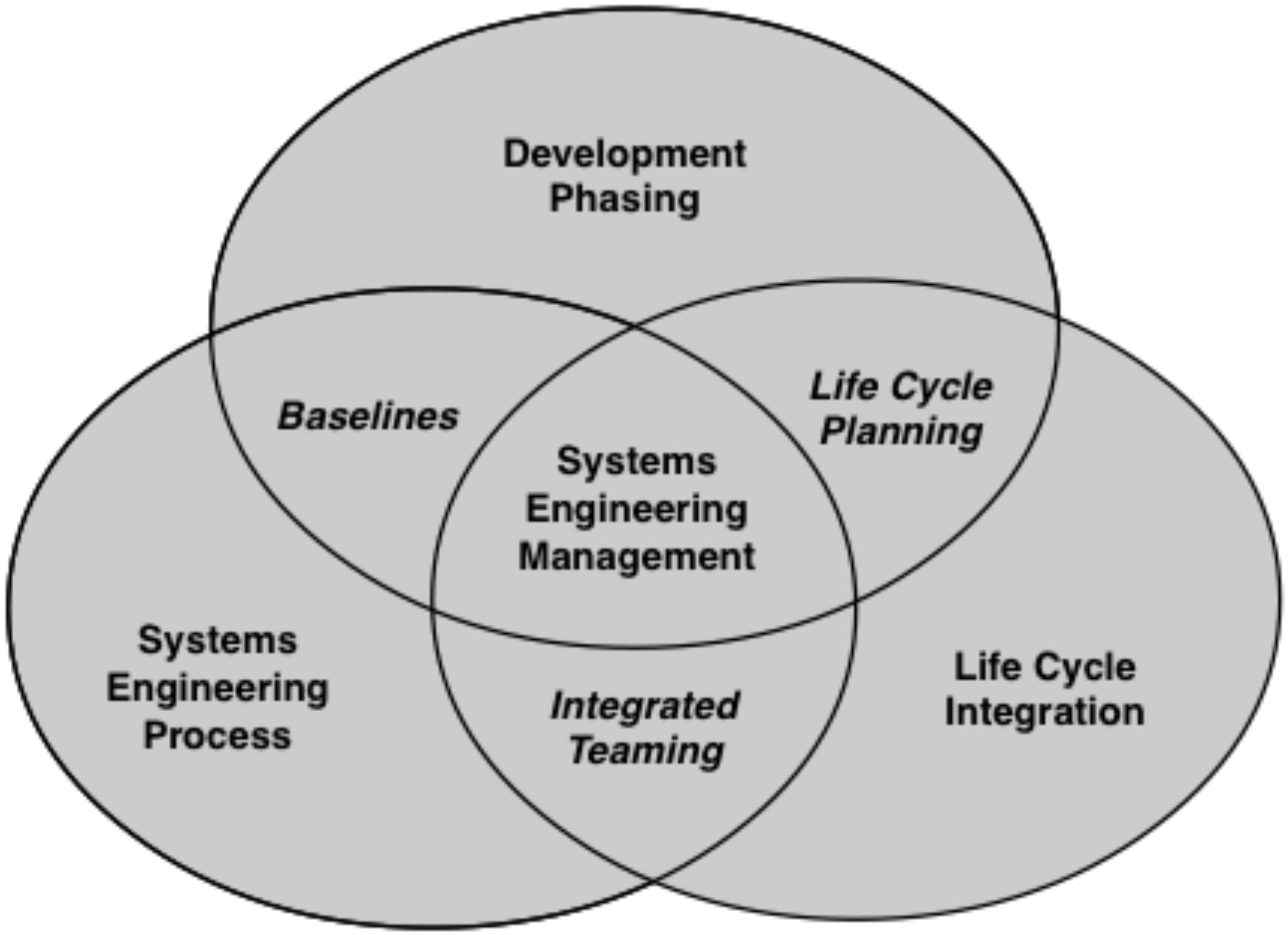
- … relies on modelling and simulation methods to validate requirements or to evaluate the system.

- … applies to the following areas and industries: embedded systems *(e.g. audio and video encoding/decoding, set top box, home automation, smart building, smart city, etc.)*, transport *(automotive, rail, avionics, etc.)*, military, telecom, healthcare, energy, etc.

Systems Engineering (SE) ...

- focuses on:
  - defining customer needs and required functionality early in the development cycle
  - documenting requirements
  - design synthesis and system validation
- considers the complete problem:
  - Operations, Cost & Schedule, Performance, Training & Support, Test, Disposal, Manufacturing...
- integrates all the disciplines and specialty groups that proceeds from concept to production to operation
- considers both the business and the technical needs

Systems Engineering (SE) …

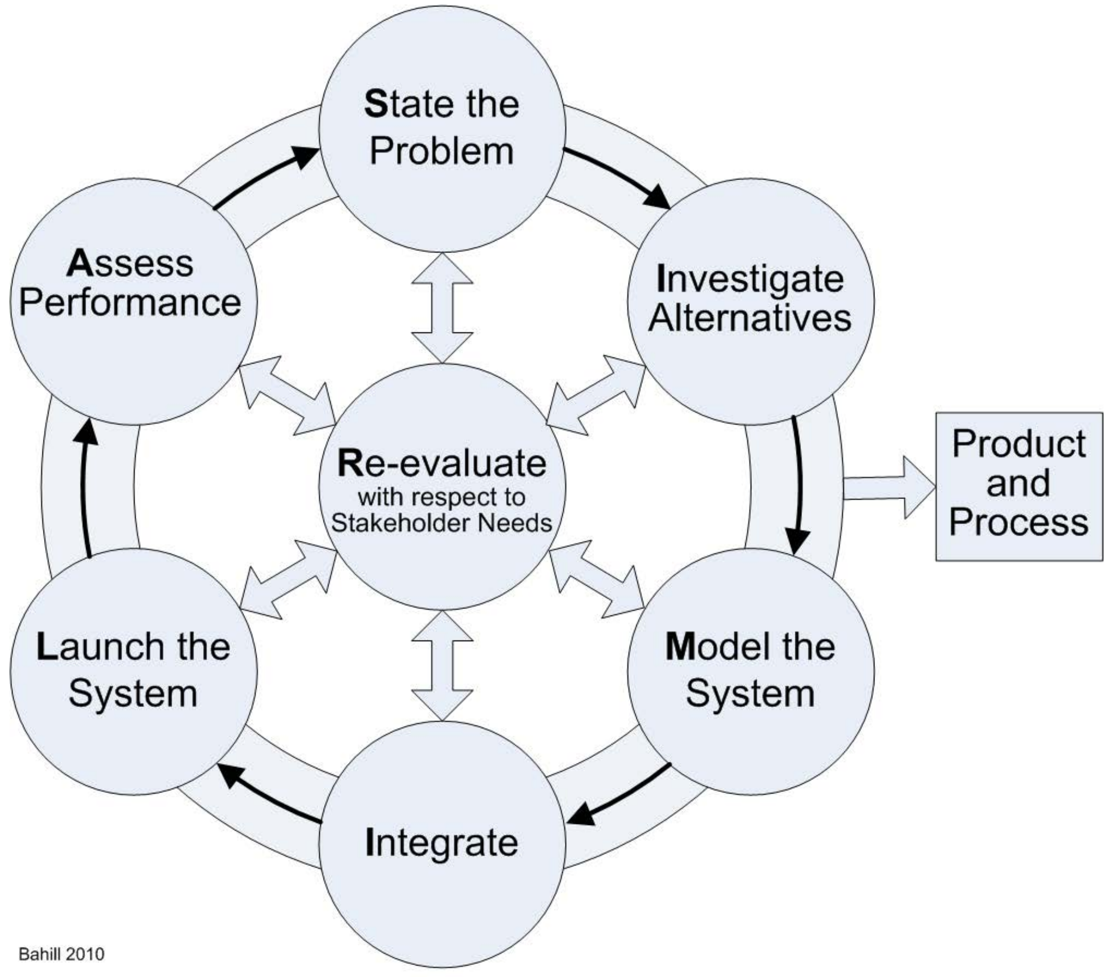


Source: *Systems Engineering Fundamentals*. Defense Acquisition University Press, 2001 (cf. http://www.dau.mil/pubscats/PubsCats/SEFGuide%2001-01.pdf)

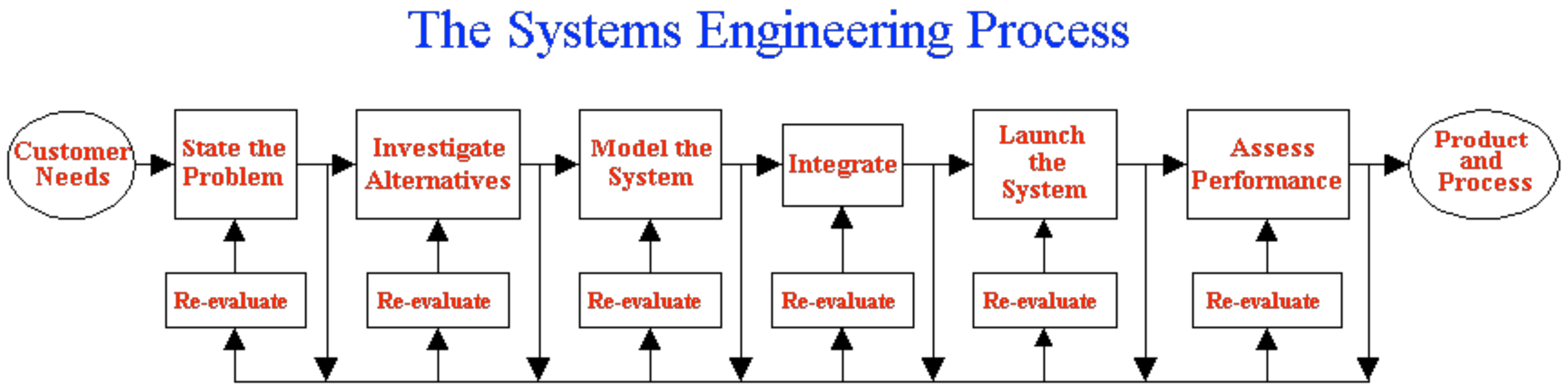
- The International Council on Systems Engineering
- Mission: Share, promote and advance the best of SE
- Vision: The world's authority on Systems Engineering
- Goals:
  - To provide a focal point for dissemination of SE knowledge
  - To promote collaboration in SE practice, education, and research
  - To assure the establishment of competitive, scaleable professional standards in the practice of SE
  - To improve the professional status of all persons engaged in the practice of SE
  - To encourage governmental and industrial support for research and educational
- *Cf.* http://www.incose.org/

The SIMILAR Process (INCOSE)

- State the problem
- Investigate alternatives
- Model the system
- Integrate
- Launch the system
- Assess performance
- Re-evaluate



The SIMILAR Process



Outline

The advent of SysML…

- *01/1997 : UML v1.0*
- 2001 : INCOSE & OMG form the Systems Engineering Domains Special Interest Group (SE DSIG)
- 03/2003 : UML for Systems Engineering RFP
- *06/2003 : MDA Guide v1.0.1*
- *01/2005 : UML v1.4.2 (ISO/IEC 19501)*
- *07/2005 : UML v2.0*
- 07/2006 : SysML is officially adopted by the OMG
- 09/2007 : SysML v1.0
- 11/2008 : SysML v1.1
- *08/2011 : UML v2.4.1 (current version)*
- 06/2012 : SysML v1.3 (current version)

SysML: Who is behind?

- Industry
  - *American Systems, BAE Systems, Boeing, Deere & Company, EADS Astrium, Eurostep, Israel Aircraft Industries, Lockheed Martin, Motorola, NIST, Northrop Grumman, oose.de, Raytheon, Thales, …*
- Tool vendors
  - *Artisan, EmbeddedPlus, Gentleware, IBM, I-Logix, Mentor Graphics, PivotPoint Technology, Sparx Systems, Telelogic, vitech, …*
- Other organisations
  - *AP-233, INCOSE, Georgia Institute of Technology, AFIS, …*

SysML: a modeling language for SE

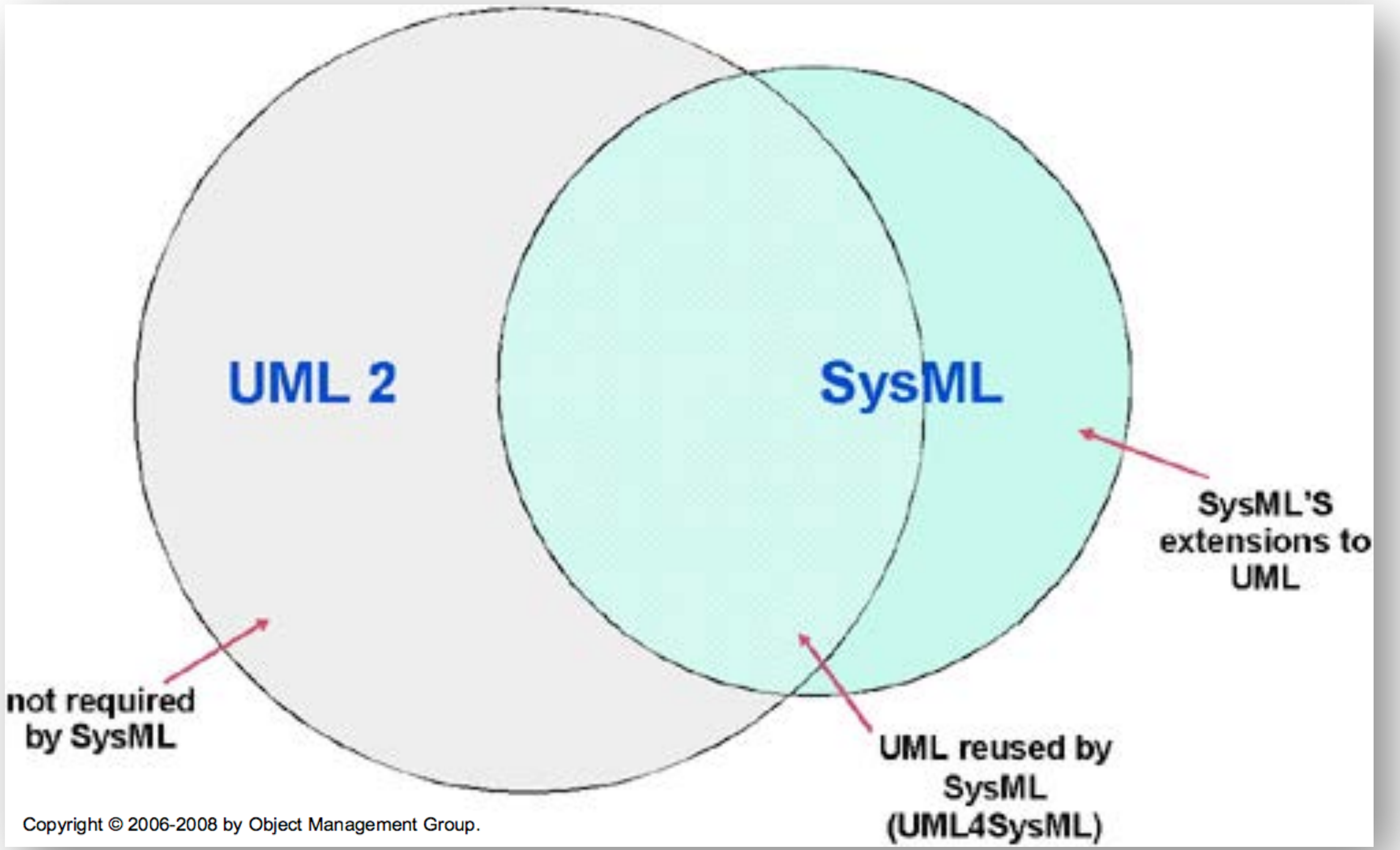
- Standard modeling language for SE to **analyze**, **specify**, **design**, and **verify** complex systems
- Intended to
  - enhance systems quality
  - improve the ability to exchange systems engineering information amongst tools
  - help bridge the semantic gap between systems, software, and other engineering disciplines

SysML Overview

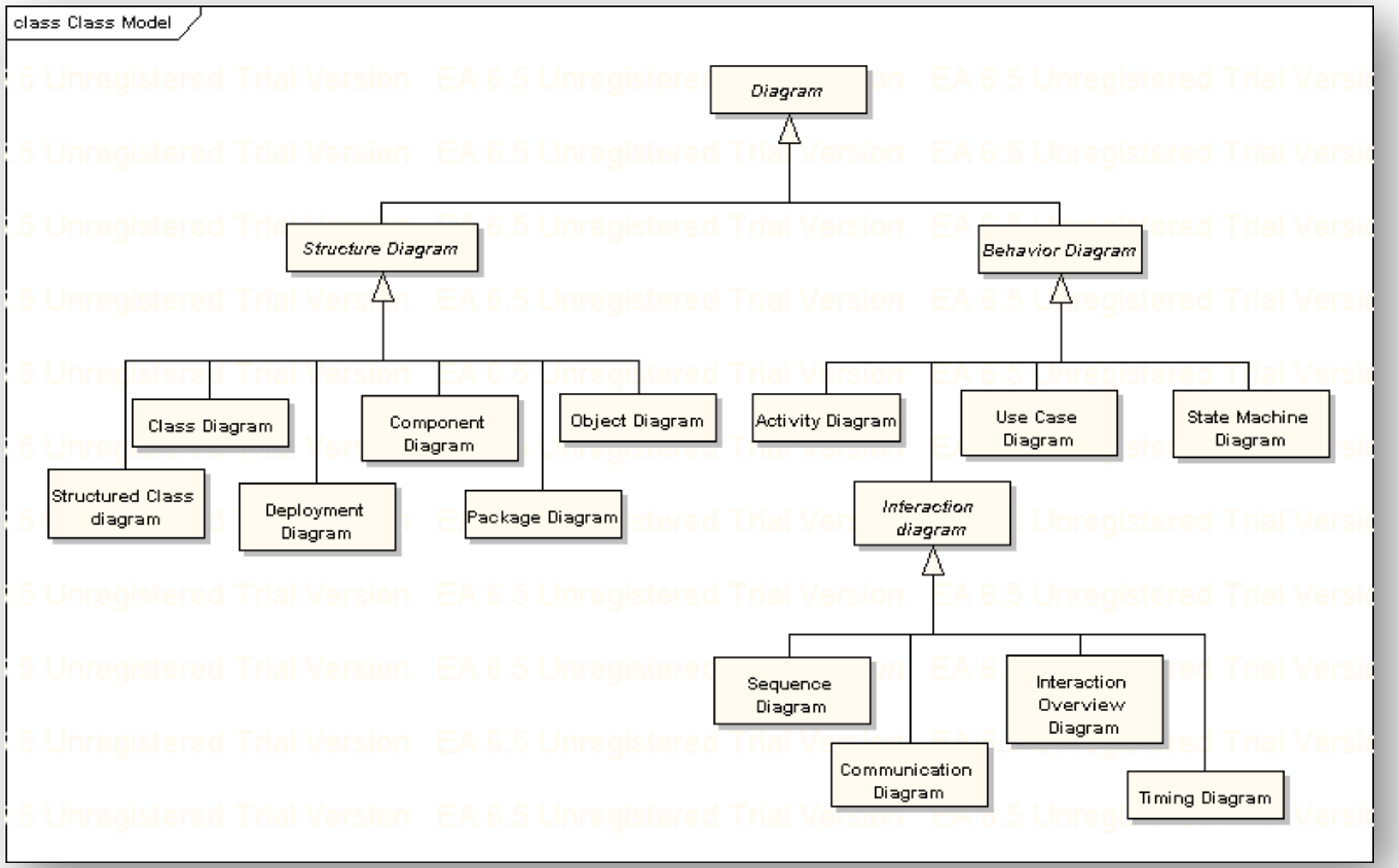
- is based on UML (v2.x)
- involves modeling blocks instead of modeling classes
- provides a vocabulary that's more suitable for SE

SysML Overview

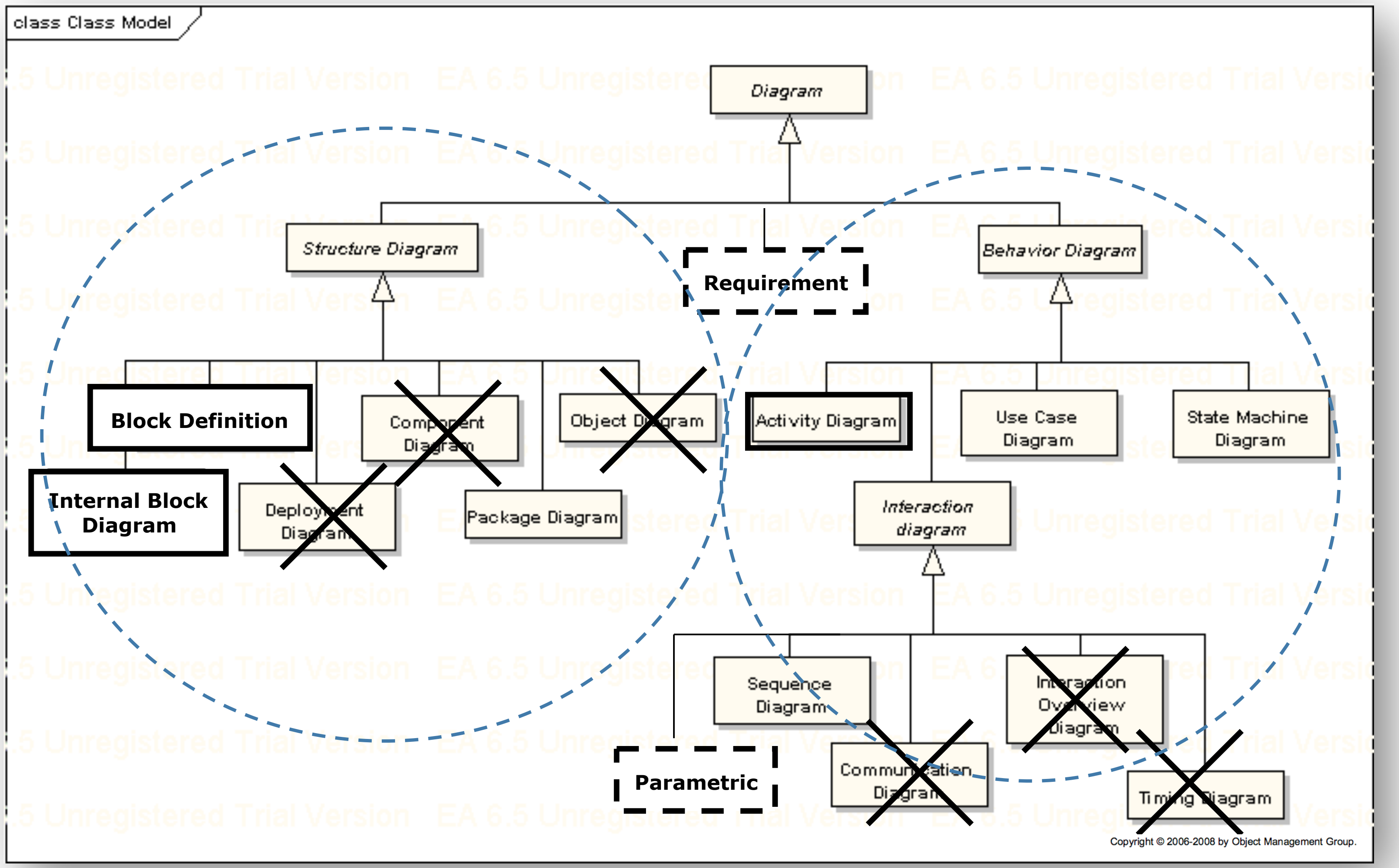
- **SysML**: the OMG Systems Modeling Language
  - ➡ **Systems** Engineering
- **UML**: the OMG Software Modeling Language
  - ➡ **Software** Engineering


UML 2
SysML
SysML'S extensions to UML
not required by SysML
UML reused by SysML (UML4SysML)
Copyright © 2006-2008 by Object Management Group.

UML: 13 diagrams



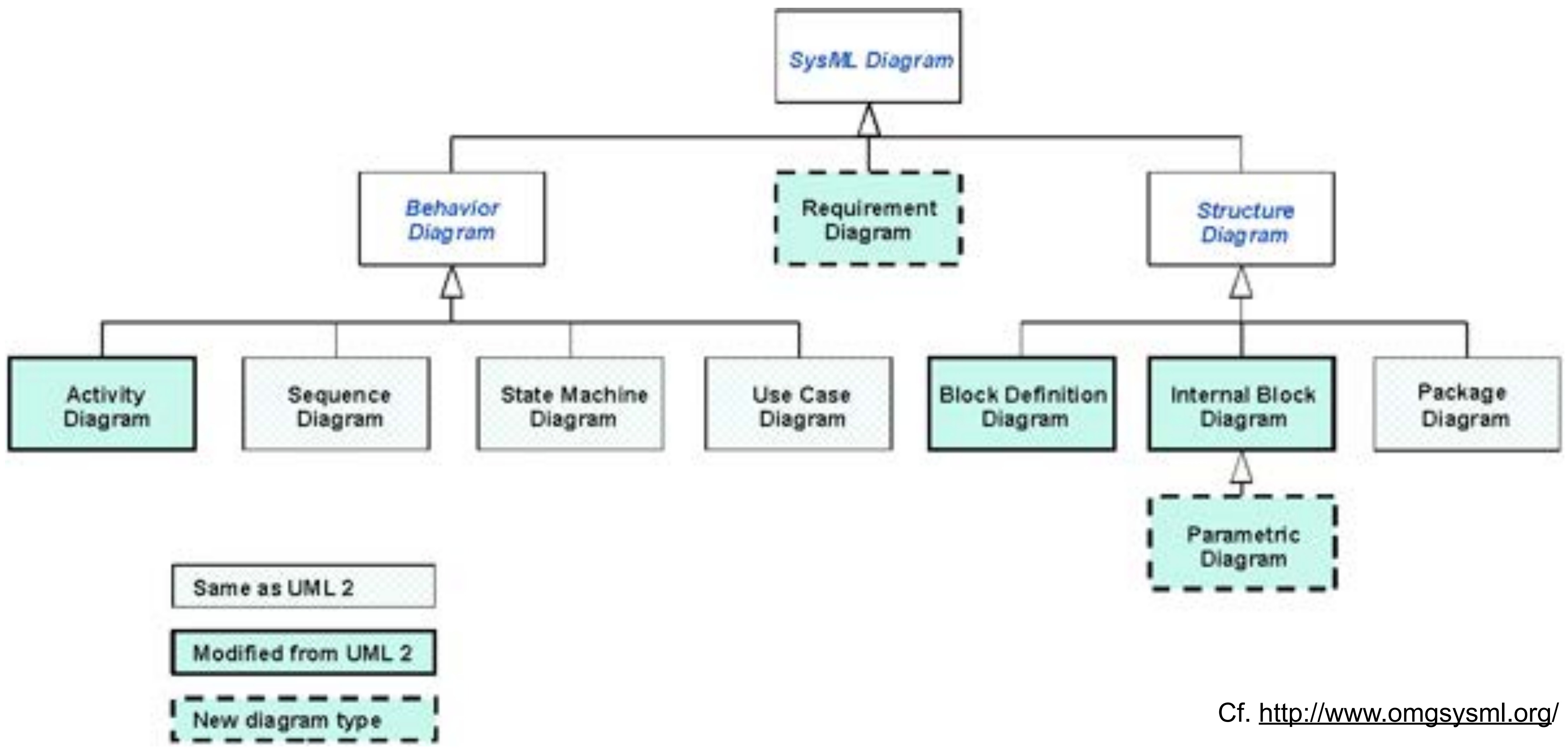
SysML: 13-7+2 = 9 diagrams



SysML diagrams

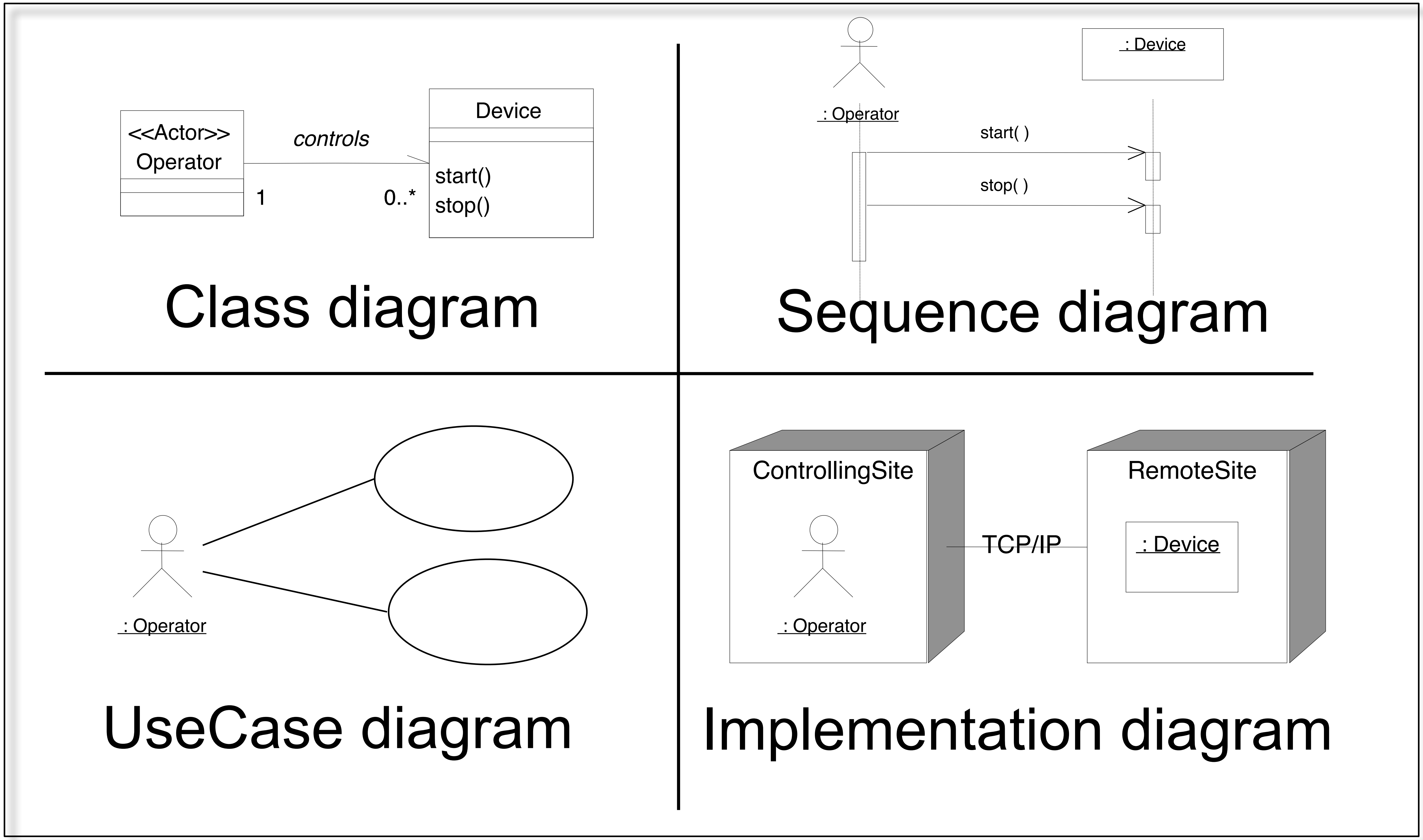
- Structure diagrams
  - The Block Definition Diagram (BDD), replacing the UML2 class diagram
  - The Internal Block Diagram (IBD), replacing the UML2 composite structure diagram
  - The Parametric Diagram, a SysML extension to analyse critical system parameters
  - The Package Diagram remains unchanged
- Dynamic diagrams
  - The activity diagram has been slightly modified in SysML
  - The sequence, state chart, and use case diagrams remain unchanged
- The requirements diagrams is a SysML extension

SysML diagrams

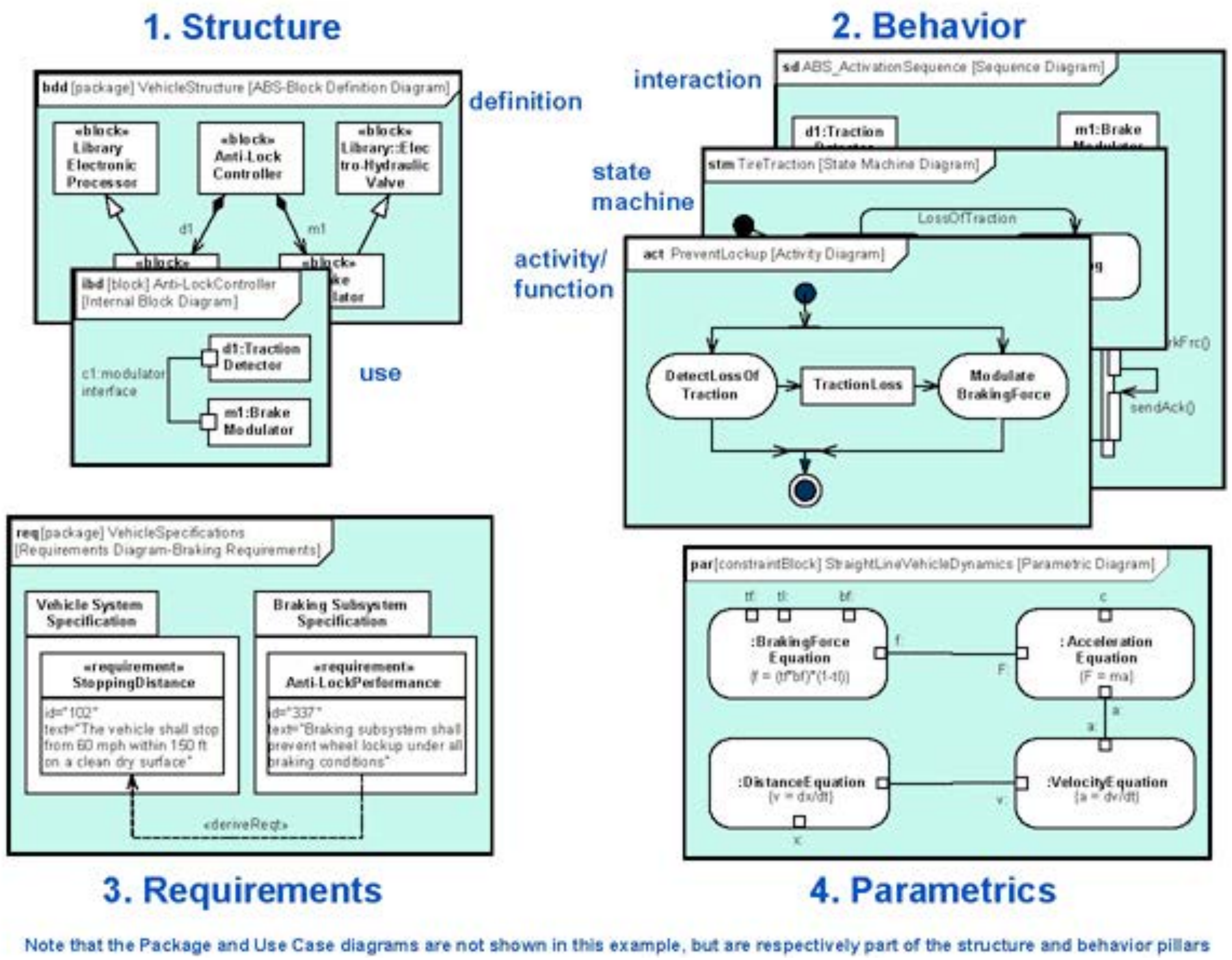


Cf. http://www.omgsysml.org/

The Four Pillars of UML



The Four Pillars of SysML

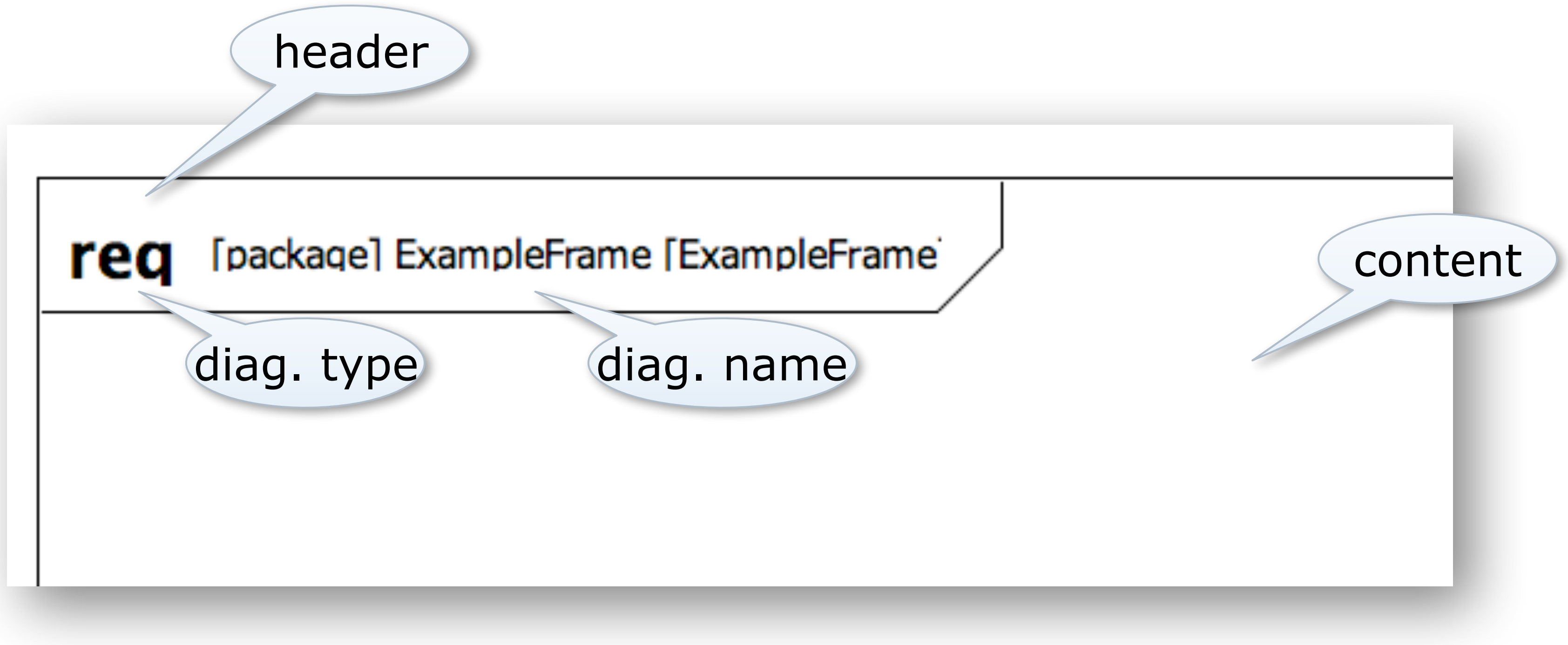

1. Structure
2. Behavior
3. Requirements
4. Parametrics
definition
use
interaction
state machine
activity/ function
bdd [package] VehicleStructure [ABS-Block Definition Diagram]
«block» Library Electronic Processor
«block» Anti-Lock Controller
«block» Library::Electro-Hydraulic Valve
d1
m1
ibd [block] Anti-LockController [Internal Block Diagram]
d1:Traction Detector
m1:Brake Modulator
c1:modulator interface
sd ABS_ActivationSequence [Sequence Diagram]
stm TireTraction [State Machine Diagram]
LossOfTraction
act PreventLockup [Activity Diagram]
DetectLossOf Traction
TractionLoss
Modulate BrakingForce
sendAck()
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"
«requirement» Anti-LockPerformance
id="337" text="Braking subsystem shall prevent wheel lockup under all braking conditions"
«deriveReqt»
par [constraintBlock] StraightLineVehicleDynamics [Parametric Diagram]
:BrakingForce Equation [f = (tf*bf)*(1-tl)]
:Acceleration Equation [F = ma]
:DistanceEquation [v = dx/dt]
:VelocityEquation [a = dv/dt]
Note that the Package and Use Case diagrams are not shown in this example, but are respectively part of the structure and behavior pillars

Cf. http://www.omgsysml.org/

SysML diagram frames

- Each SysML diag. represents a model element
- Each SysML diag. must have a Diagram Frame
- Diagram context is indicated in the header:
  - Diagram kind (req, act, bdd, ibd, sd, etc.)
  - Model element type (package, block, activity, etc.)
  - Model element name
  - User defined diagram name or view name
- A separate diagram description block is used to indicate if the diagram is complete, or has elements elided

SysML diagram frames (e.g.)

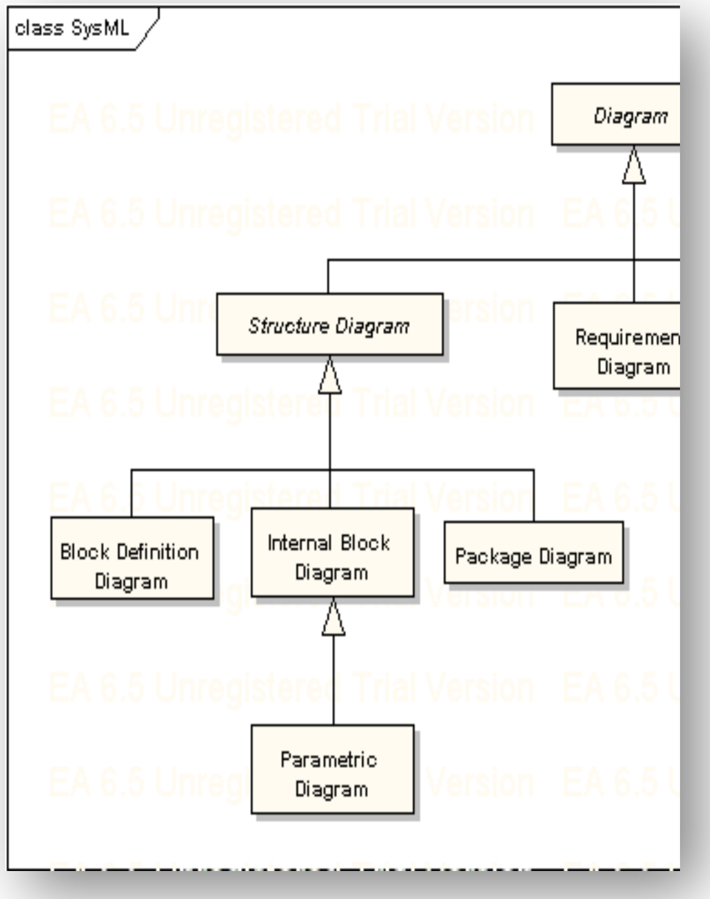


Outline

- From **Software** Engineering to **Systems** Engineering
- SysML: Overview
- SysML Structure Diagrams
- SysML Behavioral Diagrams
- SysML Extensions: Requirement Diagram & Allocation
- Conclusion

SysML Structure Diagrams

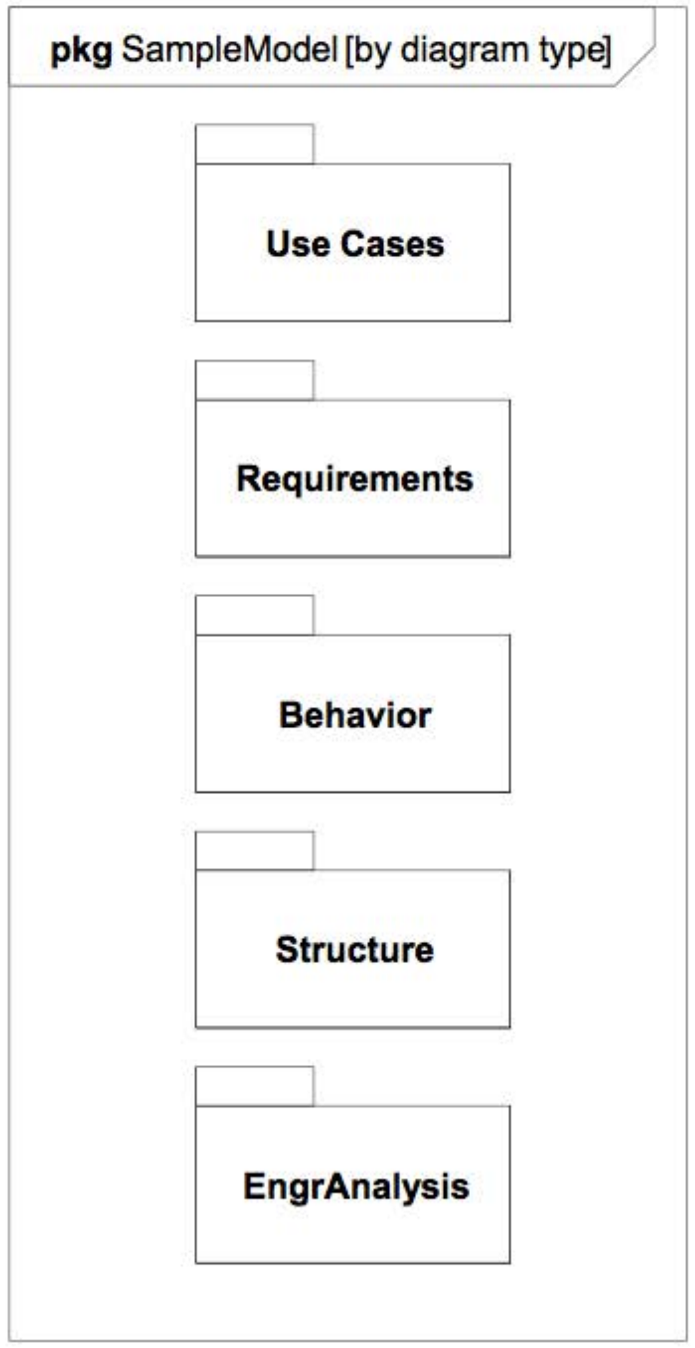
- Package Diagram
- Block Definition Diagram
- Internal Block Definition Diagram
- Parametric Diagram



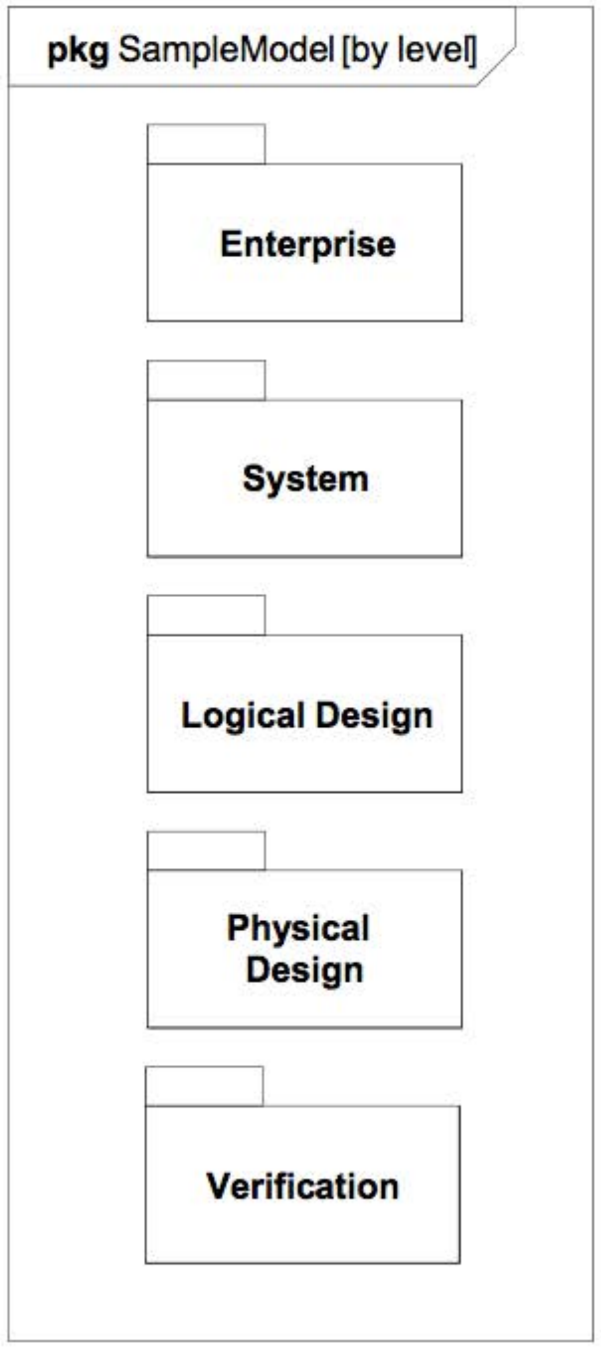
Package Diagram (pkg)

- ~Same as UML
- To organize the model
  - Groups model elements into a name space
  - Often represented in tool browser
  - Supports model configuration management (check-in/out)
- Model can be organized in multiple ways:
  - System hierarchy (e.g., enterprise, system, component)
  - Diagram kind (e.g., requirements, use cases, behavior)
  - Use viewpoints to augment model organization
- Value Types: reusable types for properties or attributes in the model (new SysML extension)

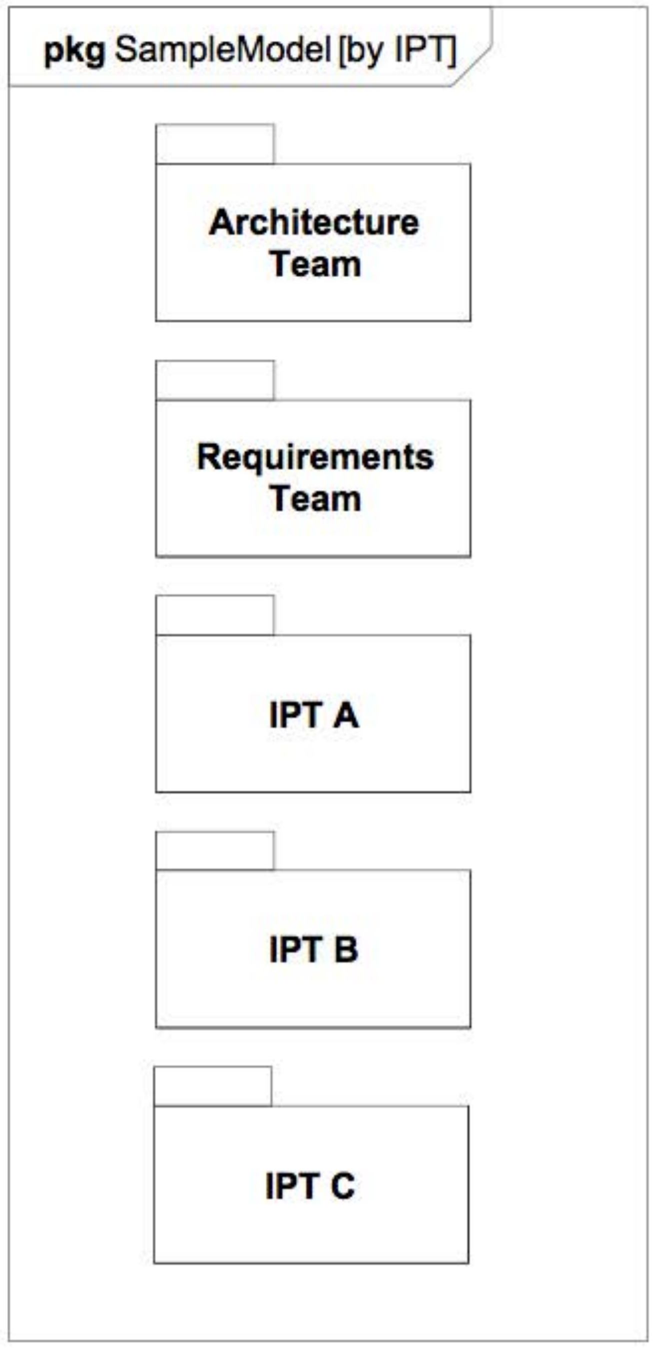
Package Diagram (pkg)



By Diagram Type

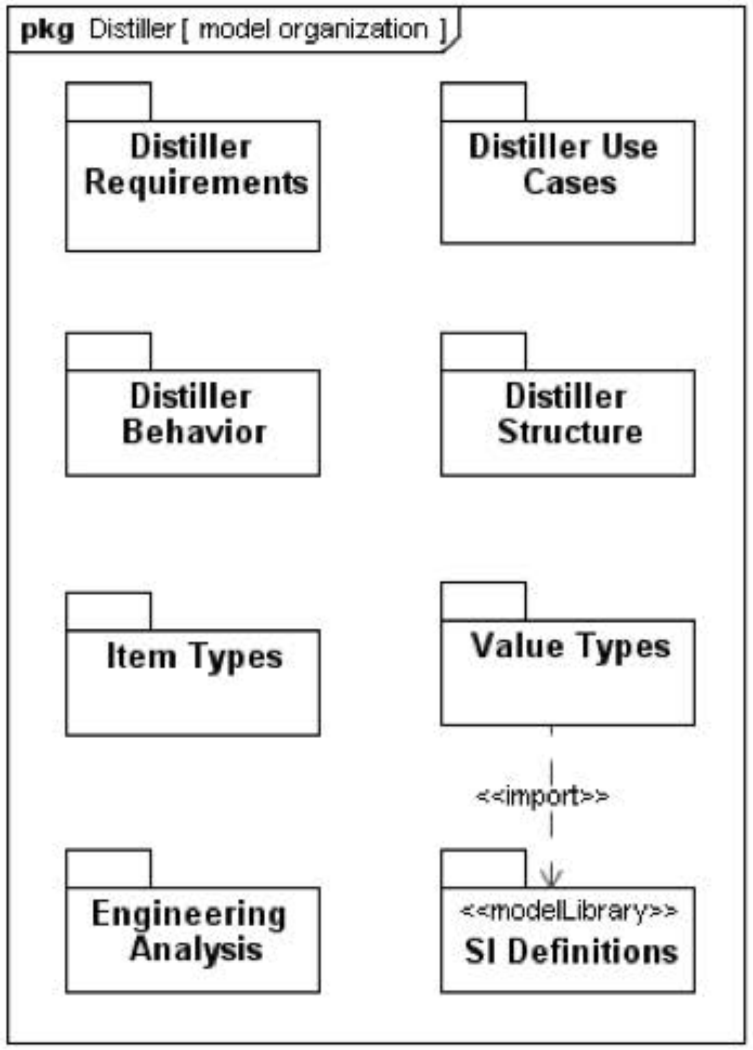


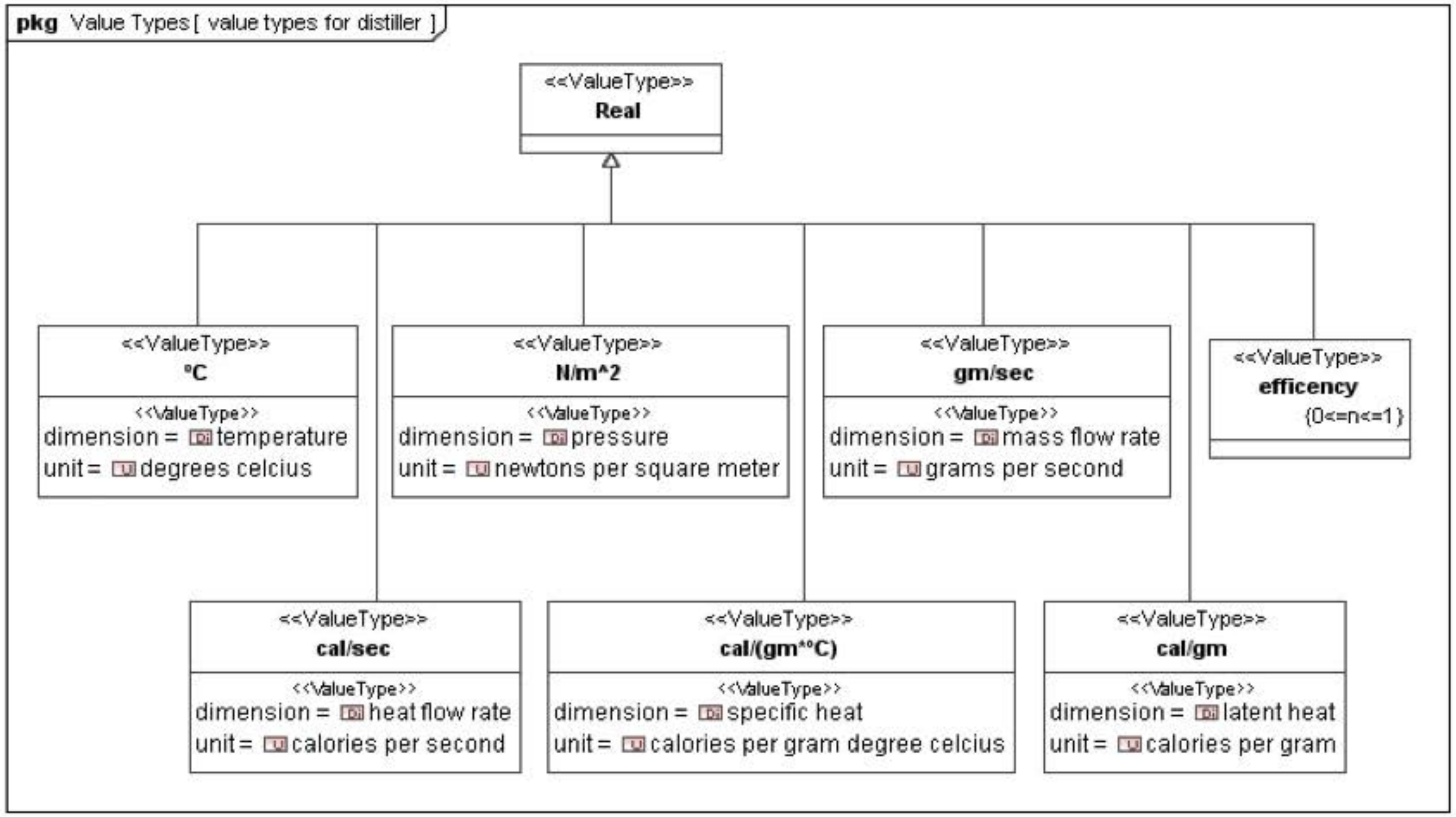
By Hierarchy



By IPT

OMG Distiller Example (pkg)



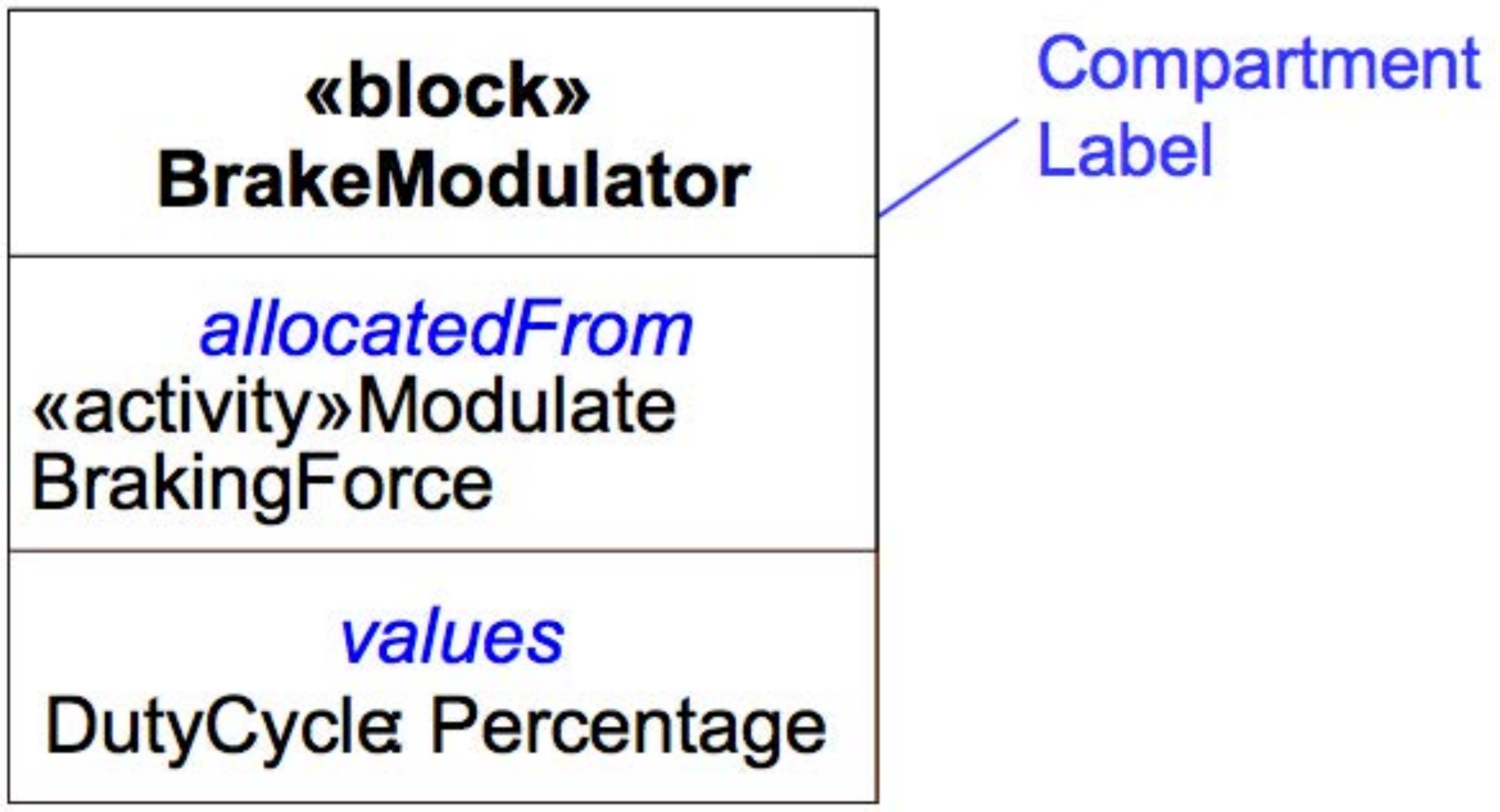


Block Definition Diagram (bdd)

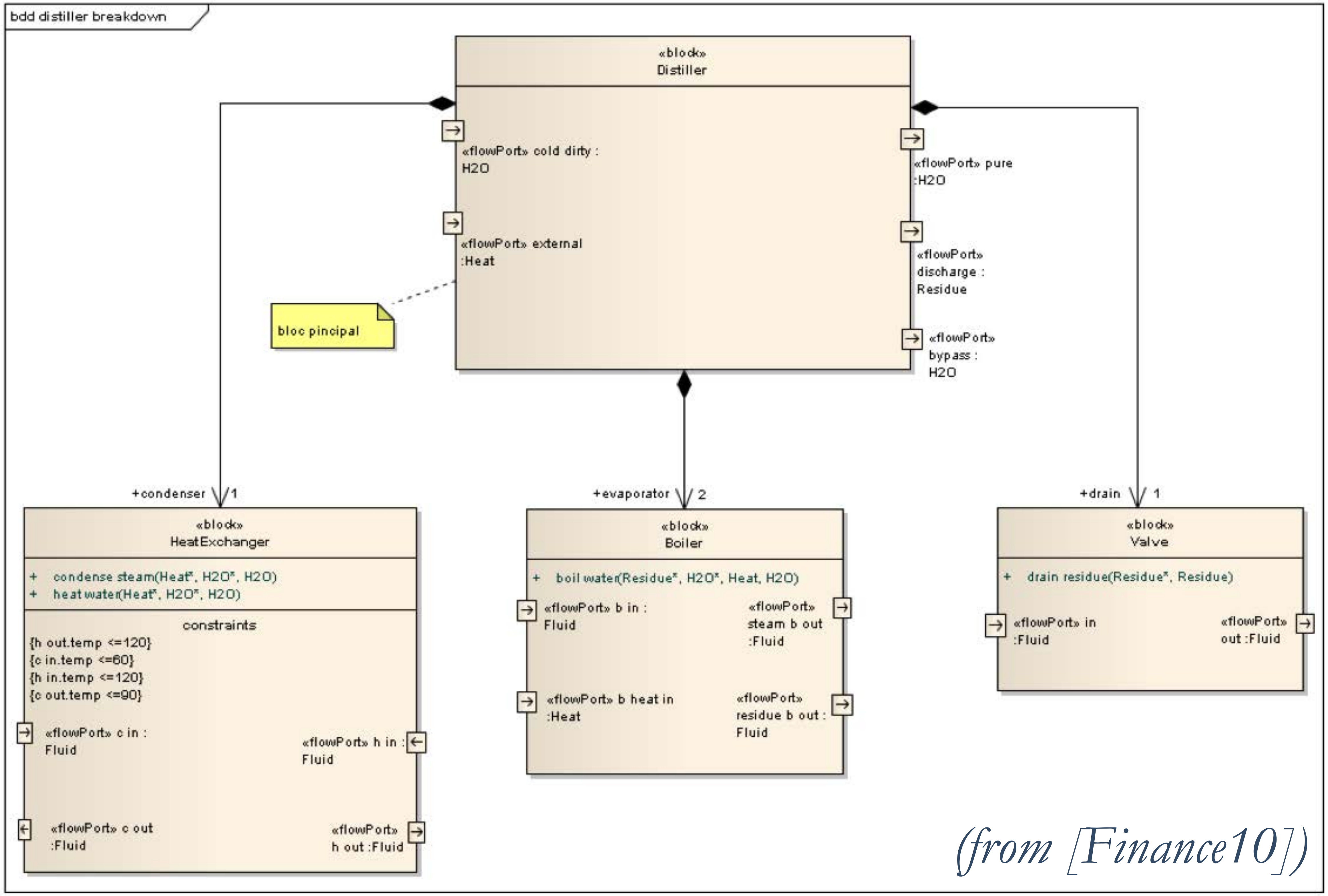
- The BDD provides a black box representation of a system block alongside the hierarchy of its composite blocks.
- The BDD can include blocks of any type including software, hardware, etc.
- A block
  - provides a unifying concept to describe the structure of an element or system
  - encompasses software, hardware, data, processes, personnel, and facilities.
  - is shown as a UML class, stereotyped « block ».

SysML Block

- Compartments
  - Properties
  - Operations
  - Constraints
  - Allocations
  - Requirements
- User defined!



OMG Distiller Example (bdd)



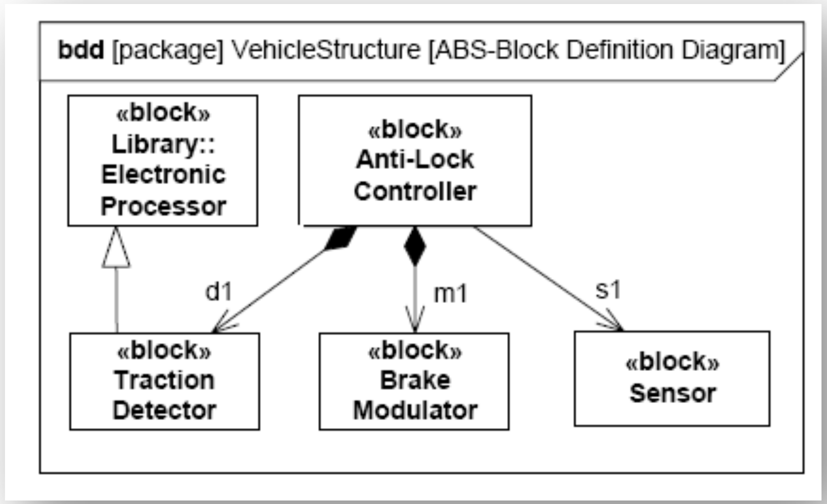
(from [Finance10])

Internal Block Diagram (ibd)

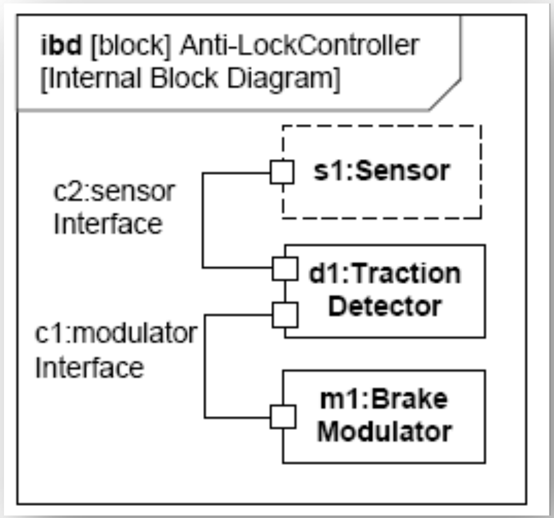
- Provides the white box or internal view of a system block
- Usually instantiated from the BDD to represent the final assembly
- Composite blocks from the BDD are instantiated on the IBD as parts
- Parts are assembled through connectors, linking them directly or via their ports (standard and/or flow ports)
- Redefines the UML2 composite structure diagram with blocks and flow ports.

Block Definition vs. Usage

- Definiton:
  - Block is a definition/type
  - Capture properties, etc.
  - Reused in multiple contexts
- Usage:
  - Part is the usage of a block in the context of a composing block
  - Also known as a role

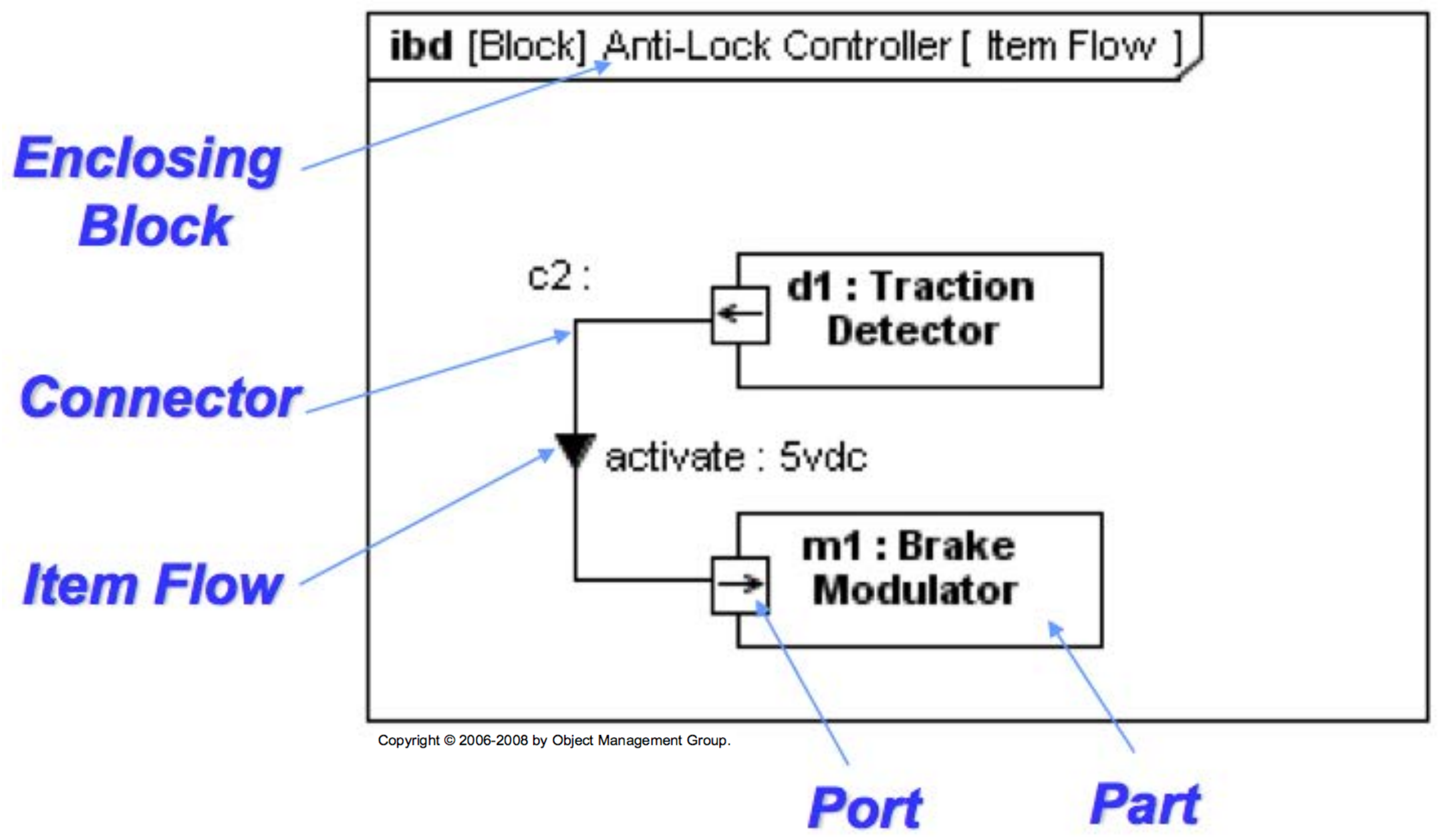

bdd [package] VehicleStructure [ABS-Block Definition Diagram]
«block» Library:: Electronic Processor
«block» Anti-Lock Controller
d1
m1
s1
«block» Traction Detector
«block» Brake Modulator
«block» Sensor

Block Definition Diagram



Internal Block Diagram

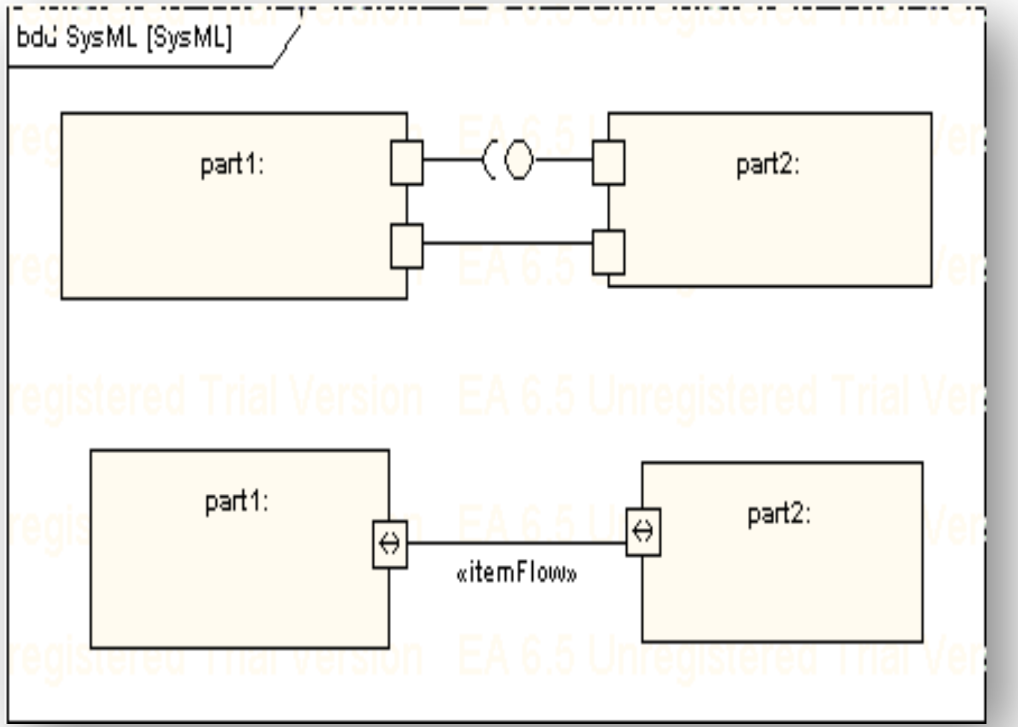
Internal Block Diagram (ibd)



SysML Ports

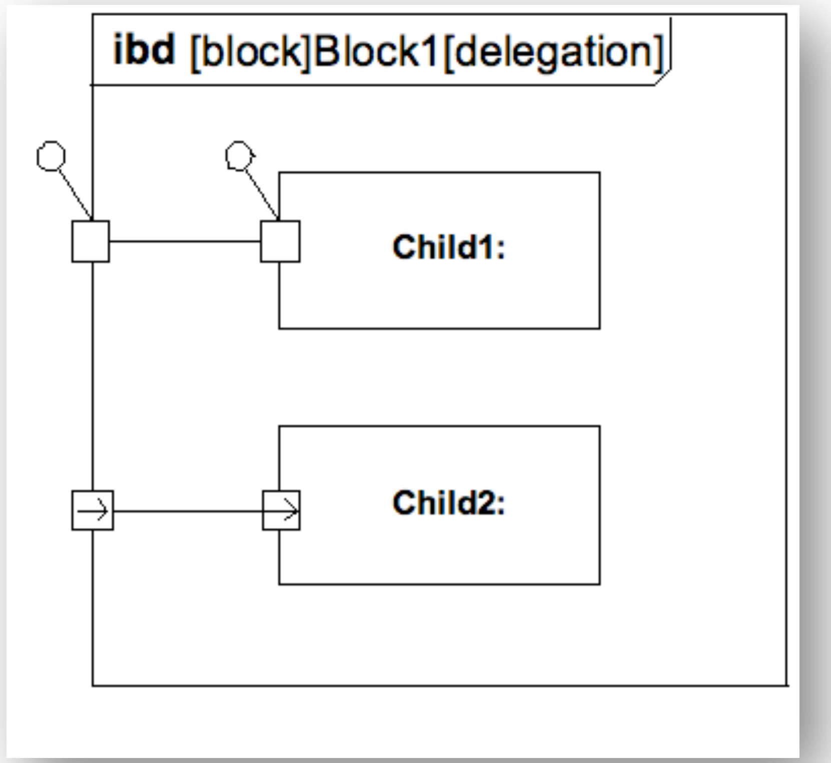
- **Specifies interaction points on blocks and parts**
  - ➡ Integrates behavior with structure

- Standard (UML) Port:
  - Specifies a set of required or provided operations and/or signals
  - Typed by a UML interface
- Flow Port:
  - Specifies what can flow in or out of block/part
  - Typed by a block, value type, or flow specification
  - Atomic, non-atomic, and conjugate variations

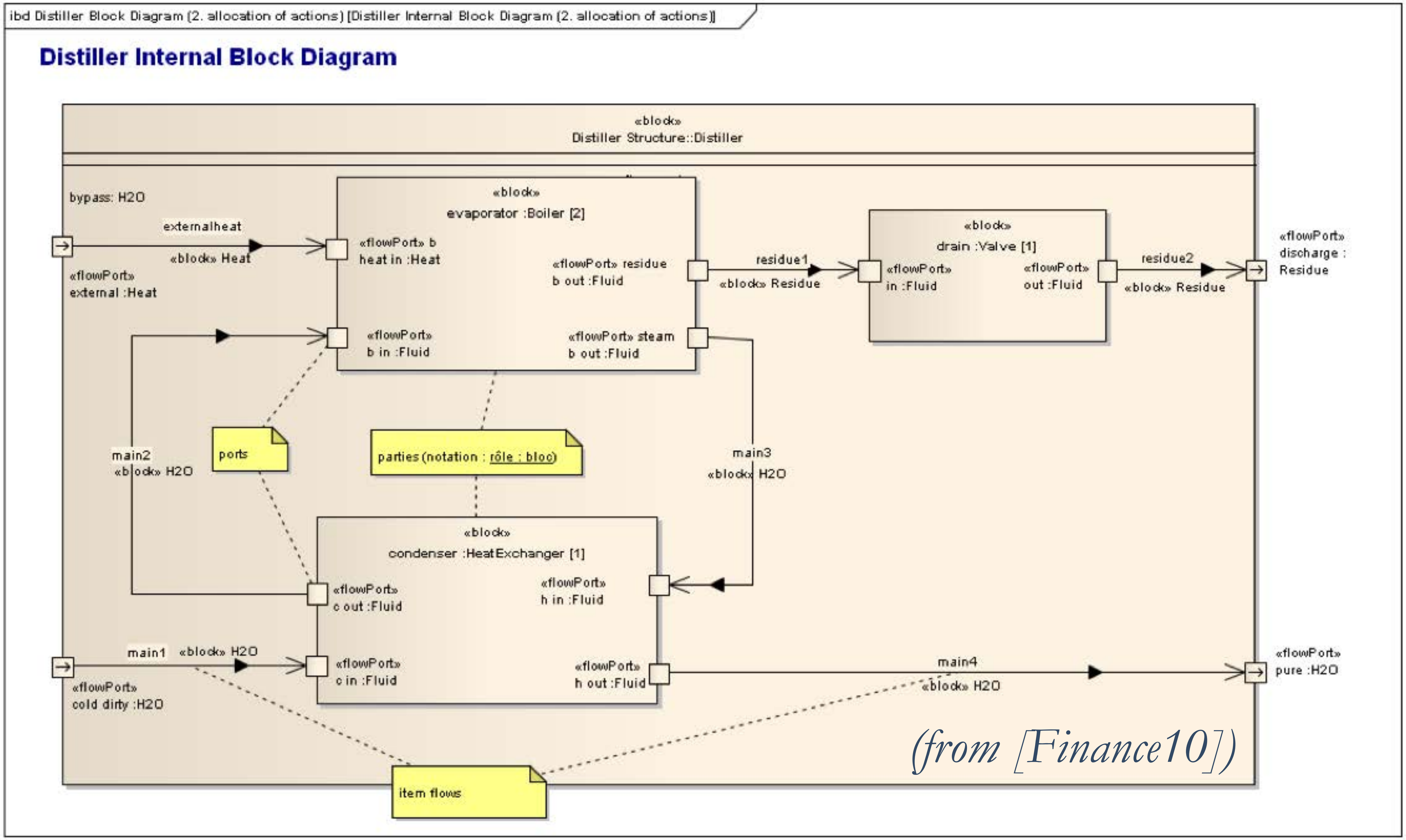


SysML Ports: **delegation**

- to preserve **encapsulation** of block
- interactions at outer ports are **delegated** to ports of child parts
- ports must **match**
  - same kind, type, direction, etc.
- connectors can **cross boundary** without requiring ports at each level of nested hierarchy



OMG Distiller Example (ibd)



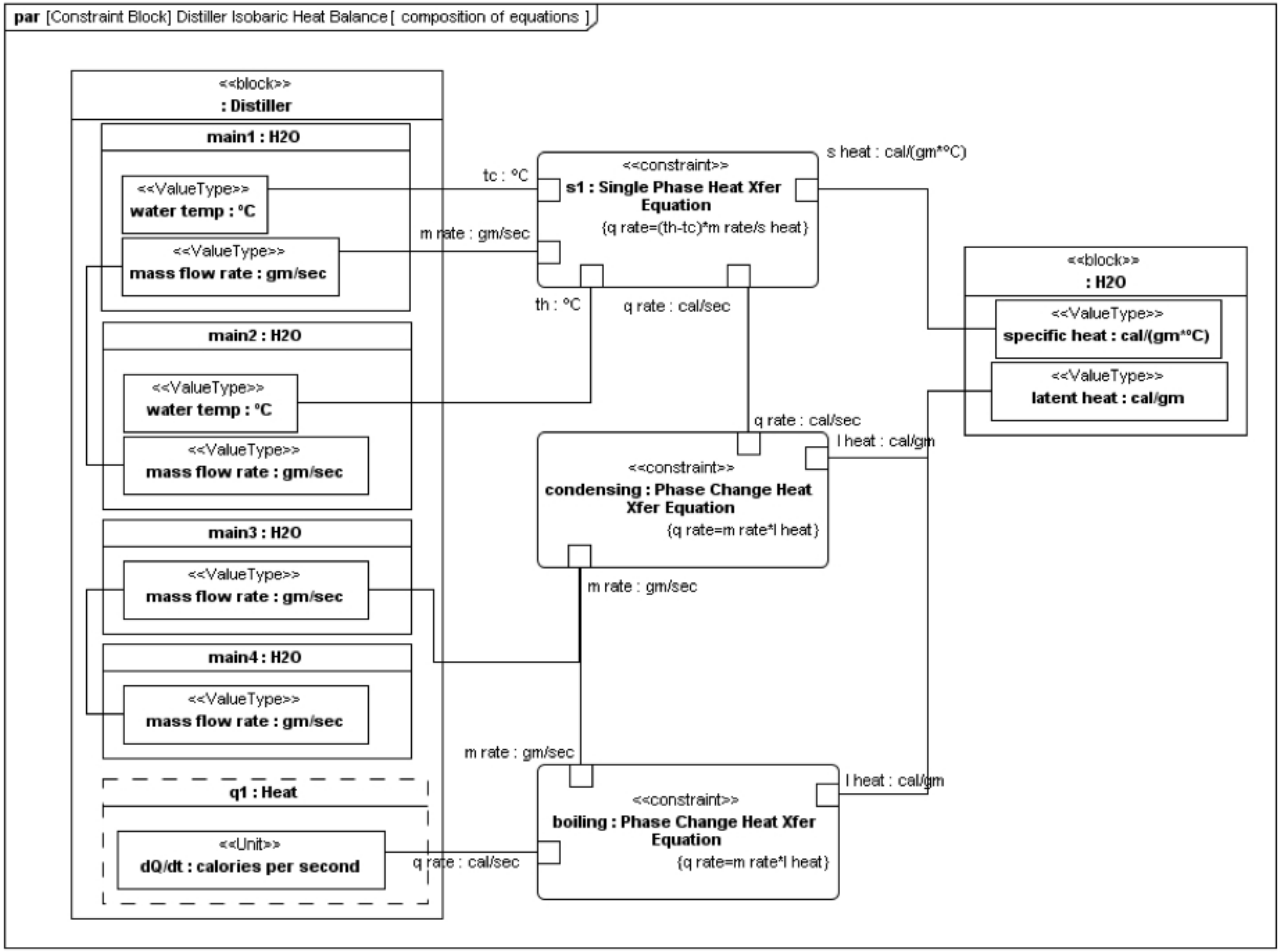
Parametric Diagram (par)

(SysML extension)

- To express constraints between value properties
  - equations
  - support for engineering analysis (e.g., performance)
  - identification of critical performance properties
- Constraint block captures equations
  - Expression language can be formal (e.g., MathML, OCL)
  - Computational engine is not provided by SysML
- Parametric diagram
  - usage of the constraints in an analysis context

SM@RT

OMG Distiller Example (par)

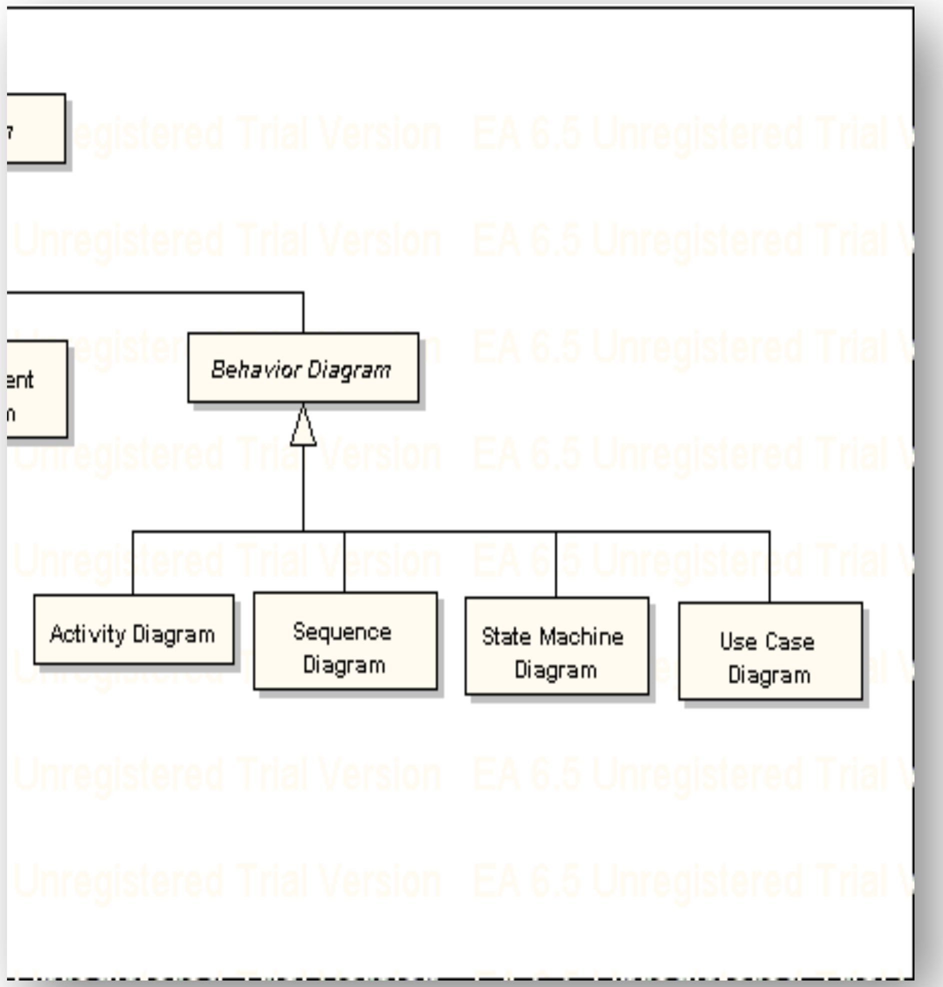


Outline

- From **Software** Engineering to **Systems** Engineering
- SysML: Overview
- SysML Structure Diagrams
- SysML Behavioral Diagrams
- SysML Extensions: Requirement Diagram & Allocation
- Conclusion

SysML Behavioral Diagrams

- Activity Diagram
- Sequence Diagram
- State Machine Diagram
- Use Case Diagram



Activity Diagram (act)

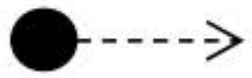
- to specify
  - controlled sequence of actions
  - the **flow** of inputs/outputs
  - **control**, including sequence and conditions for coordinate activities

- Swimlanes
  - to show **responsibility** of the activity

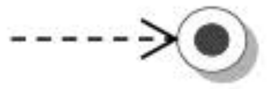
Activity Diagram (act)

- Improvements from UML:
  - **continuous** or discrete flow
  - **control** operators
    - to start/stop other actions
  - **Overwrite** and **NoBuffer** ports
    - for continuous flows
  - **probabilities** on transitions or parameter

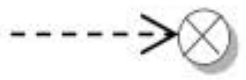
Routing Flow



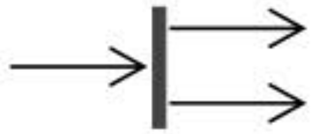
Initial Node – On execution of parent control token placed on outgoing control flows



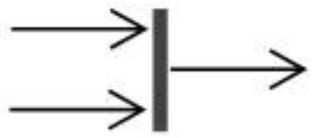
Activity Final Node – Receipt of a control token terminates parent



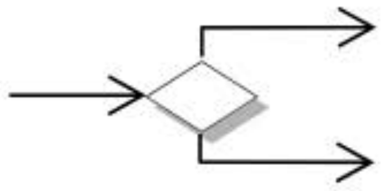
Flow Final Node – Sink for control tokens



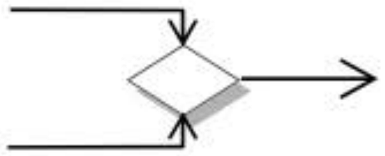
Fork Node – Duplicates input (control or object) tokens from its input flow onto all outgoing flows



Join Node – Waits for an input (control or object) token on all input flows and then places them all on the outgoing flow



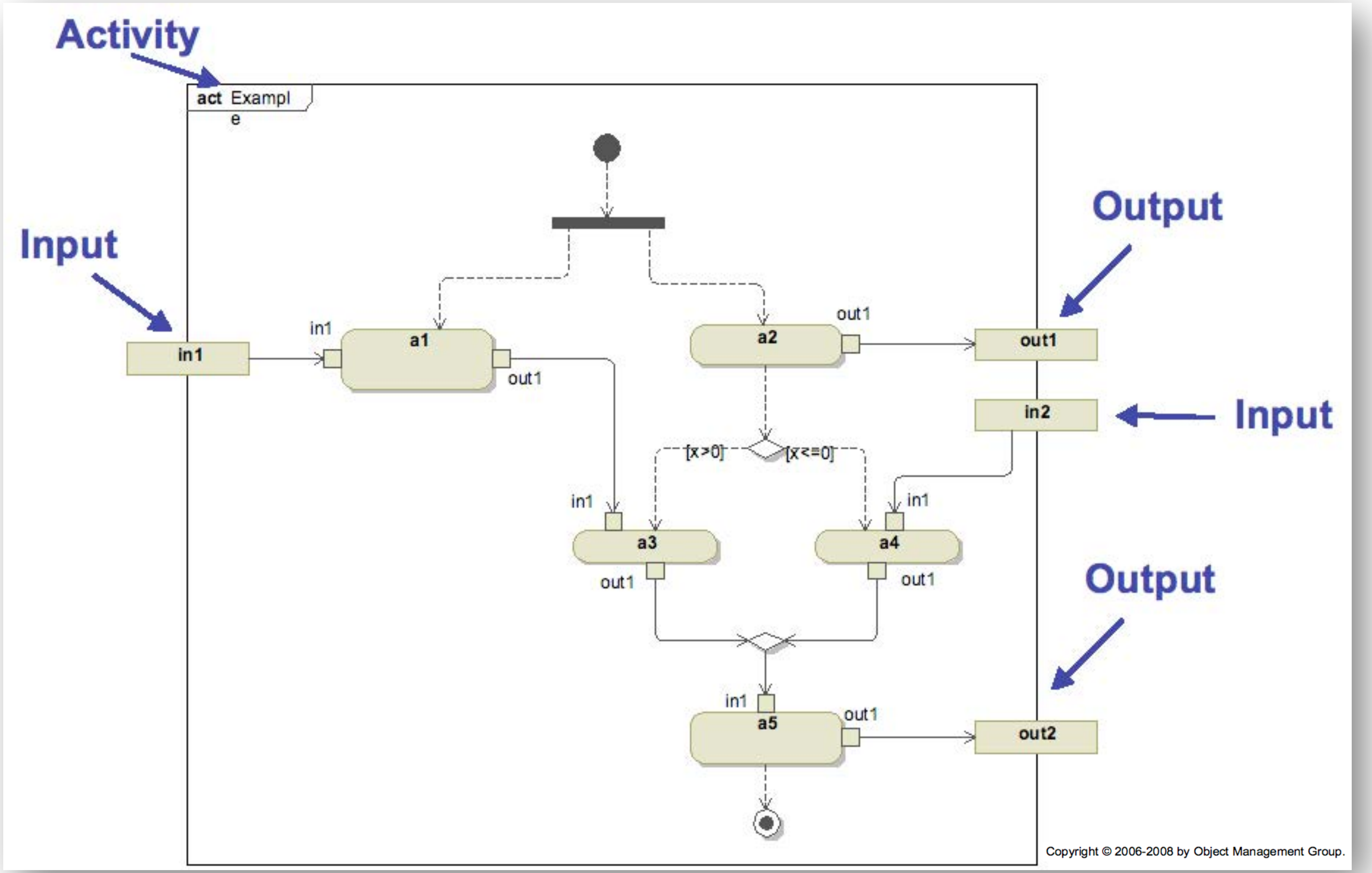
Decision Node – Waits for an input (control or object) token on its input flow and places it on one outgoing flow based on guards



Merge Node – Waits for an input (control or object) token on any input flows and then places it on the outgoing flow

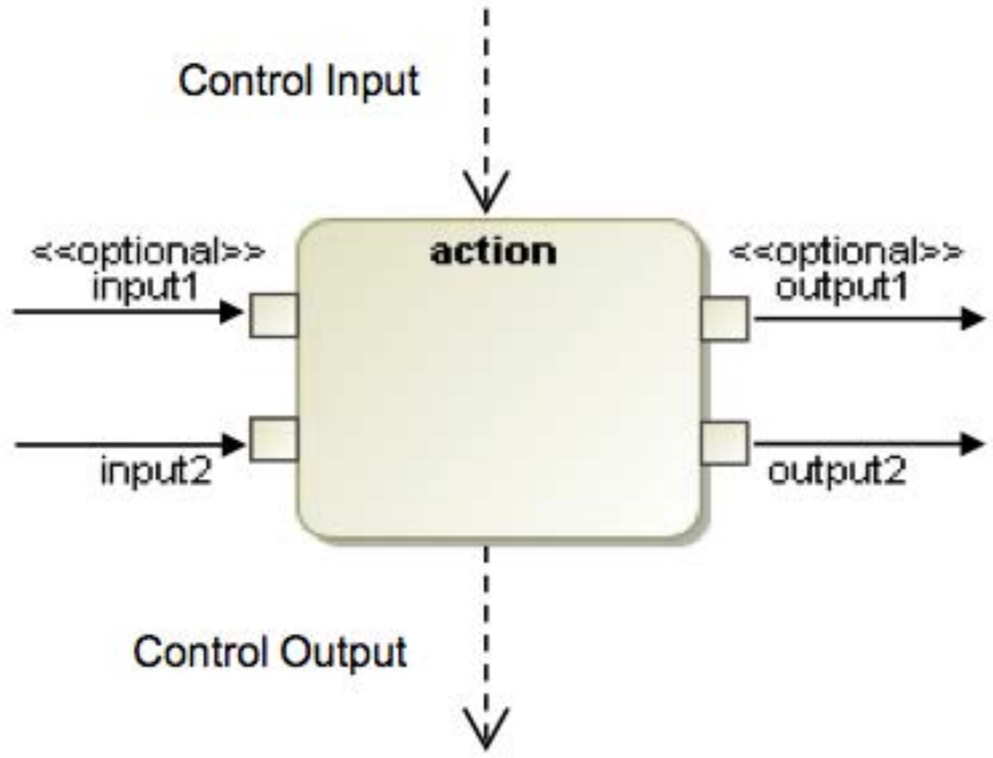
Guard expressions can be applied on all flows

Activity Diagram (act)

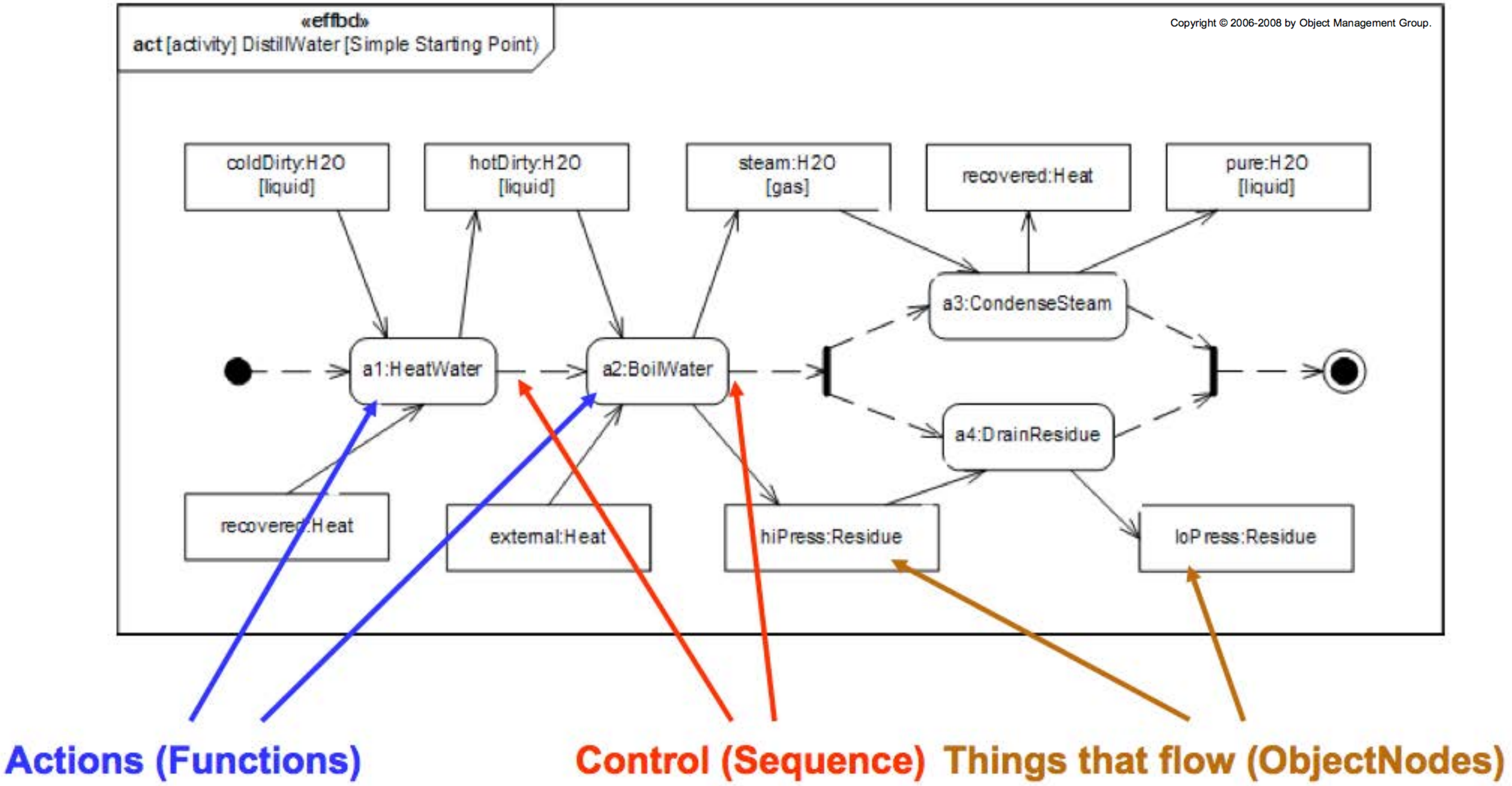


Actions Process Flow of Control and Data

- Two types of flow:
  - Object/Data and Control
- Unit of flow is called a «token» (consumed & produced by actions)



OMG Distiller Example (act)

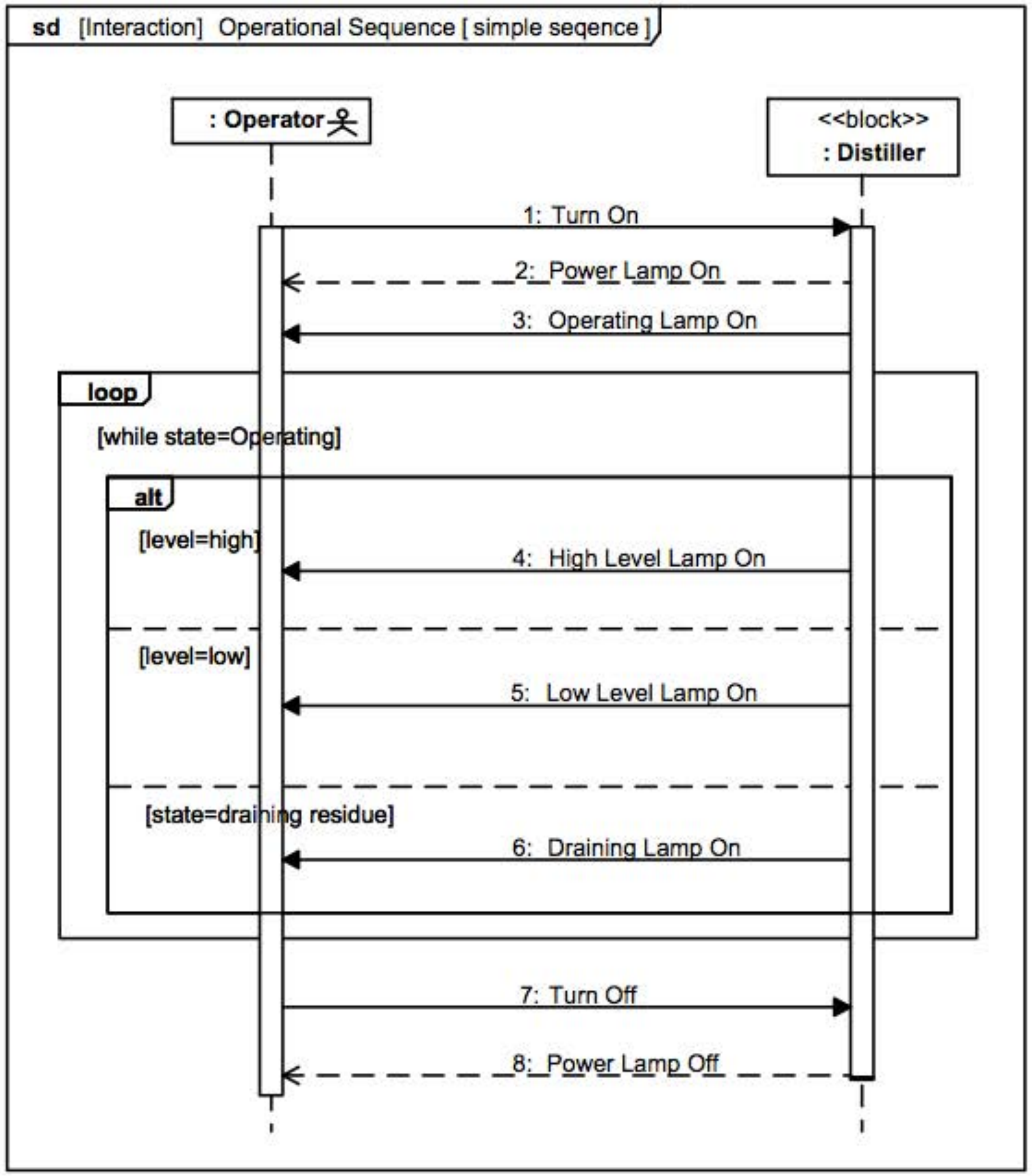


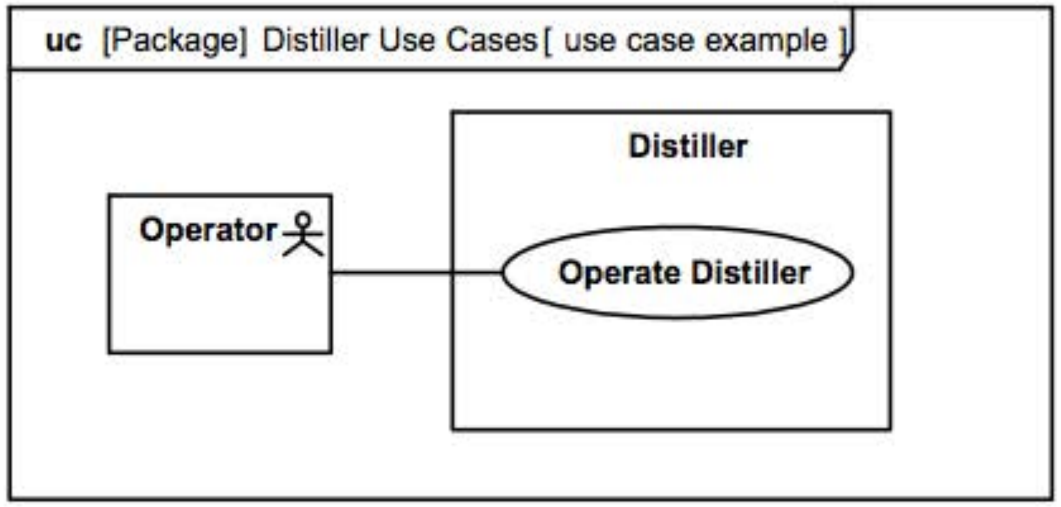
Interaction Diagrams (sdm, sd & uc)

State Machine, Sequence, and Use Case Diagrams:

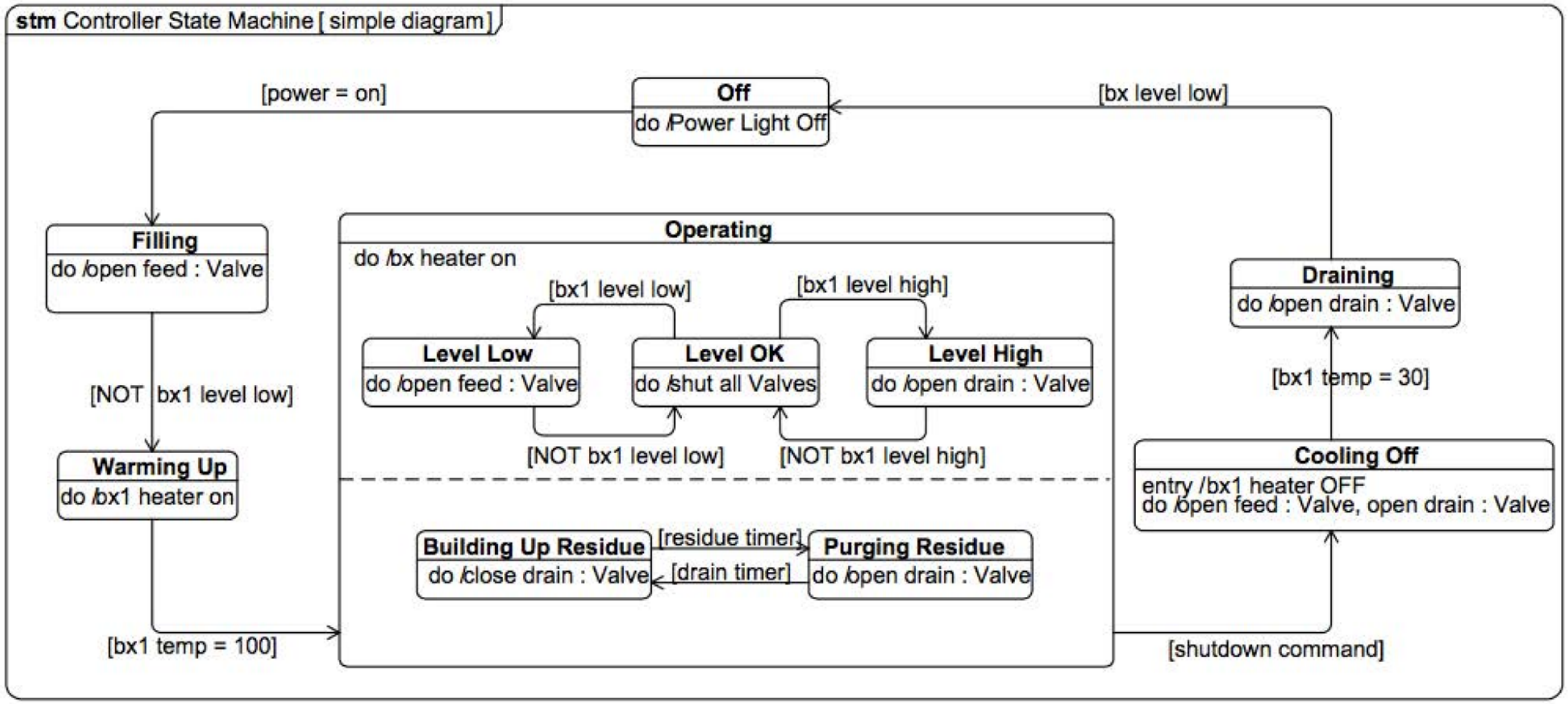
Like in UML!

OMG Distiller Example (sd & uc)





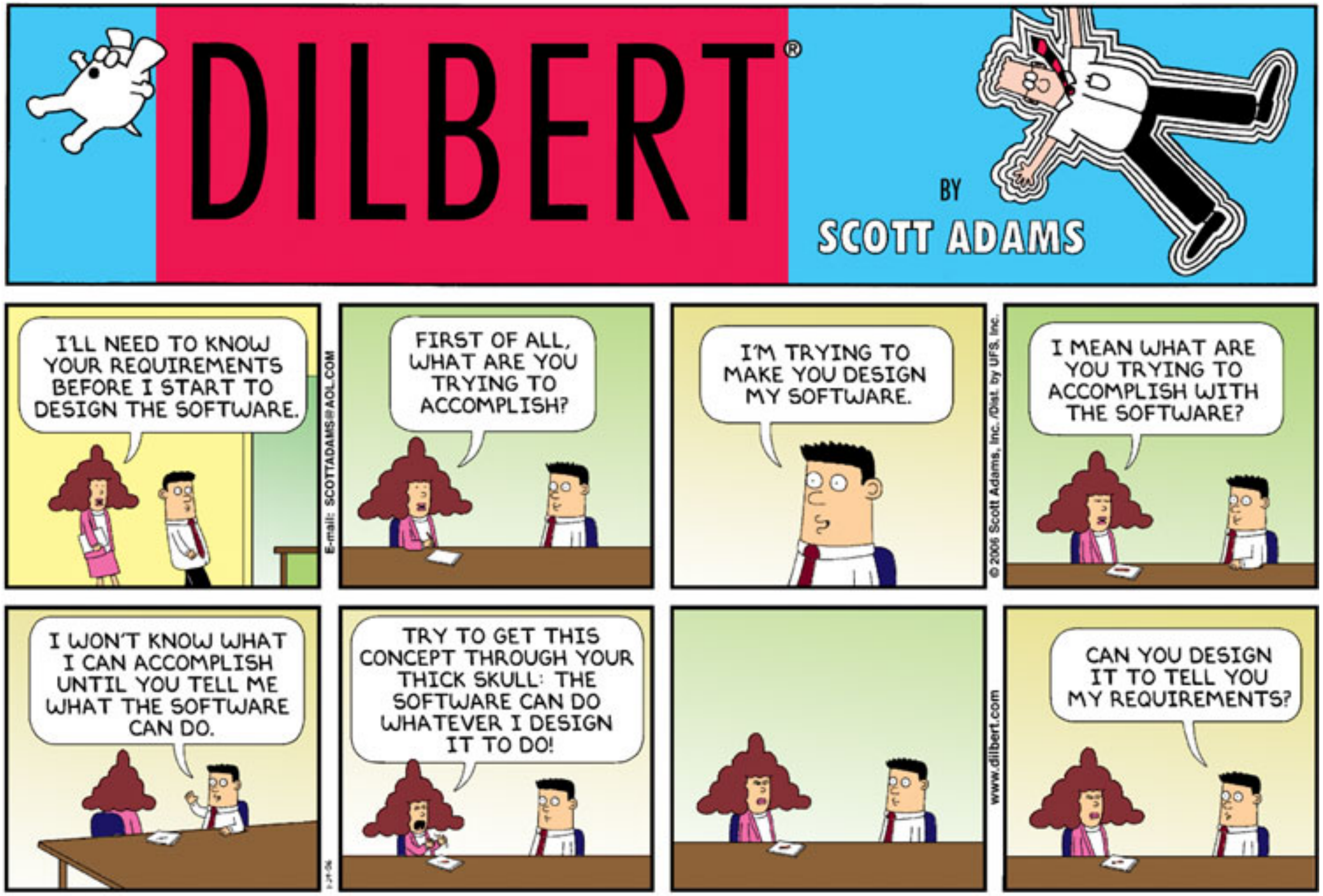
OMG Distiller Example (sdm)



Outline

SysML Requirement Diagram

(SysML extension)


DILBERT®
BY SCOTT ADAMS
I'LL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.
E-mail: SCOTTADAMS@AOL.COM
FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?
I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.
© 2006 Scott Adams, Inc./Dist. by UFS, Inc.
I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?
I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.
TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!
www.dilbert.com
CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?

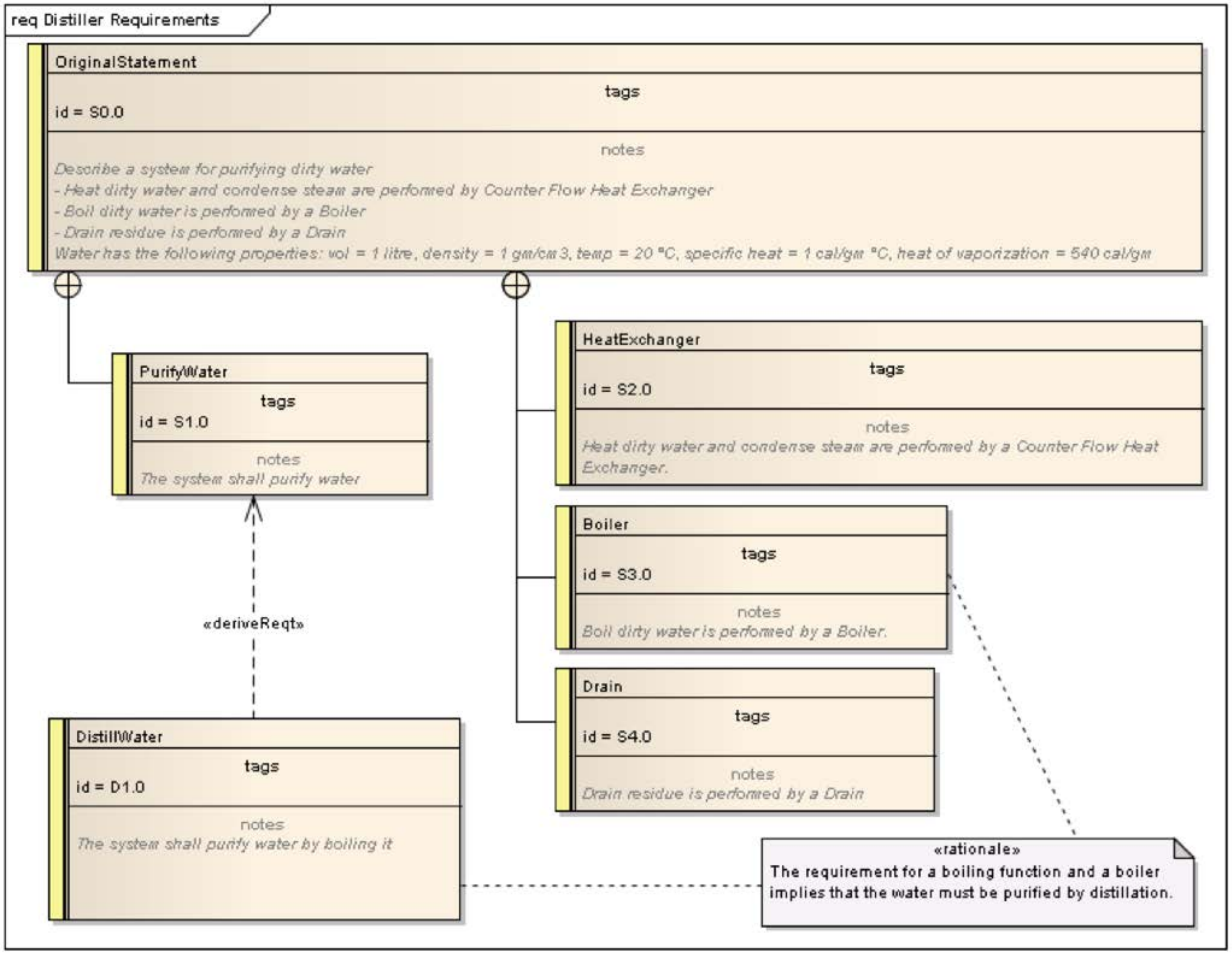
© Scott Adams, Inc./Dist. by UFS, Inc.

SysML Requirement Diagram

(SysML extension)

- <<*requirement*>> allows to represent a text based requirement and their dependencies
  - Includes one identifier id and some textual properties
  - Can add user defined properties
  - Can add user defined requirement categories
- Requirements can be
  - decomposed
  - specialized
- Requirement relationships
  - *« deriveRqt », « refine », « satisfy », « verify », « trace », « copy »*
- <<*Problem*>> *and* <<Rationale>>:
  - can be attached to any model Element to capture *Issues* and *Decisions*

OMG Distiller Example (req)


req Distiller Requirements
OriginalStatement
tags
id = S0.0
notes
Describe a system for purifying dirty water
- Heat dirty water and condense steam are performed by Counter Flow Heat Exchanger
- Boil dirty water is performed by a Boiler
- Drain residue is performed by a Drain
Water has the following properties: vol = 1 litre, density = 1 gm/cm3, temp = 20 °C, specific heat = 1 cal/gm °C, heat of vaporization = 540 cal/gm
PurifyWater
tags
id = S1.0
notes
The system shall purify water
HeatExchanger
tags
id = S2.0
notes
Heat dirty water and condense steam are performed by a Counter Flow Heat Exchanger.
Boiler
tags
id = S3.0
notes
Boil dirty water is performed by a Boiler.
Drain
tags
id = S4.0
notes
Drain residue is performed by a Drain
«deriveReqt»
DistillWater
tags
id = D1.0
notes
The system shall purify water by boiling it
«rationale»
The requirement for a boiling function and a boiler implies that the water must be purified by distillation.

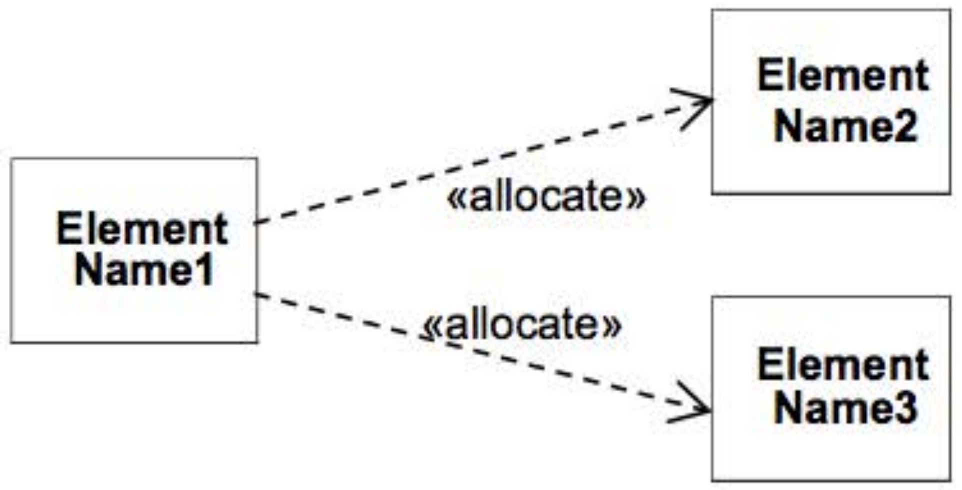
(from [Finance10])

Allocations

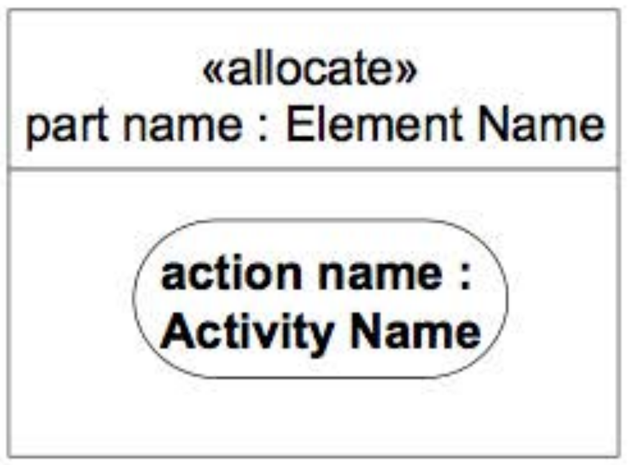
(SysML extension)

- Term from the systems engineers' vocabulary
- General relationship between two elements of the model
- Different kinds of allocation:
  - Functionality - component
  - Logical component – physical component
  - Software – hardware
- Explicit allocation of activities to structure via swim lanes (i.e., activity partitions)
- Usable under graphical or tabular representation
- Enables consistency in the model (e.g., between dynamic model elements and static model elements).

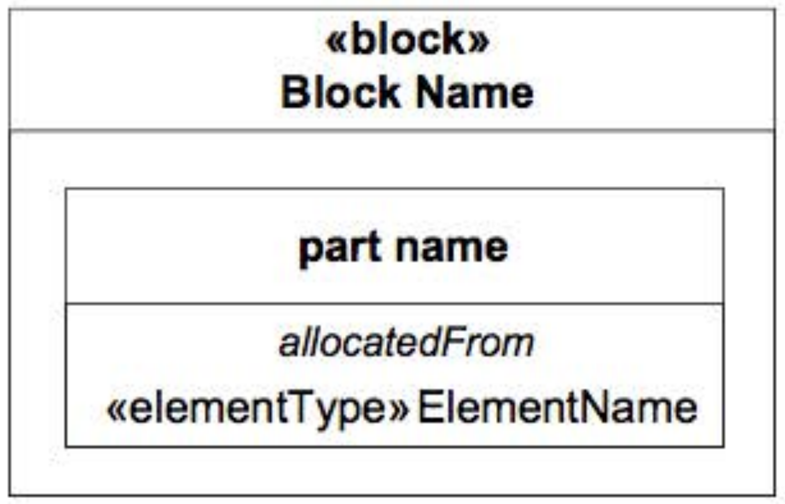
Allocations Representation



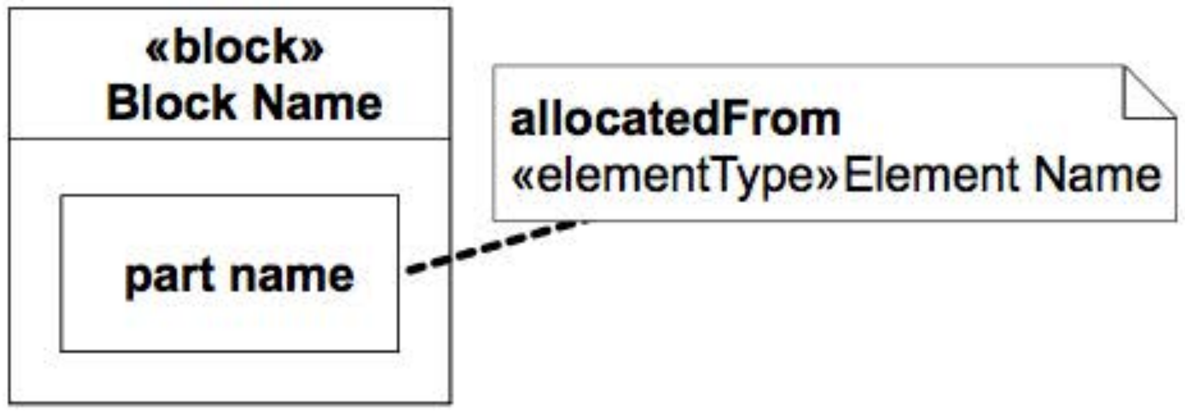
Allocate Relationship



Explicit Allocation of Action to Part Property



Compartment Notation



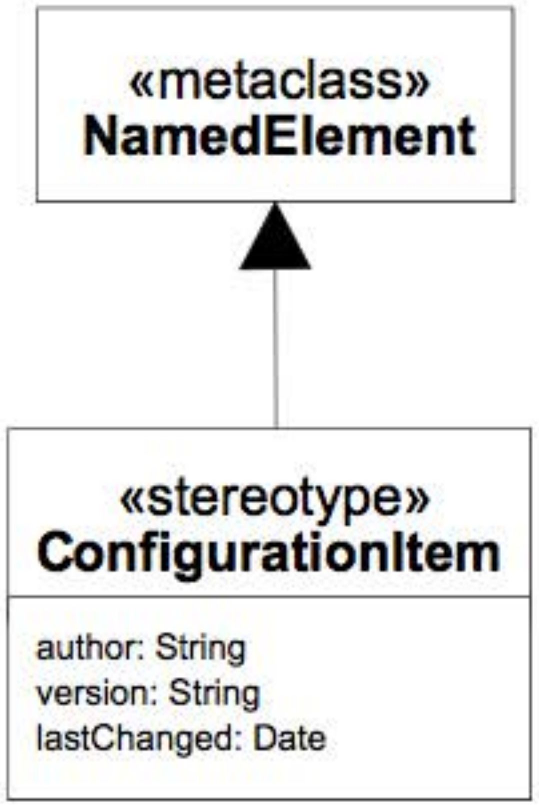
Callout Notation

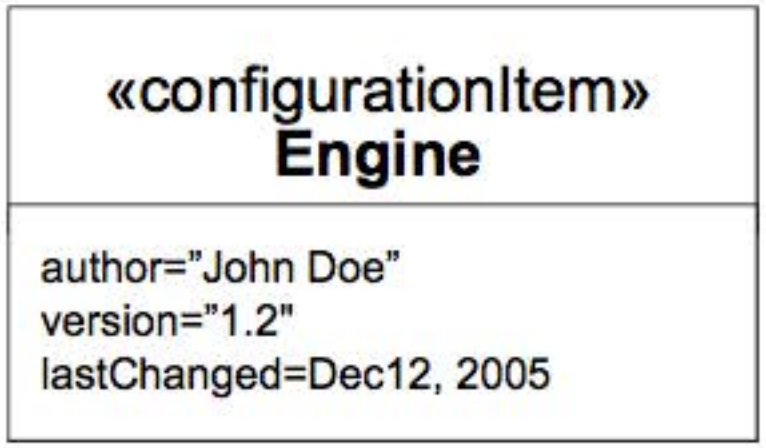
Read as follows: "part name has constraints that are allocated to/from an <<element type>> Element Name"

Stereotypes & Model Libraries

- Mechanisms for further customizing SysML Profiles represent extensions to the language
  - Stereotypes extend meta-classes with properties and constraints
    - Stereotype properties capture metadata about the model element
  - Profile is applied to user model
  - Profile can also restrict the subset of the meta-model used when the profile is applied
- Model Libraries represent reusable libraries of model elements

Stereotypes

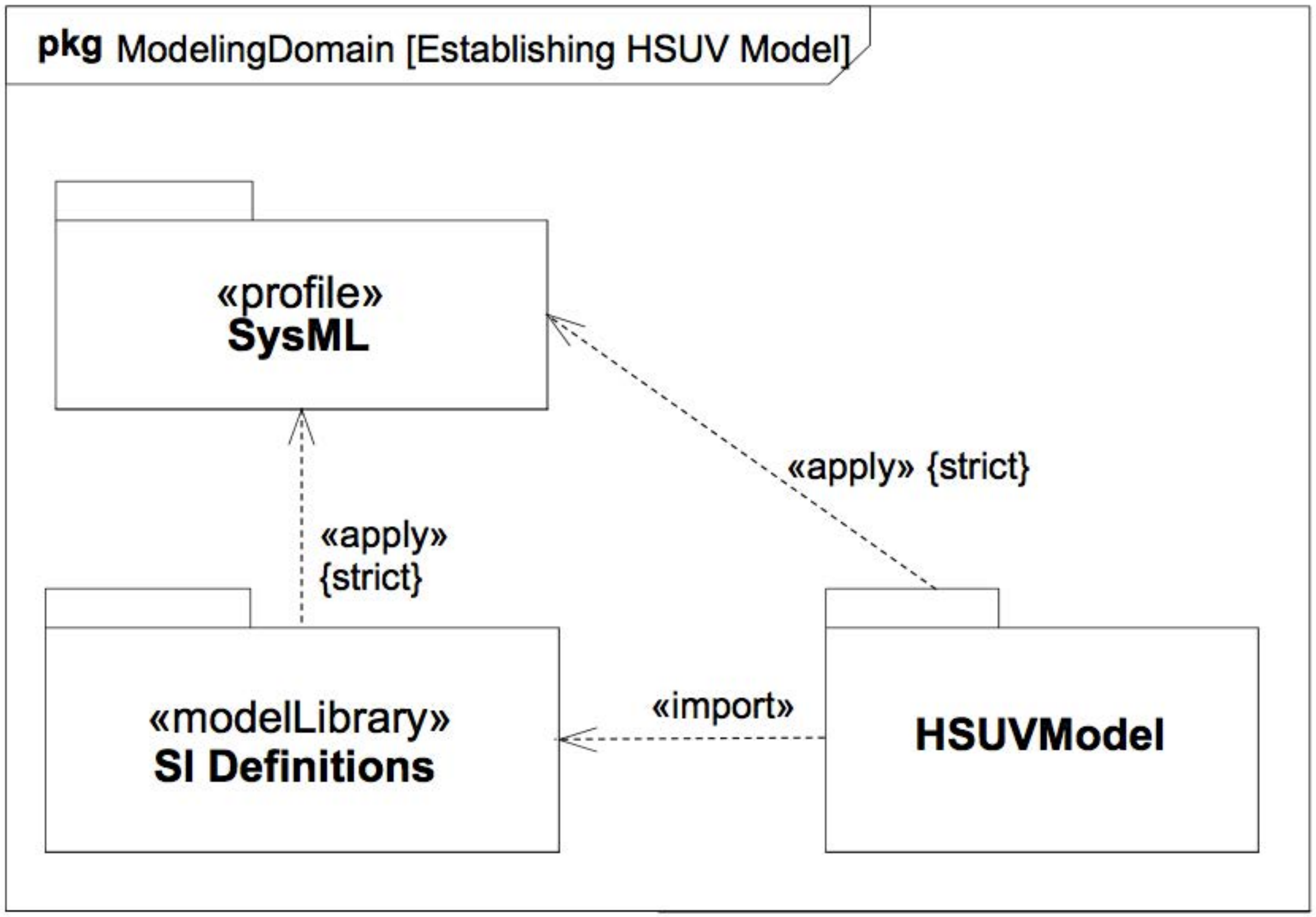




Defining the Stereotype

Applying the Stereotype

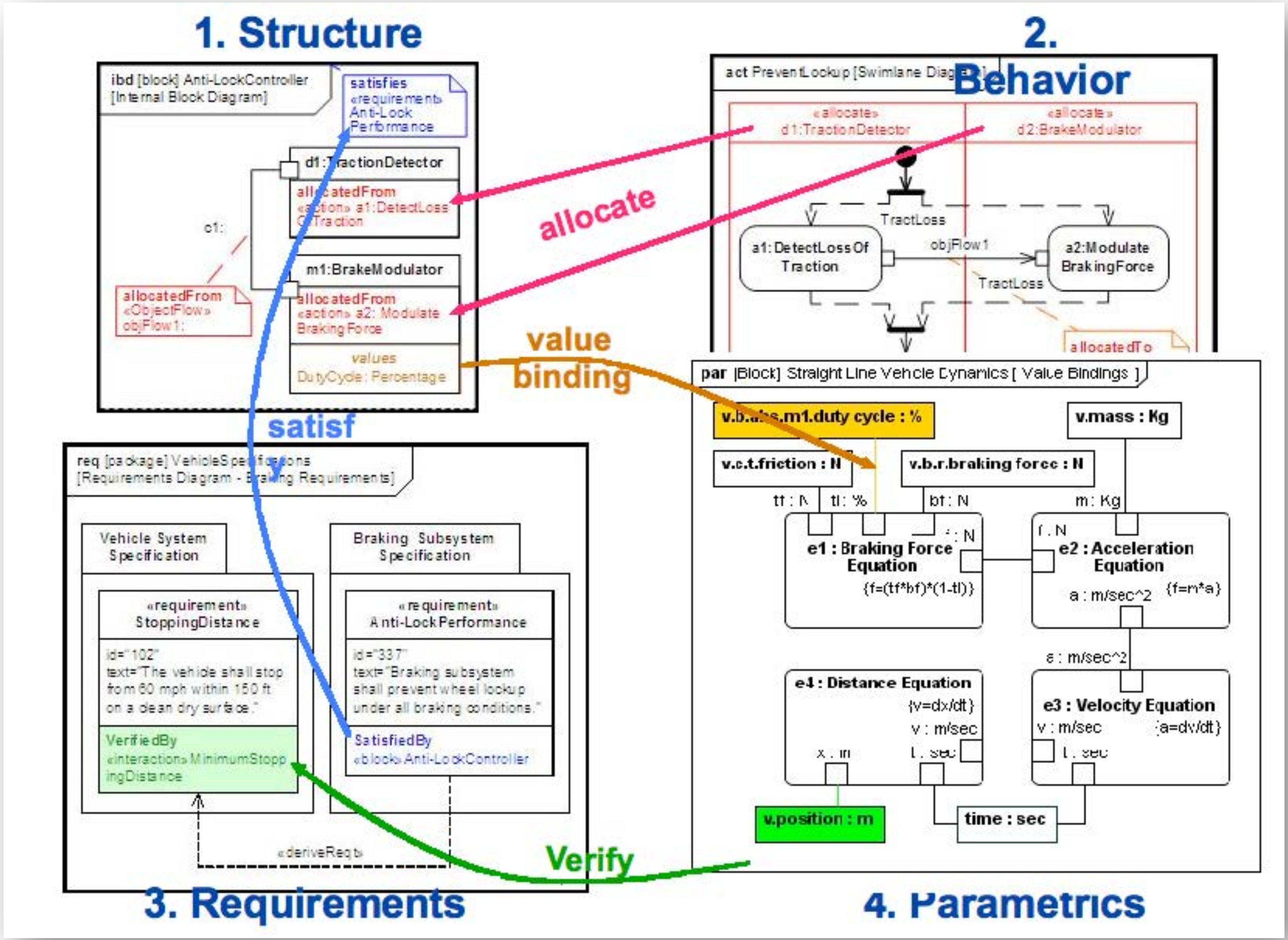
Model Libraries


pkg ModelingDomain [Establishing HSUV Model]
«profile» SysML
«apply» {strict}
«apply» {strict}
«modelLibrary» SI Definitions
«import»
HSUVModel

SM@RT

Outline

Cross Connecting Model Elements

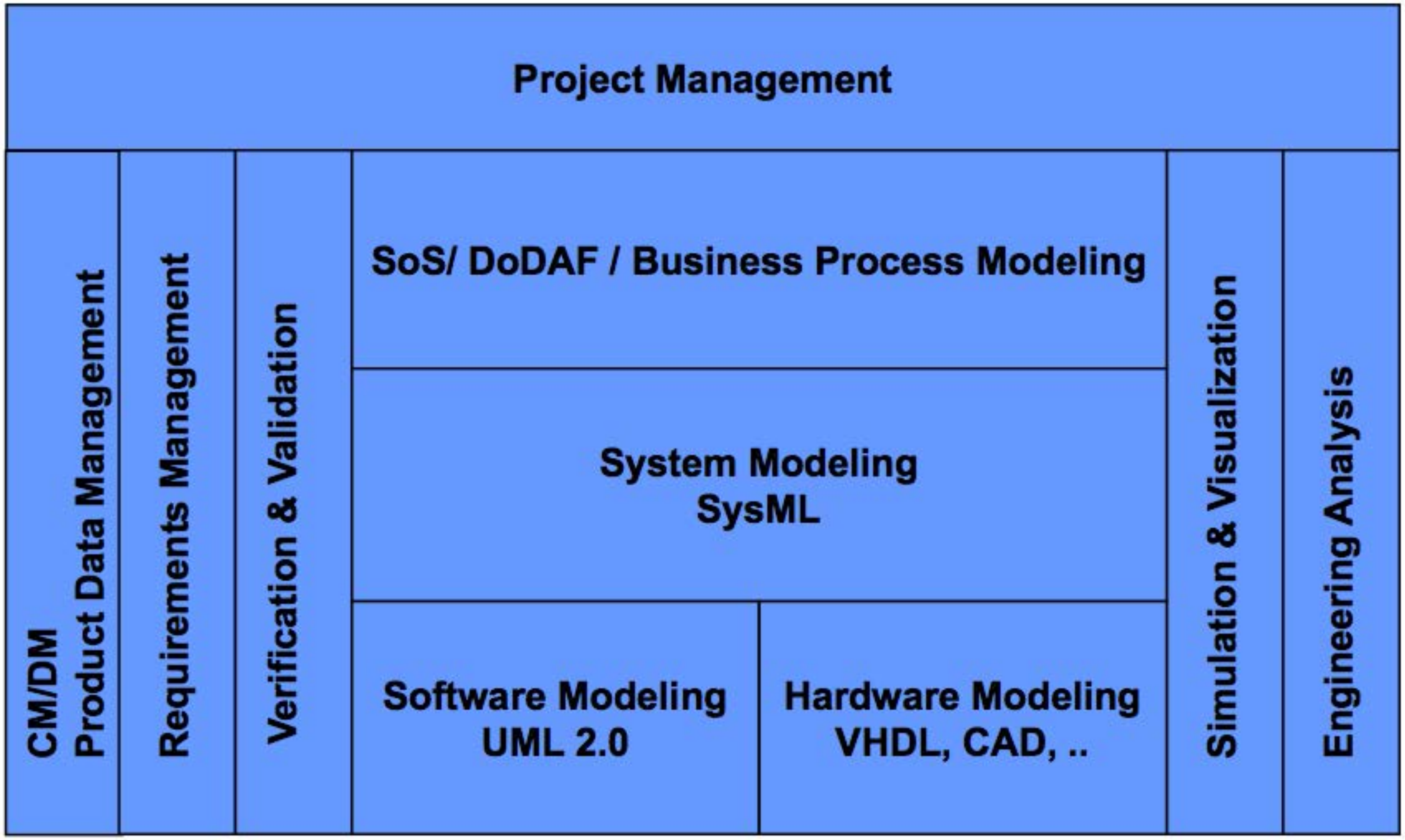


1. Structure
ibd [block] Anti-LockController [Internal Block Diagram]
satisfies «requirement» Anti-Lock Performance
d1:TractionDetector
allocatedFrom «action» a1:DetectLossOfTraction
c1:
m1:BrakeModulator
allocatedFrom «ObjectFlow» objFlow1;
allocatedFrom «action» a2: Modulate Braking Force
values DutyCycle: Percentage
2. Behavior
act PreventLockup [Swimlane Diagram]
«allocate» d1:TractionDetector
«allocate» d2:BrakeModulator
TractLoss
a1:DetectLossOf Traction
objFlow1
a2:Modulate BrakingForce
allocatedTo
allocate
value binding
satisfy
3. Requirements
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» MinimumStoppingDistance
«requirement» Anti-LockPerformance
id="337" text="Braking subsystem shall prevent wheel lockup under all braking conditions."
SatisfiedBy «block» Anti-LockController
«deriveReqt»
Verify
4. Parametrics
par [Block] Straight Line Vehicle Dynamics [Value Bindings]
v.b.abs.m1.duty cycle : %
v.mass : Kg
v.c.t.friction : N
v.b.r.braking force : N
tf : N
tl : %
bf : N
m : Kg
f : N
e1 : Braking Force Equation {f=(tf*bf)*(1-tl)}
e2 : Acceleration Equation {f=m*a}
a : m/sec^2
e4 : Distance Equation {v=dx/dt}
v : m/sec
x : m
t : sec
e3 : Velocity Equation {a=dv/dt}
v.position : m
time : sec

Conclusion

- SysML is:
  - a specific language for complex systems
  - strongly UML-Based
  - focusing on specification, analysis, design and verification
- SysML is not:
  - a method
  - just a UML profile
  - sufficient in itself

Typical Integrated Tool Environment



Tools

- Artisan (Studio)
- EmbeddedPlus (SysML Toolkit)
- No Magic (Magic Draw)
- Sparx Systems (Enterprise Architect)
- IBM (Tau and Rhapsody)

- TopCased: http://topcased.gforge.enseeiht.fr/
- Papyrus: http://www.papyrusuml.org

- Visio SysML template

References and links

- Books:
  - *« A Practical Guide to SysML », S. Friedenthal, A. Moore, R. Steiner, The MK/OMG Press, Elsevier, 2008.*
  - ***« SysML par l'exemple, un langage de modélisation pour systèmes complexes », P. Roques, Éditions Eyrolles, 2009.***
- The Official OMG SysML site:
  - http://www.omgsysml.org
  - http://www.omg.org/spec/SysML/
- INCOSE, International Council on Systems
  - http://www.incose.org/
- AFIS, Association Française d'Ingénierie Système
  - http://www.afis.fr/
- Association SysML France
  - http://sysmlfrance.blogspot.com/
- Misc:
  - Notation overview (4p.): http://www.oose.de/downloads/sysml.overview.oose.pdf