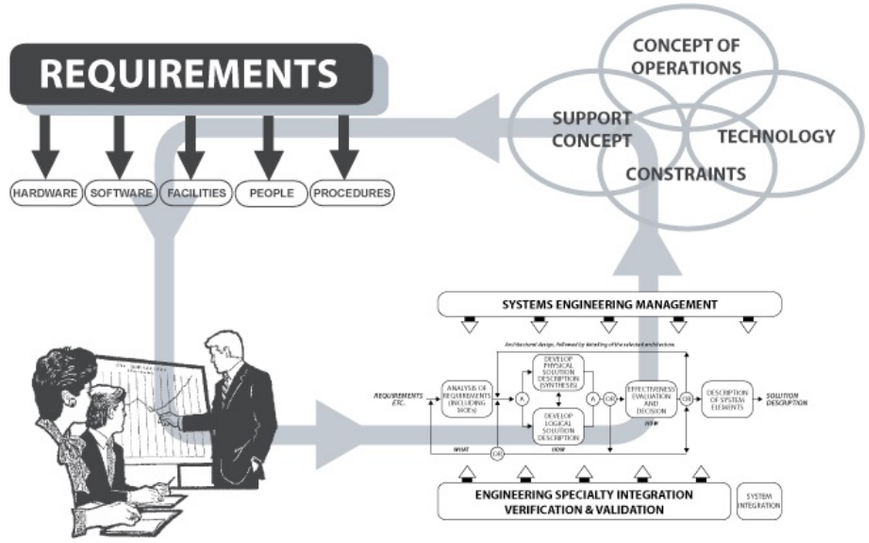


Short Introduction about Systems Engineering and SysML

Benoit Combemale
DiverSE team (IRISA & Inria)
University of Rennes
benoit.combemale@irisa.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., 2023.

Materials available on : http://combemale.fr/

Motivation

- Scale to real-world innovative complex systems
- From a software to a system viewpoint
- With a rigorous approach

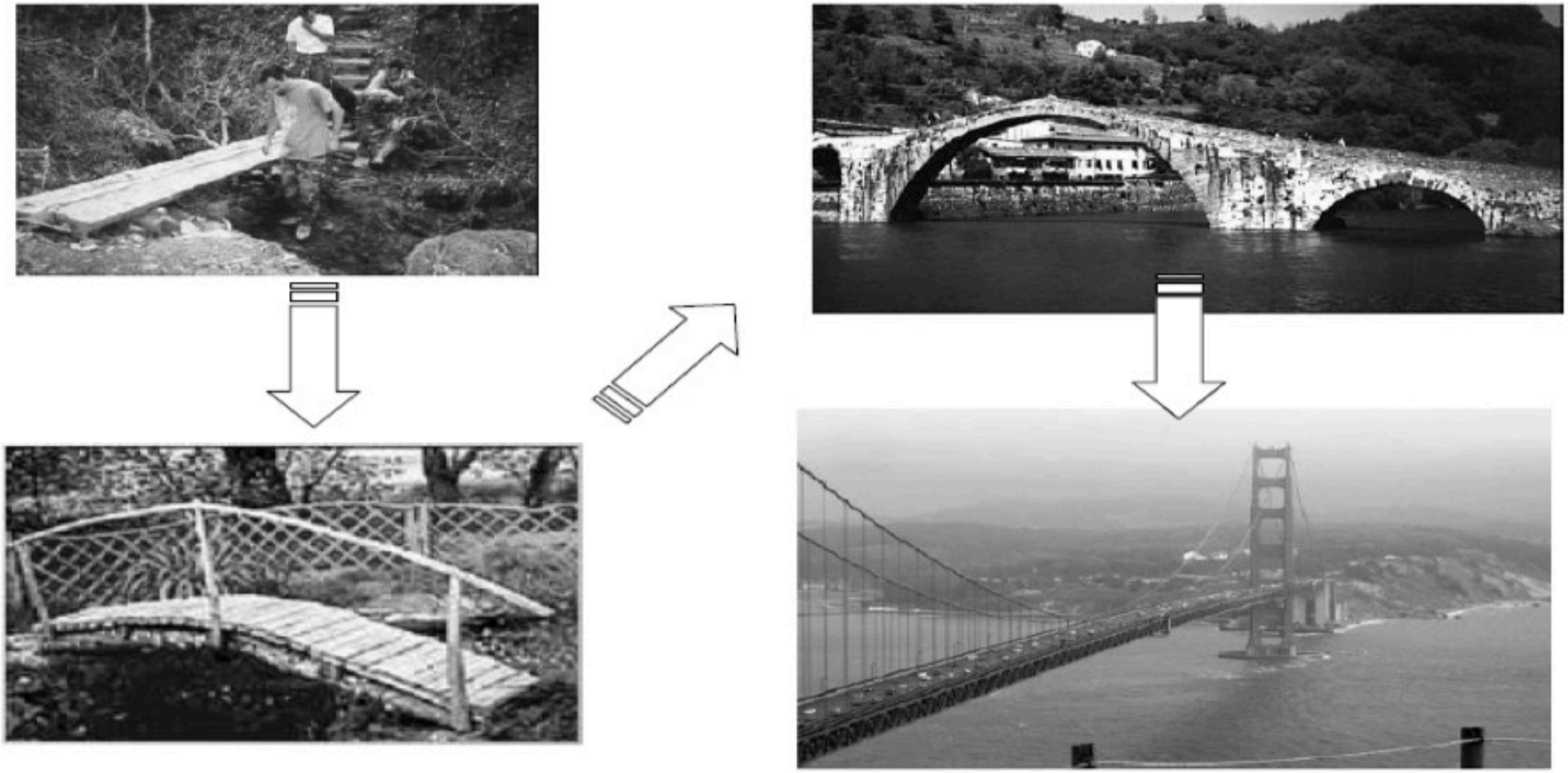
- => From craft to engineering of software-intensive systems

Objective

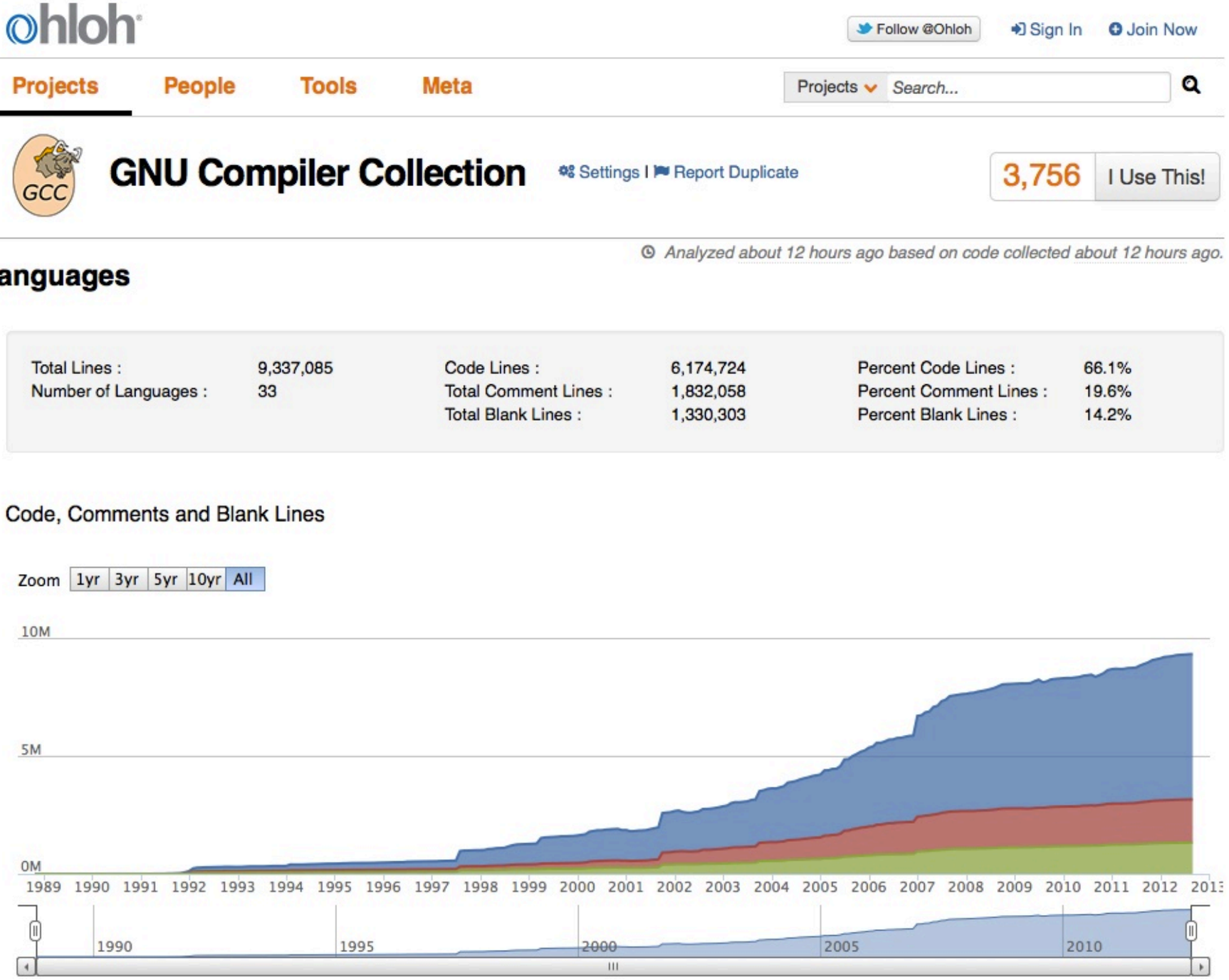
- Technical skills
  - Introduction to Systems Engineering
  - Organization of the project management
  - Get the spirit, the overall method, and the vocabulary

- Soft skills
  - Collaborative work
  - Tradeoff analysis
  - Report and argue a design

System Complexity



System Complexity



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

Linux Kernel

- Variability
- Reusability
- Durability

≈10 000 variants

≈10^{80} is the estimated number of atoms in the universe

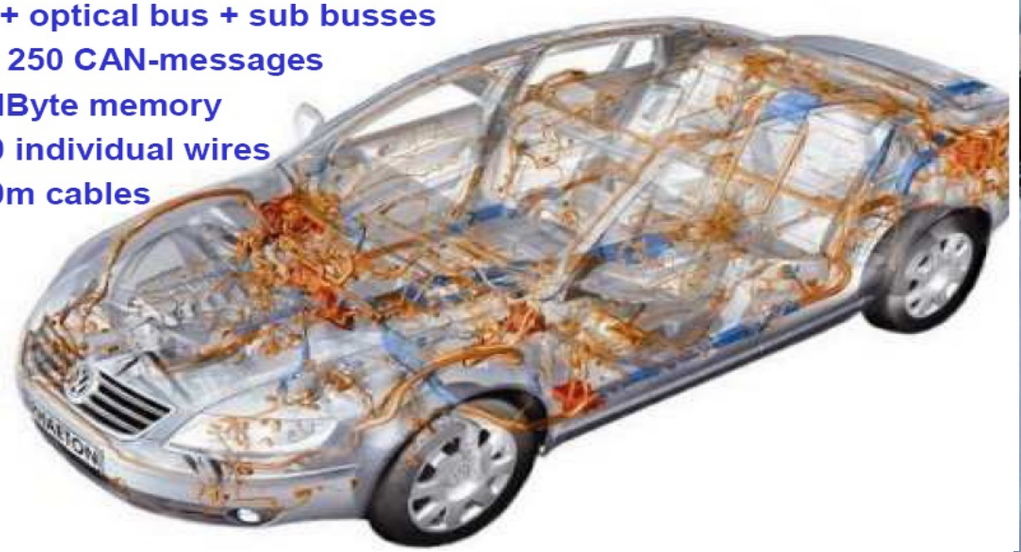
≈10^{40} is the estimated number of possible chess positions

System Complexity

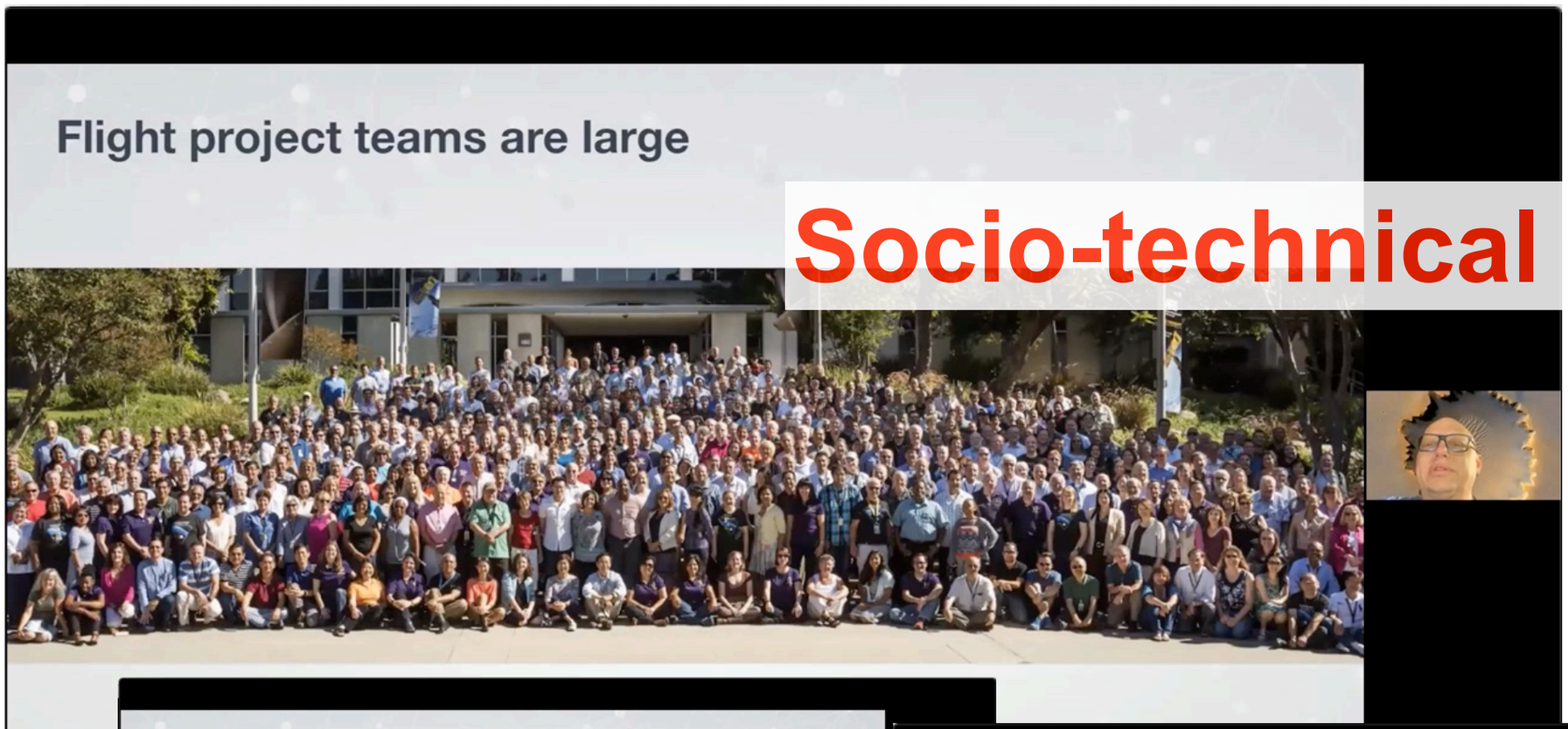
- Embedded
- Critical
- Real time

Phaeton

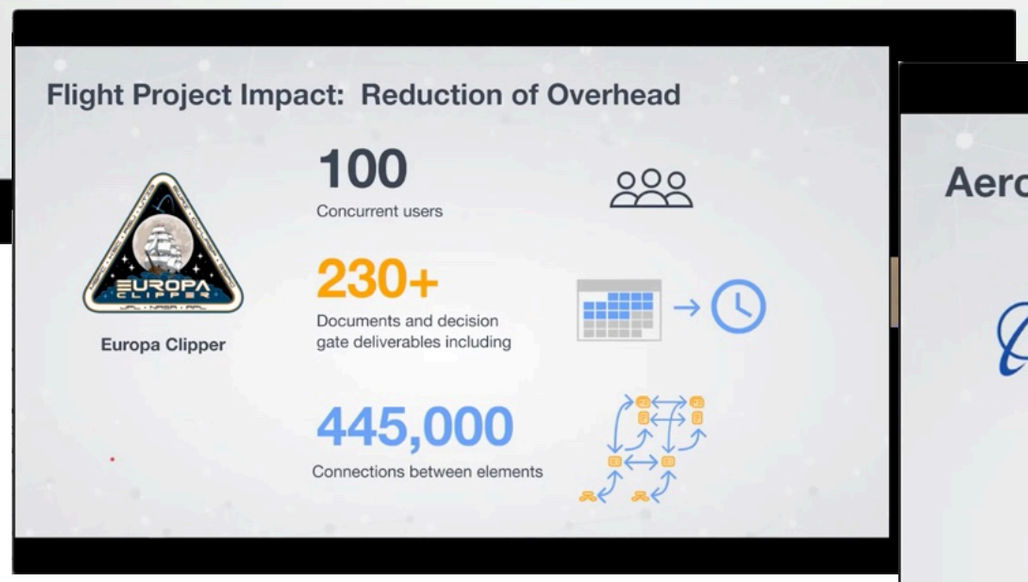
- 61 networked ECUs
- 3 bus systems + optical bus + sub busses
- 2500 signals in 250 CAN-messages
- more than 50 MByte memory
- more than 2000 individual wires
- more than 3800m cables



System Complexity



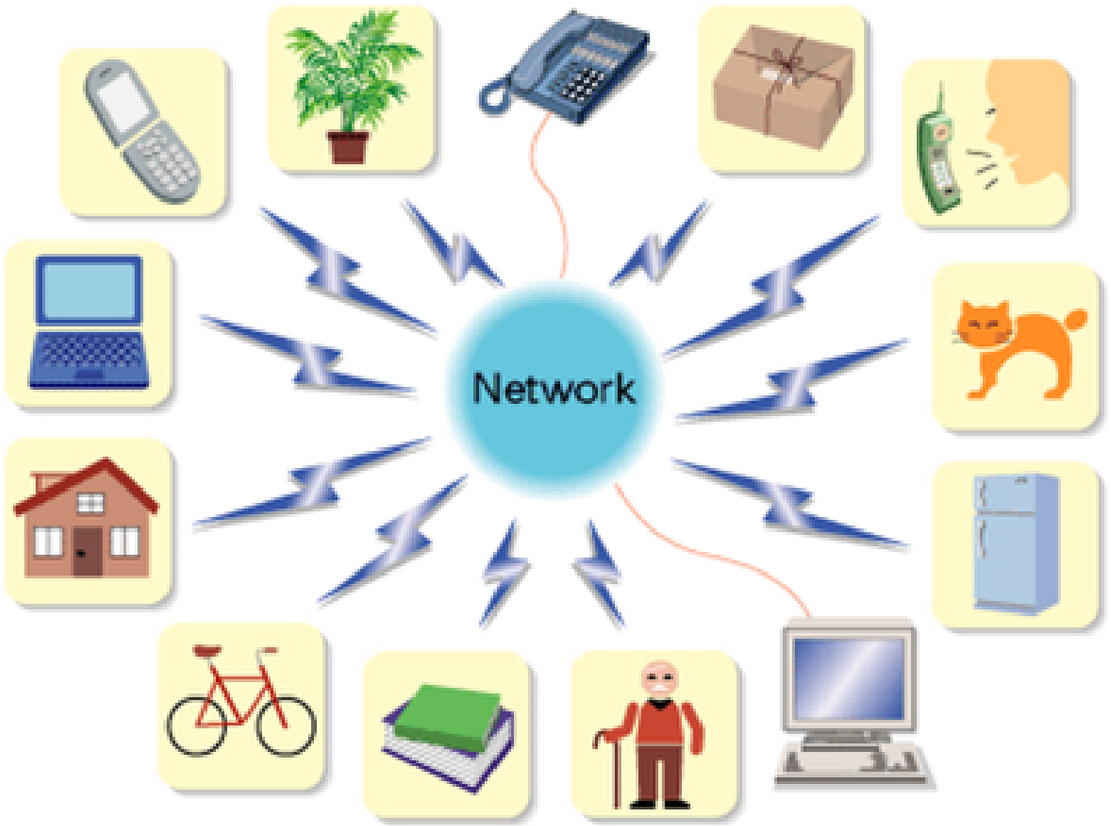
Socio-technical coordination

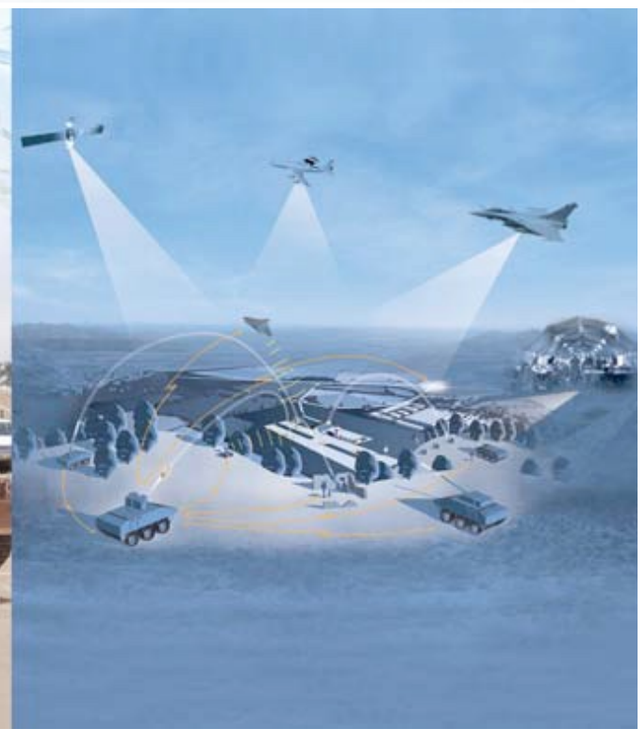


Prof. Benoit Combemale (Univ. Rennes & Inria), 2023. ***http://***

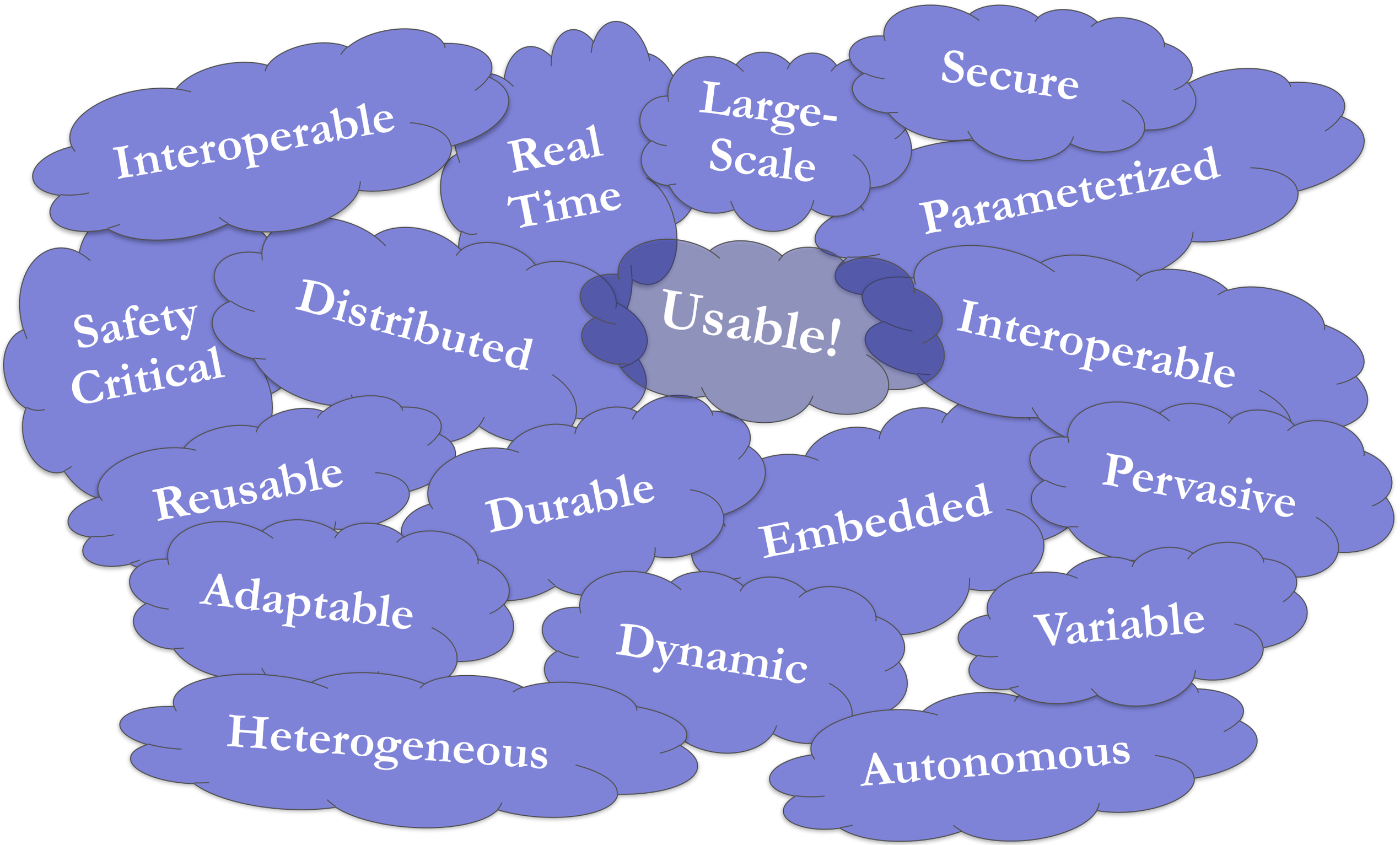
System Complexity

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





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 realized through software and hardware solutions.

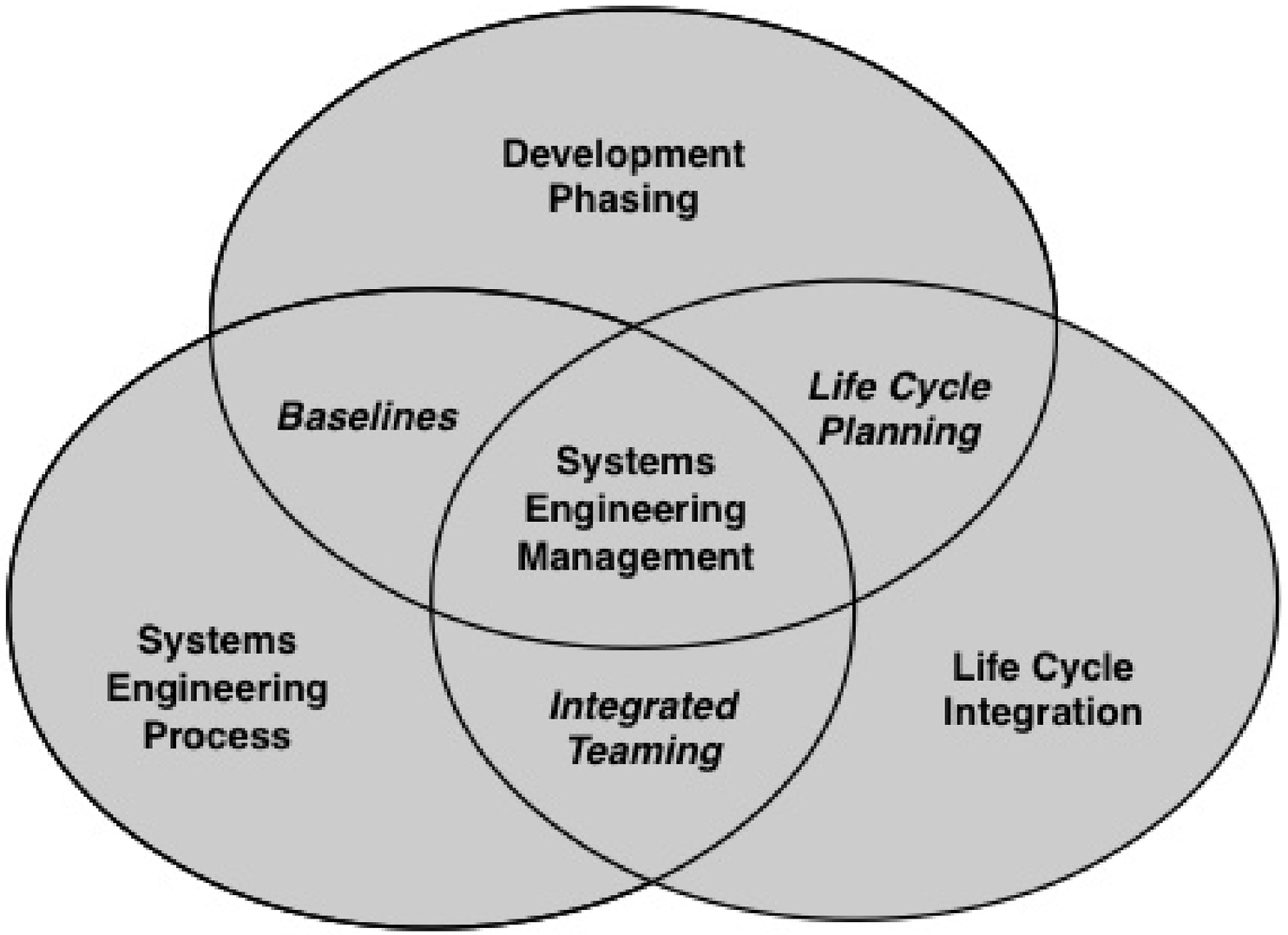
- ... relies on modelling and simulation methods to validate requirements or to evaluate the system.

- ... 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.)*, factories, 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) ...

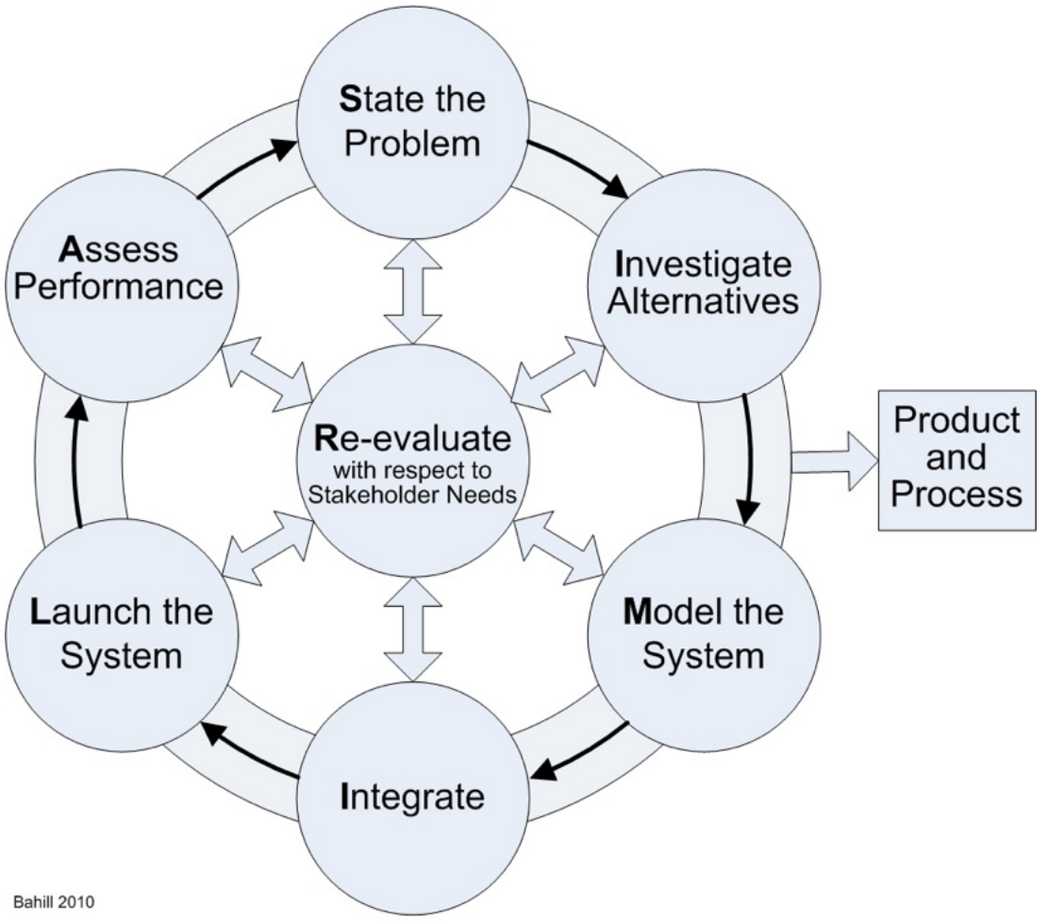


Source: *Systems Engineering Fundamentals*. Defense Acquisition University Press, 2001 (cf. http://www.dau.mil/pubscats/PubsCats/SEFGuide%2001-01.pdf)

- 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/

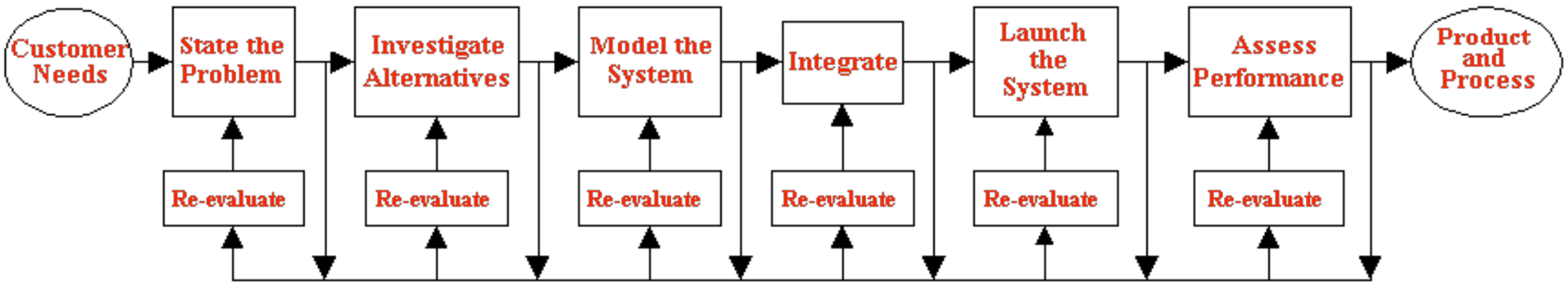
The SIMILAR Process (INCOSE)

- **S**tate the problem
- **I**nvestigate alternatives
- **M**odel the system
- **I**ntegrate
- **L**aunch the system
- **A**ssess performance
- **R**e-evaluate



The SIMILAR Process

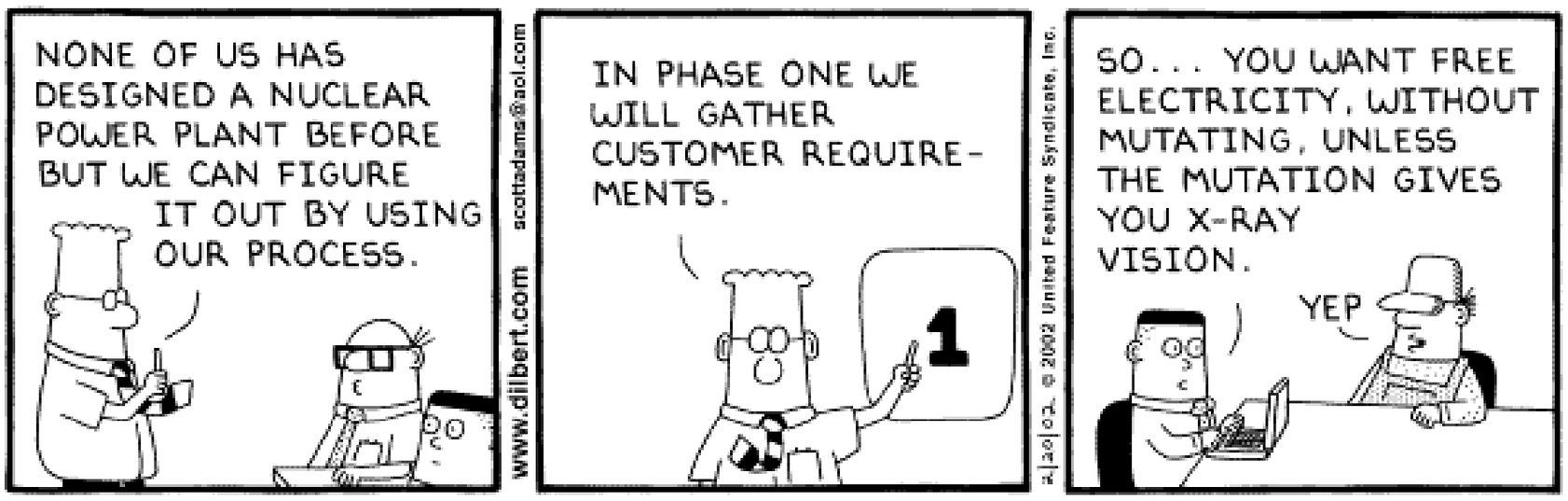
The Systems Engineering Process



The SIMILAR Process (INCOSE)

State the Problem

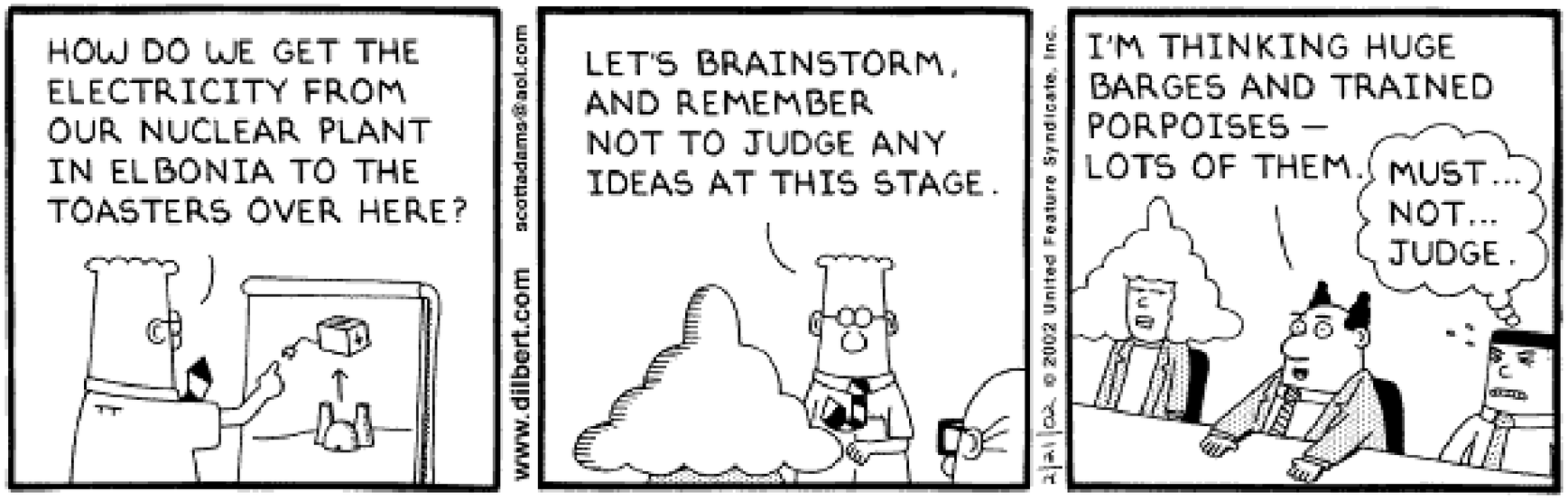
Copyright @ 2002 United Feature Syndicate, Inc.



Copyright @ 2002 United Feature Syndicate, Inc.

The SIMILAR Process (INCOSE)

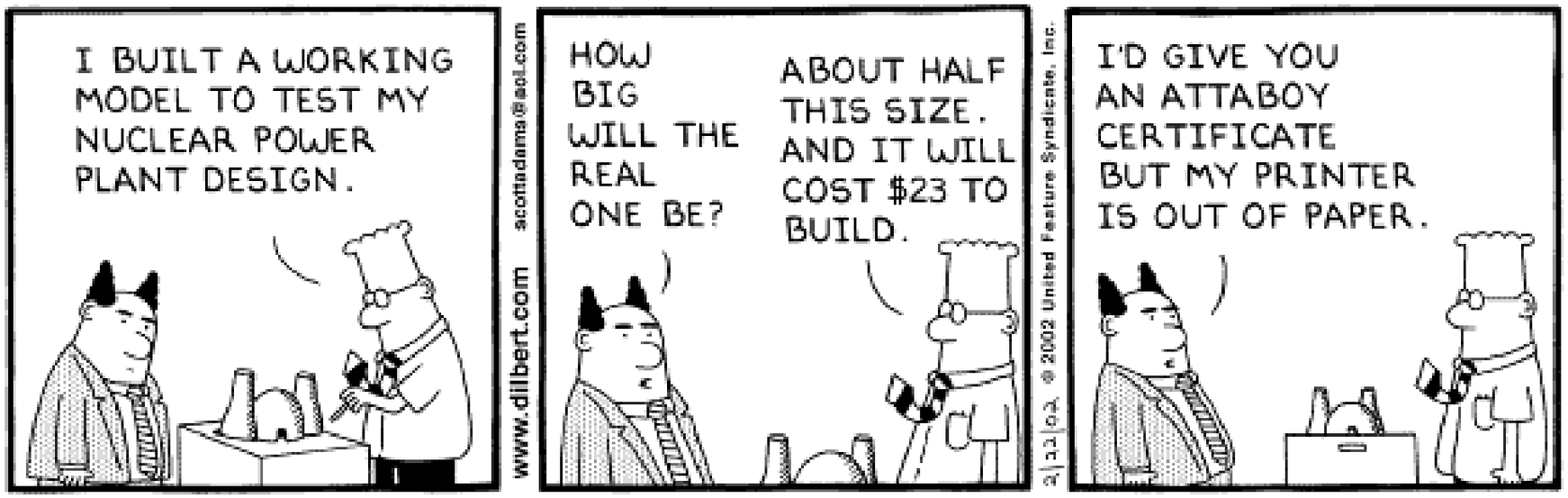
Investigate Alternatives



Copyright @ 2002 United Feature Syndicate, Inc.

The SIMILAR Process (INCOSE)

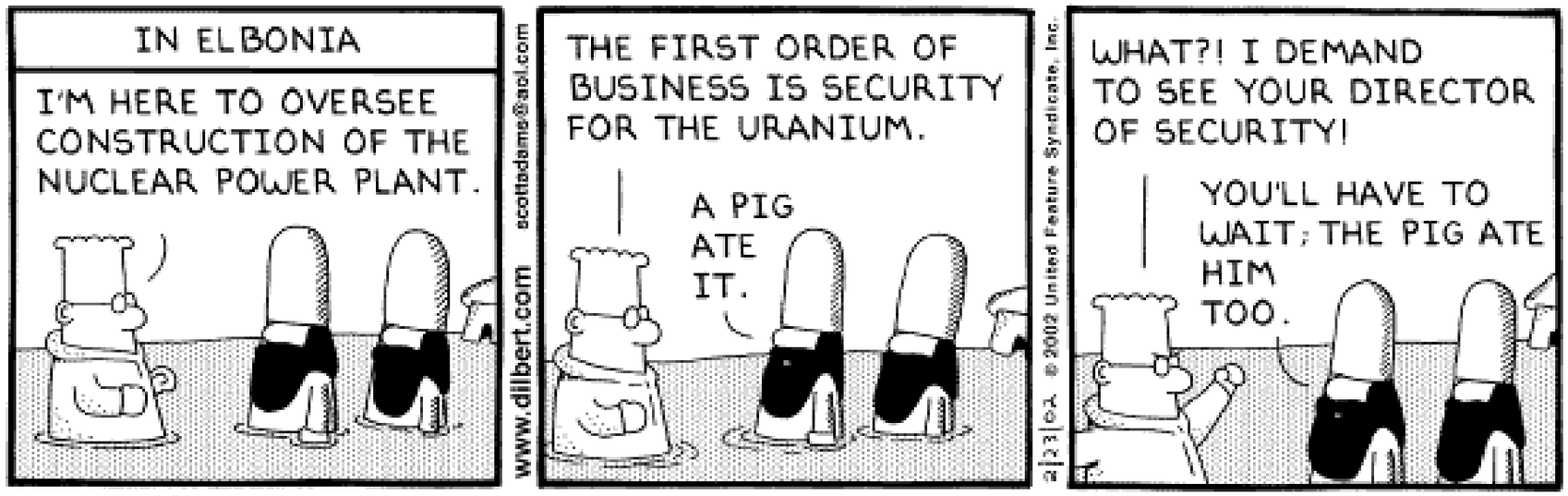
Model the System



Copyright © 2002 United Feature Syndicate, Inc.

The SIMILAR Process (INCOSE)

Asses Performance



Copyright © 2002 United Feature Syndicate, Inc.

Outline

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

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*
- 05/2017 : SysML v1.5 (current version)
- *12/2017 : UML v2.5.1 (current version)*
- 03/2019 : SysML v1.6 beta
- *Towards SysML v2!*

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

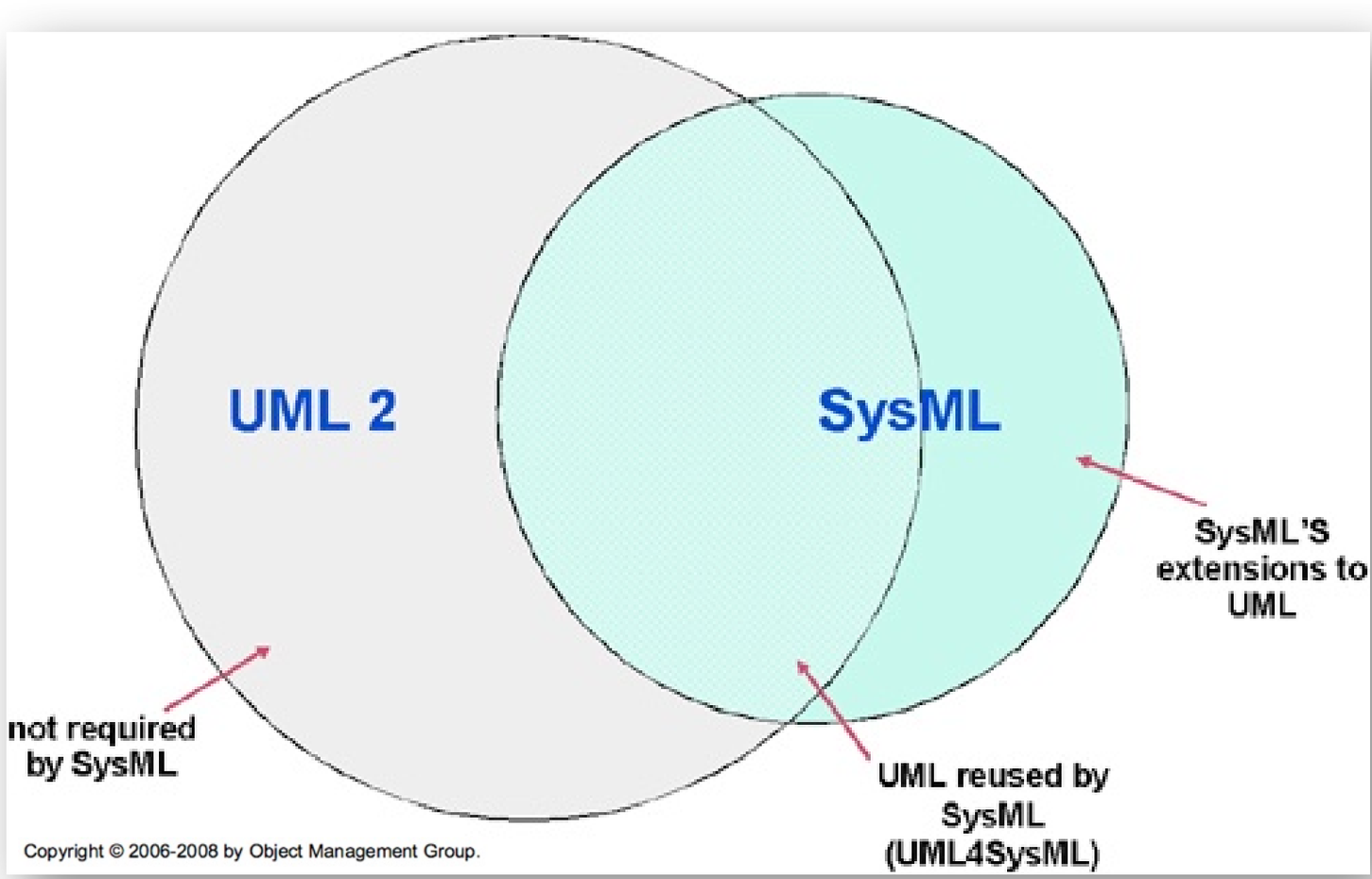
SysML Overview

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

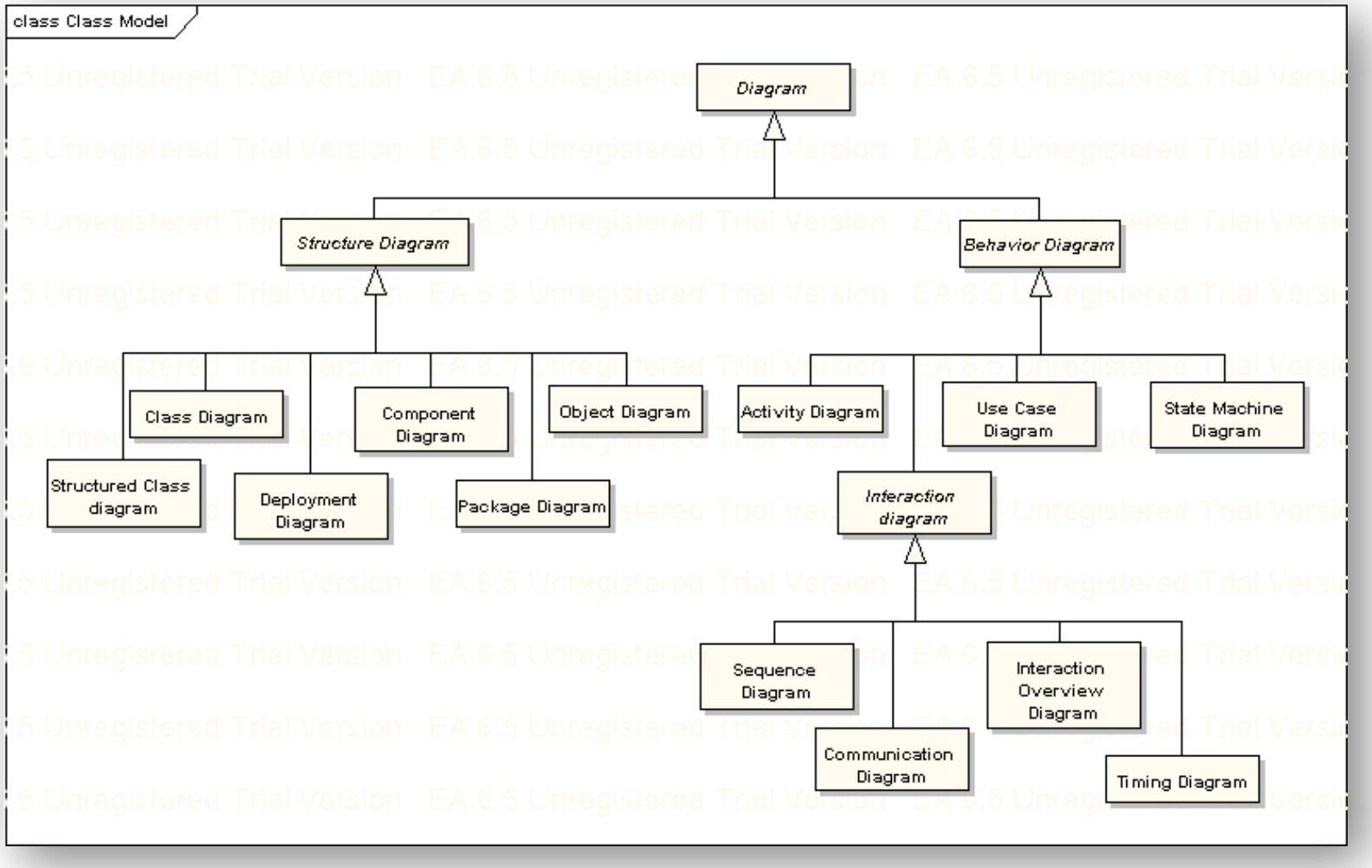
* not anymore the case in the future SysML v2

SysML Overview

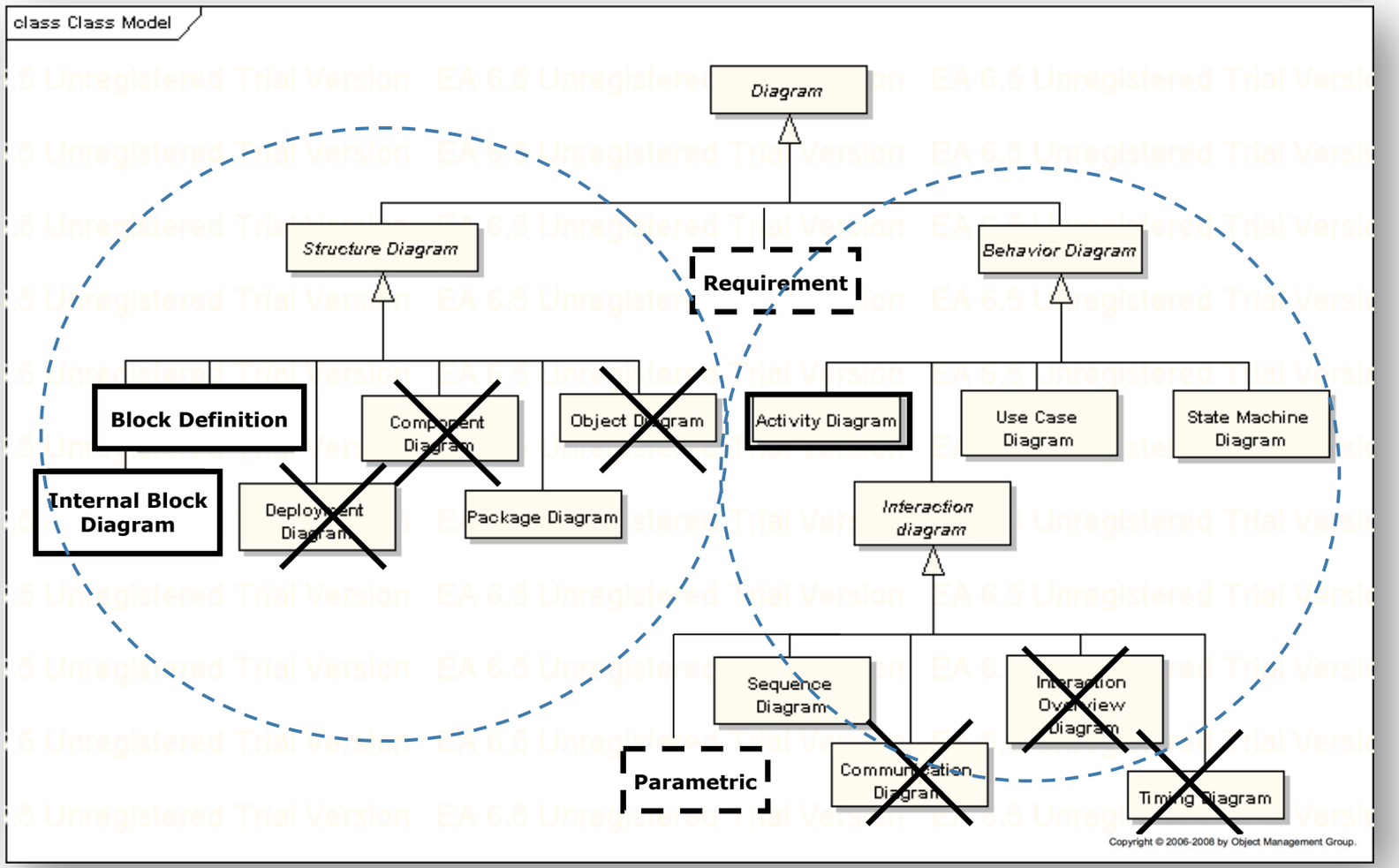
- **SysML**: the OMG Systems Modeling Language
  - ➡ **Systems** Engineering
- **UML**: the OMG Software Modeling Language
  - ➡ **Software** Engineering



UML: 13 diagrams



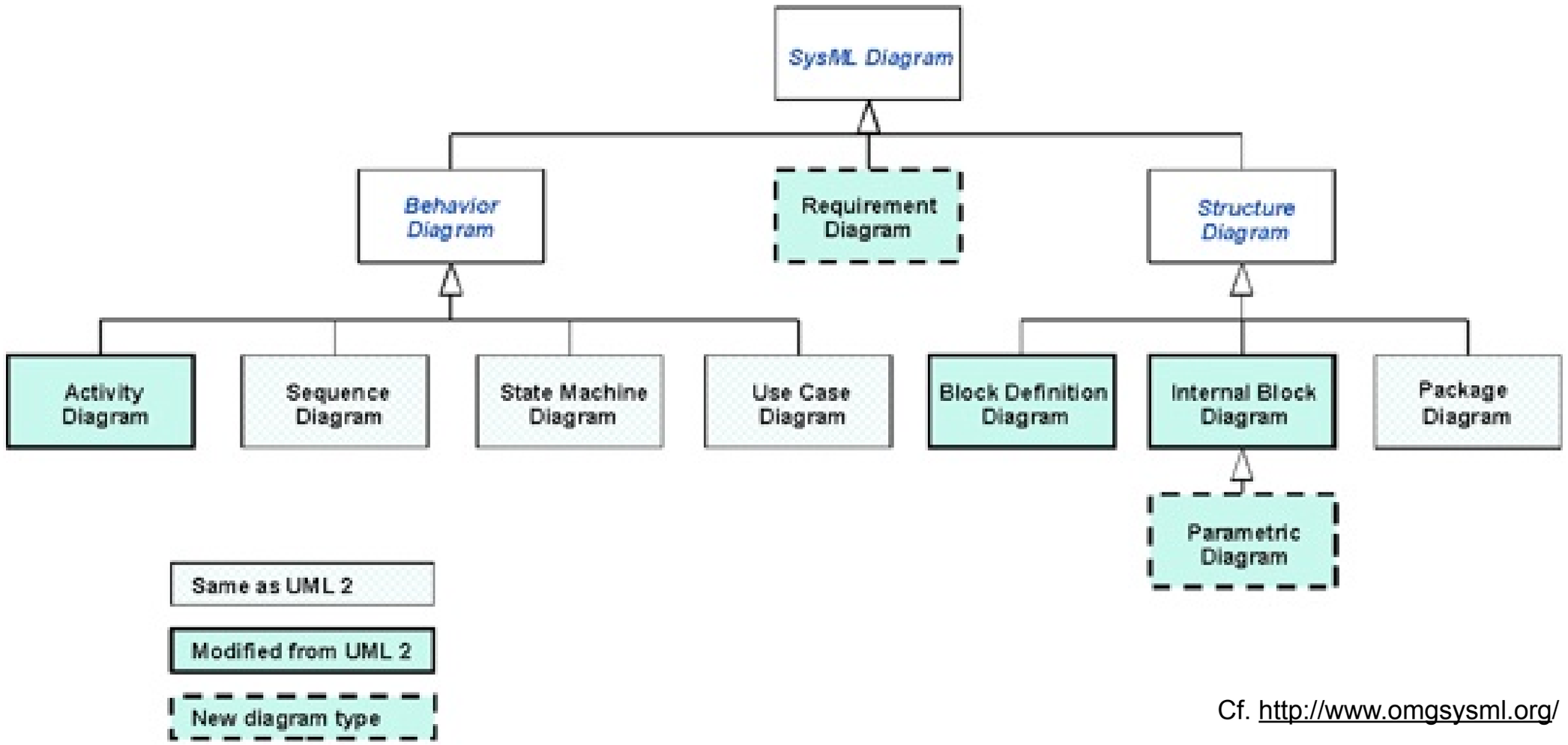
SysML: 13-6+2 = 9 diagrams



SysML diagrams

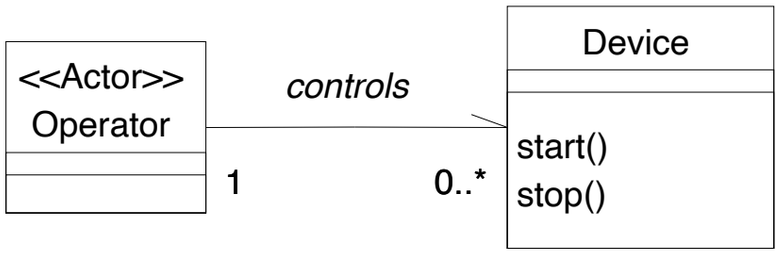
- Structure diagrams
  - The Block Definition Diagram (BDD), replacing the UML2 class diagram
  - The Internal Block Diagram (IBD), replacing the UML2 composite structure diagram
  - The Parametric Diagram, a SysML extension to analyse critical system parameters
  - The Package Diagram remains unchanged
- 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

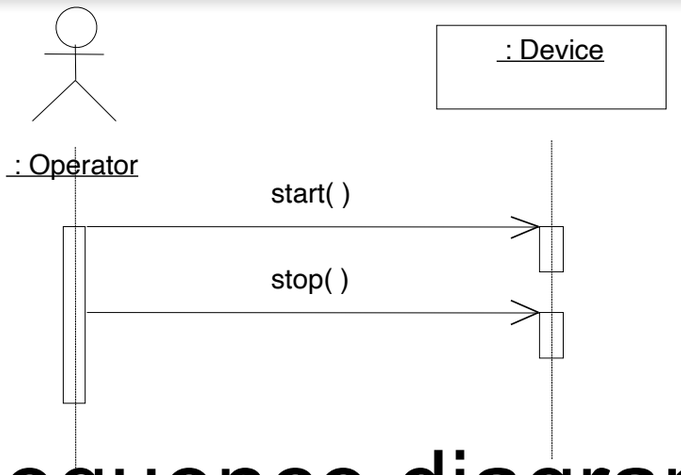


Cf. http://www.omgsysml.org/

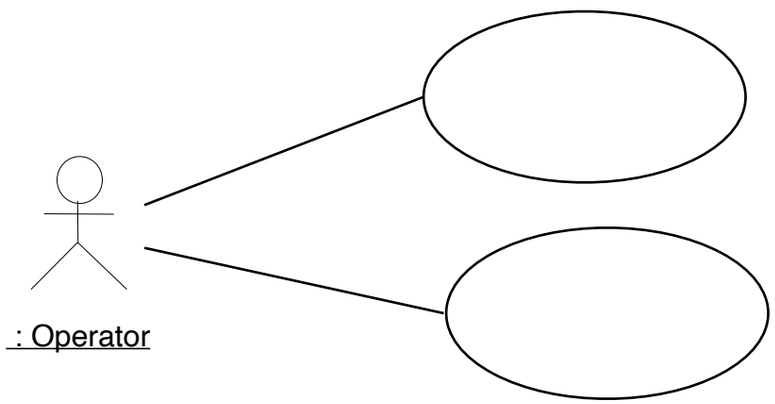
The Four Pillars of UML



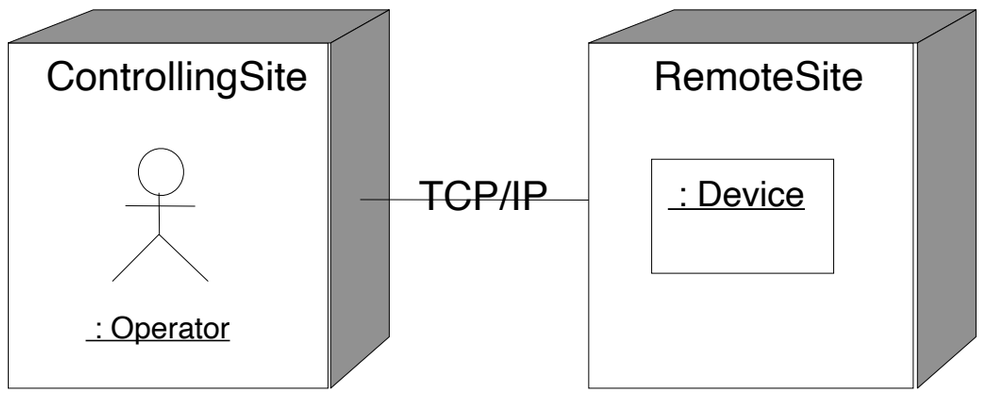
Class diagram



Sequence diagram

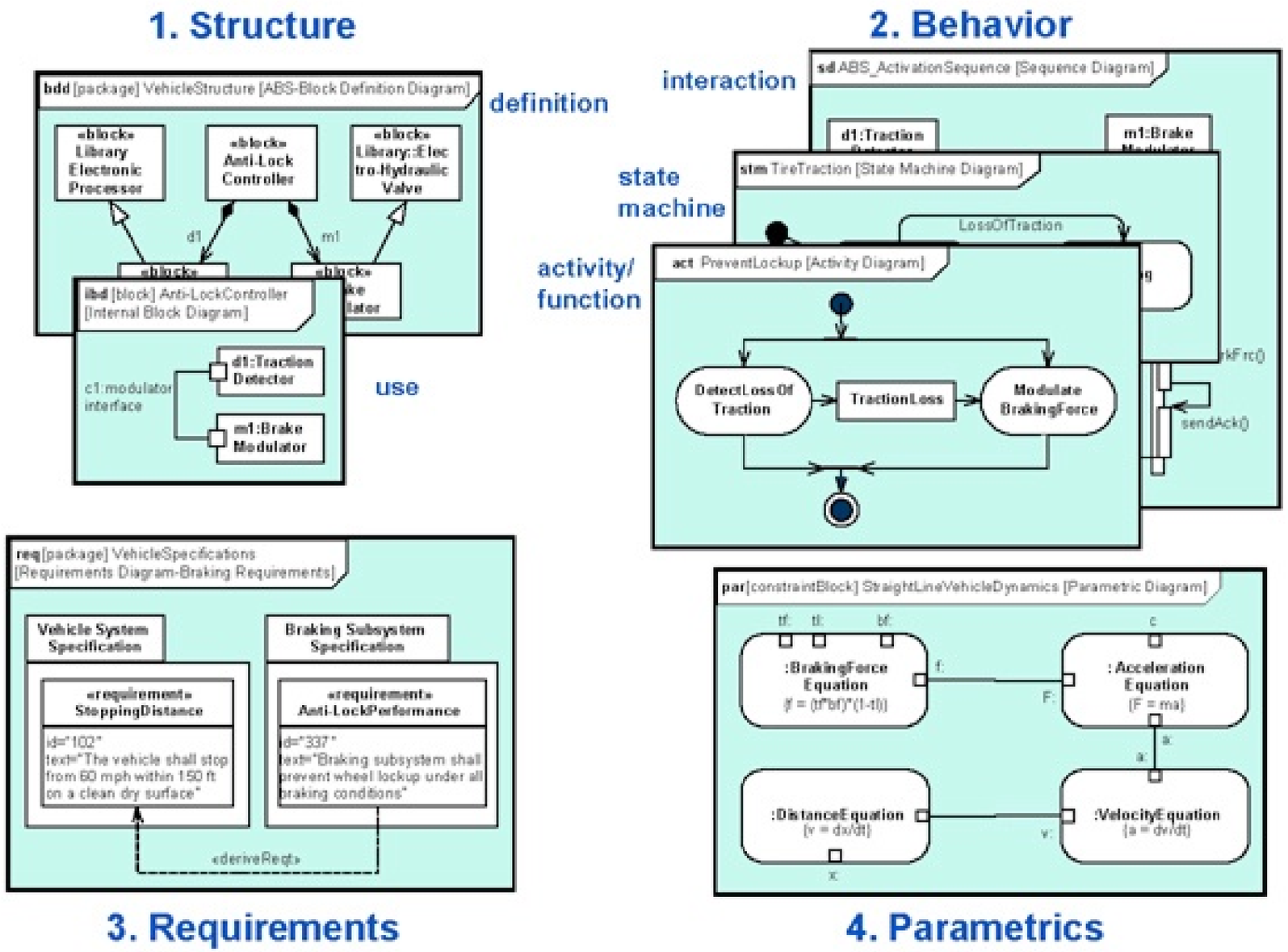


UseCase diagram



Implementation diagram

The Four Pillars of SysML



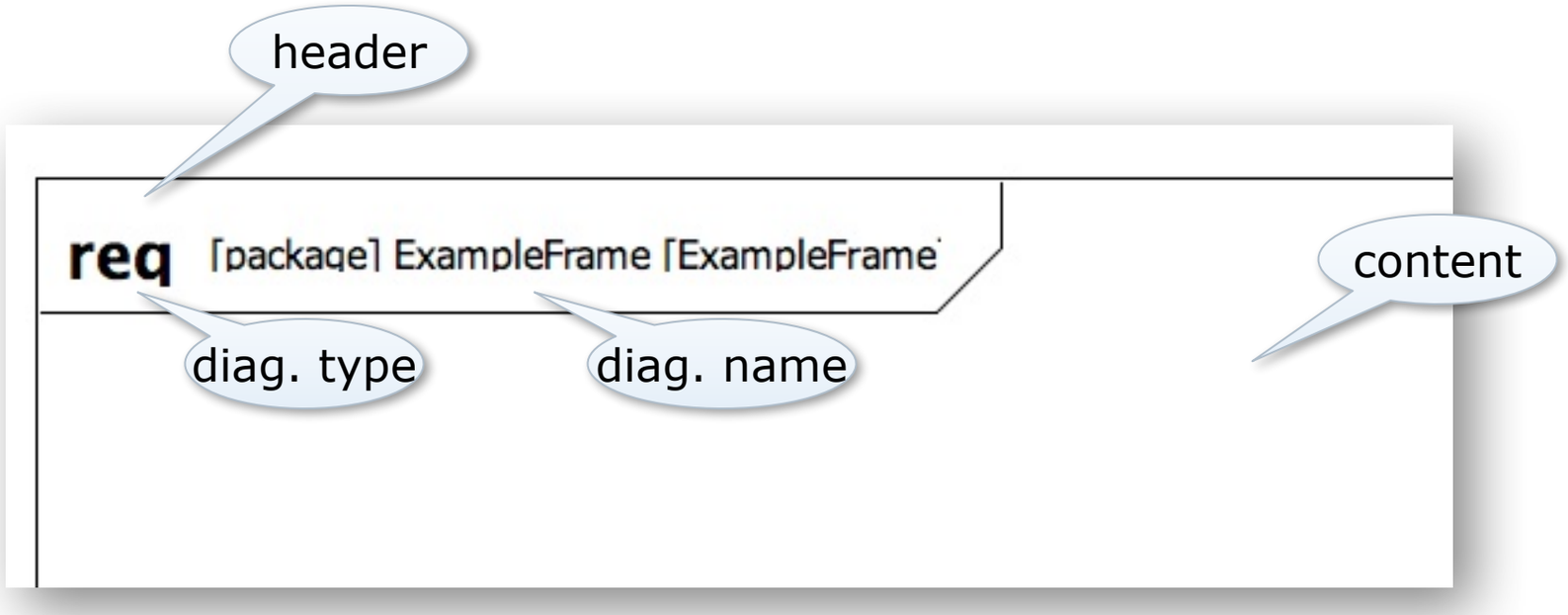
Note that the Package and Use Case diagrams are not shown in this example, but are respectively part of the structure and behavior pillars

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.)

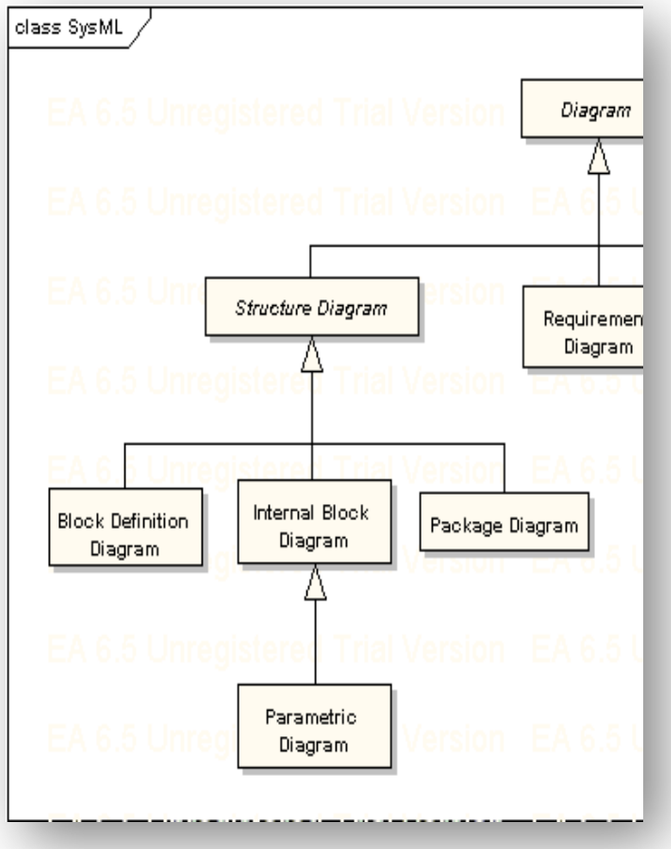


Outline

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

SysML Structure Diagrams

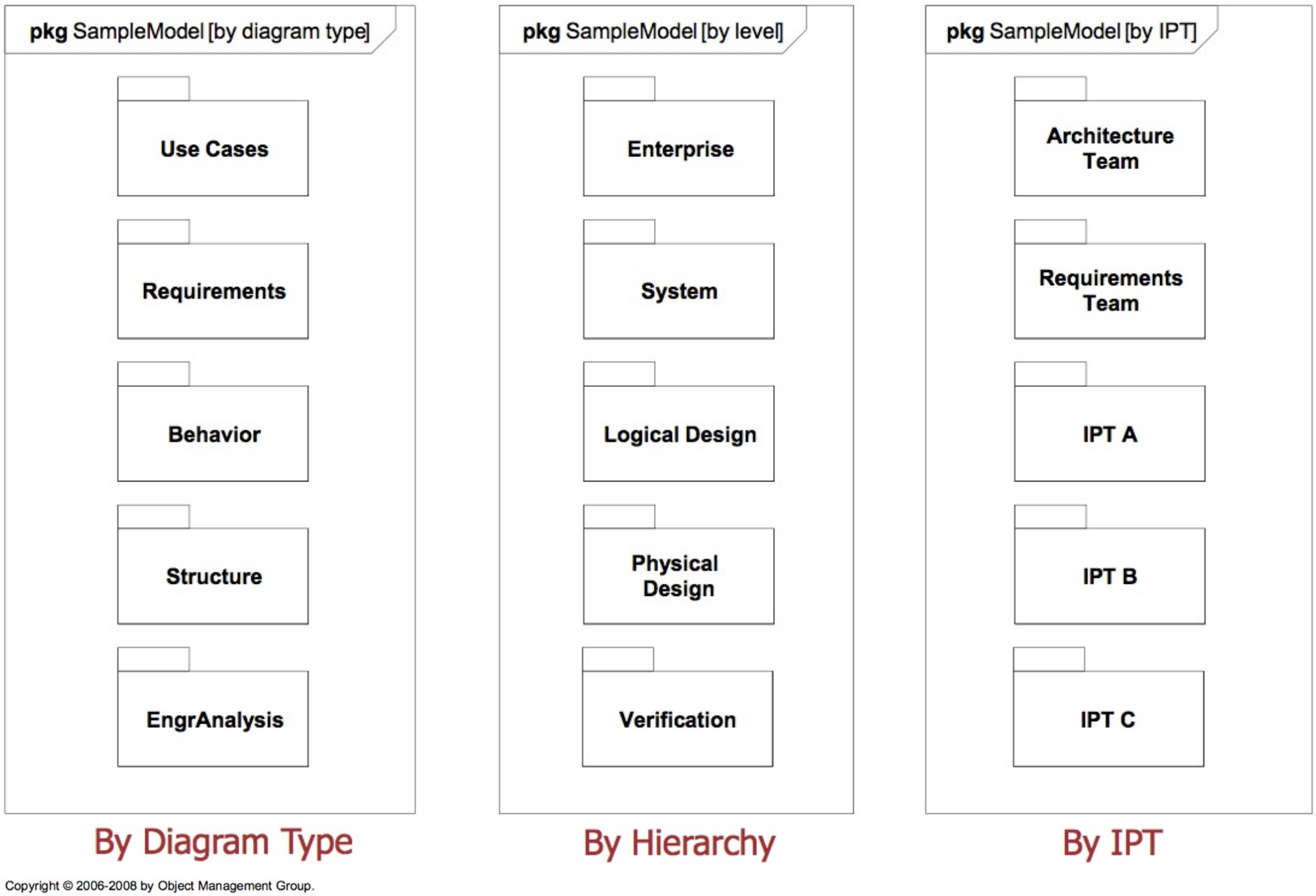
- Package Diagram
- Block Definition Diagram
- Internal Block Definition Diagram
- Parametric Diagram



Package Diagram (pkg)

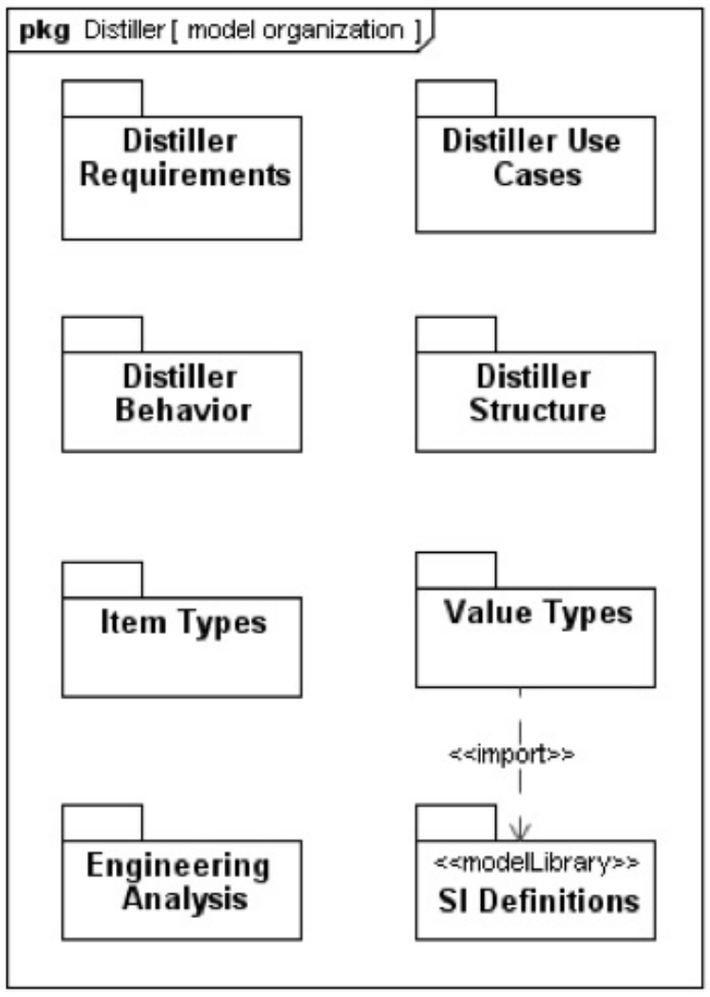
- ~Same as UML
- To organize the model
  - Groups model elements into a name space
  - Often represented in tool browser
  - Supports model configuration management (check-in/out)
- Model can be organized in multiple ways:
  - System hierarchy (e.g., enterprise, system, component)
  - Diagram kind (e.g.,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

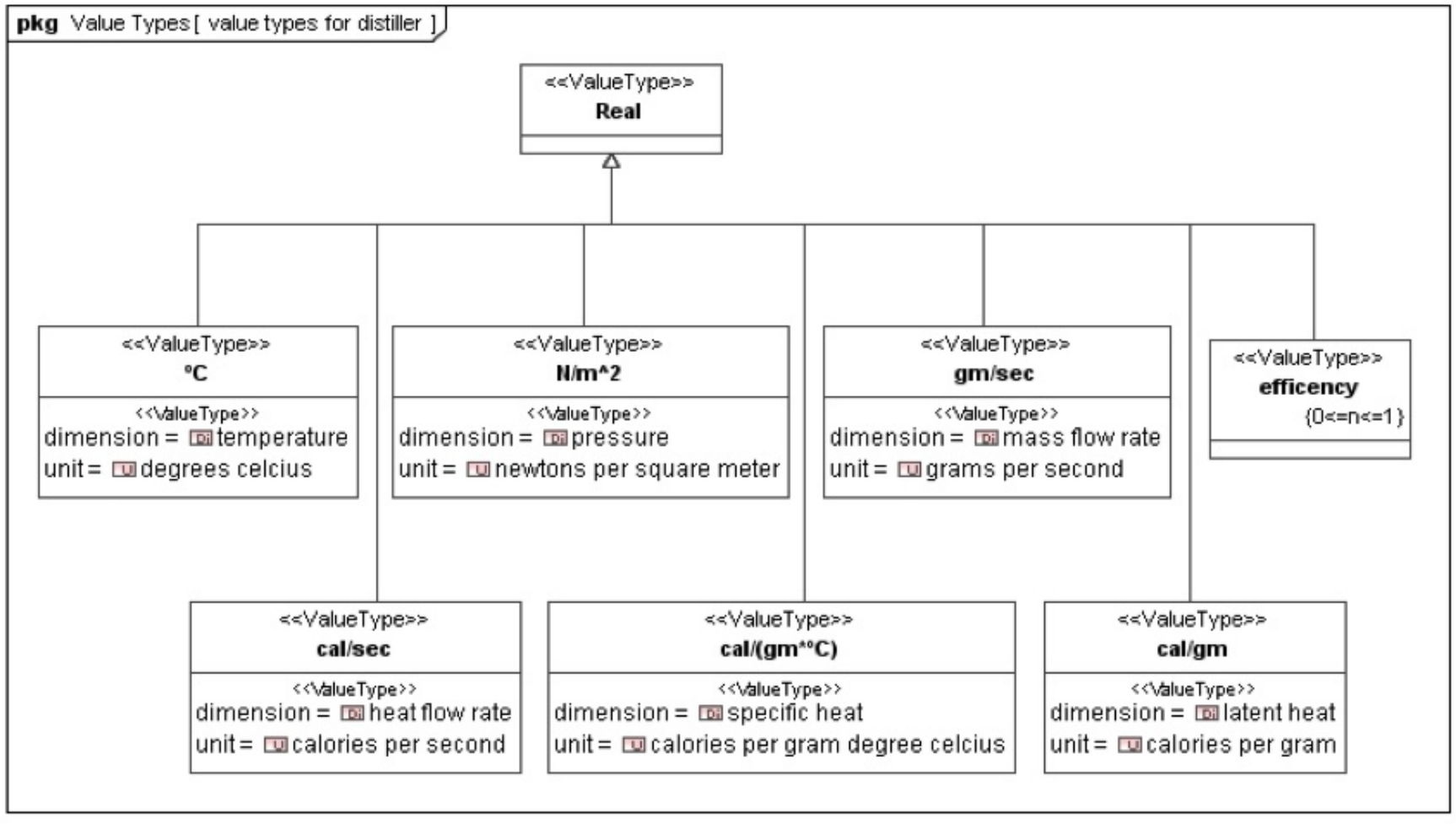
Package Diagram (pkg)



Copyright © 2006-2008 by Object Management Group.

OMG Distiller Example (pkg)





Copyright © 2006-2008 by Object Management Group.

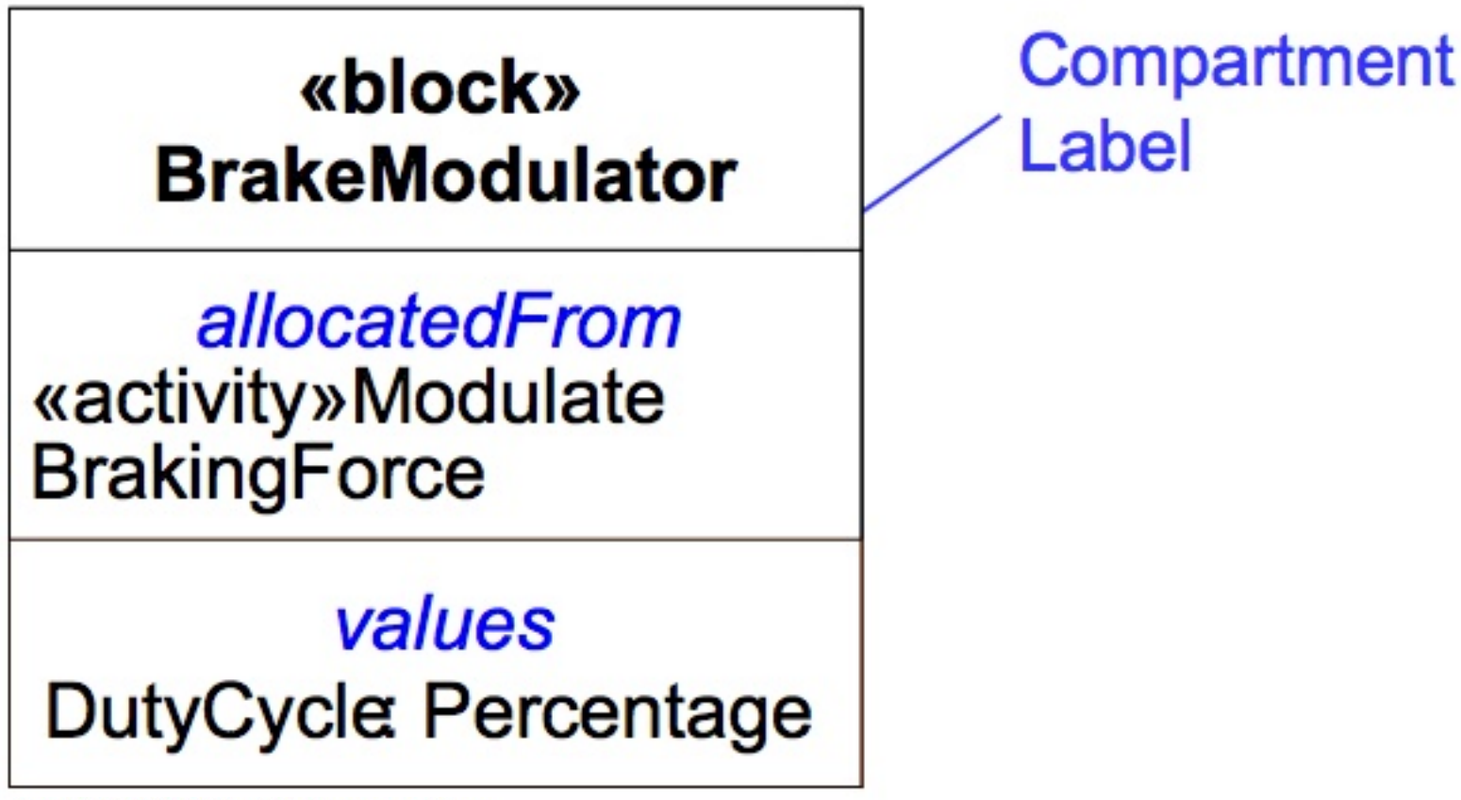
Block Definition Diagram (bdd)

- 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

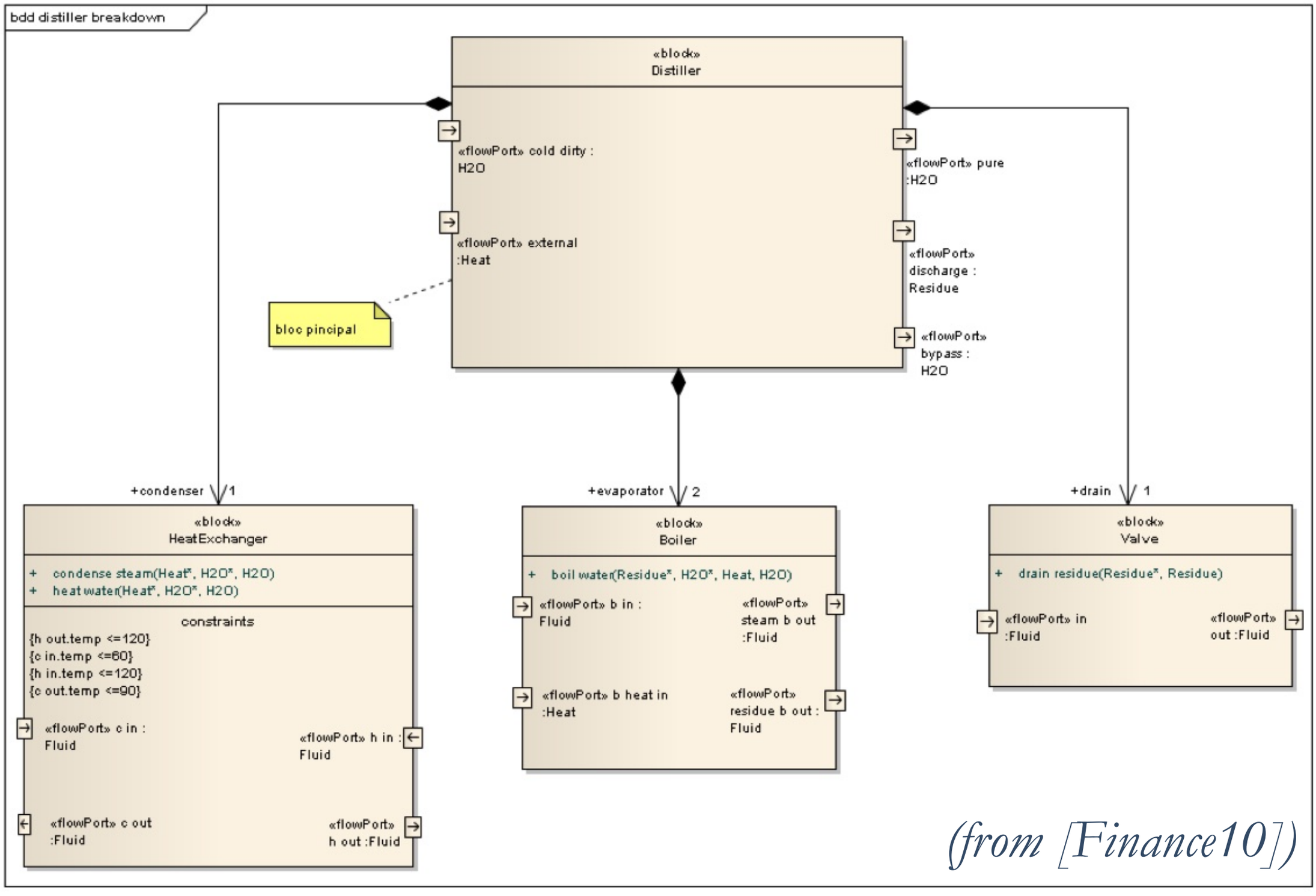
- Compartments
  - Properties
  - Operations
  - Constraints
  - Allocations
  - Requirements



Copyright © 2006-2008 by Object Management Group.

- User defined!

OMG Distiller Example (bdd)



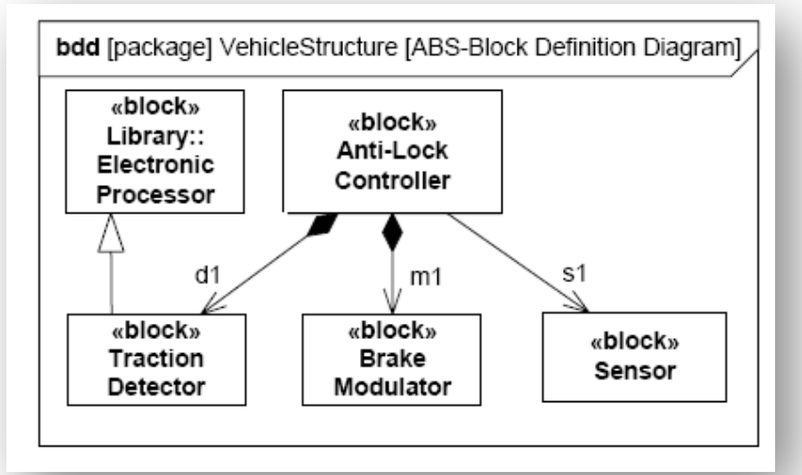
(from [Finance10])

Internal Block Diagram (ibd)

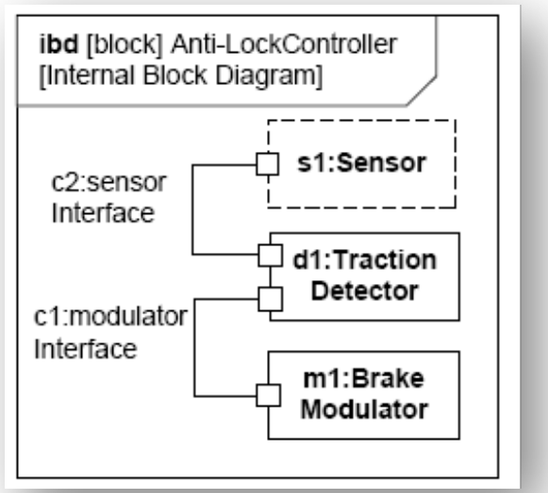
- Provides the white box or internal view of a system block
- Usually instantiated from the BDD to represent the final assembly
- Composite blocks from the BDD are instantiated on the IBD as parts
- Parts are assembled through connectors, linking them directly or via their ports (standard and/or flow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

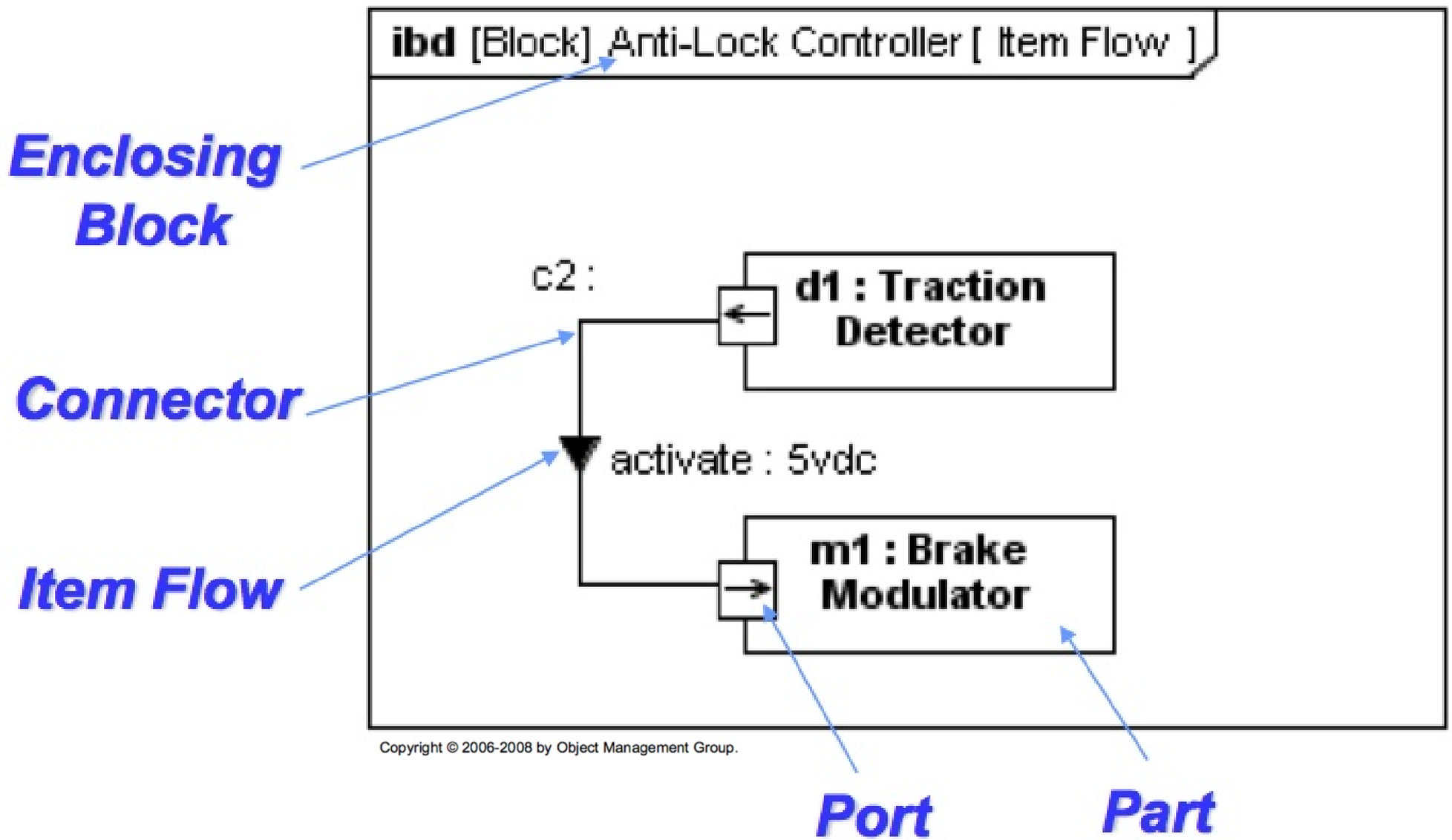


Block Definition Diagram



Internal Block Diagram

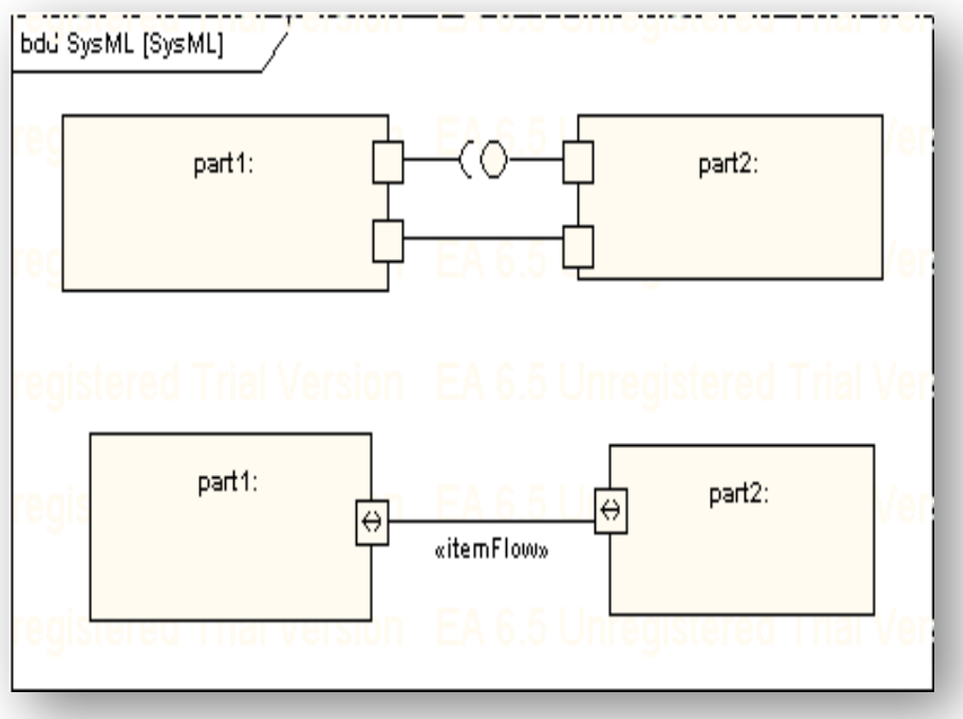
Internal Block Diagram (ibd)



SysML Ports

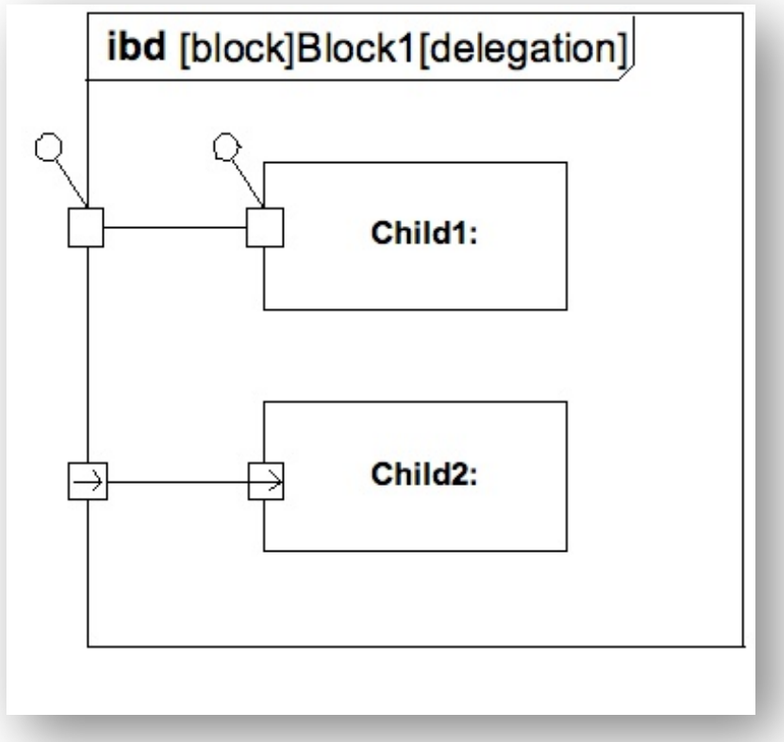
- **Specifies interaction points on blocks and parts**
  - ➡ Integrates behavior with structure

- 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

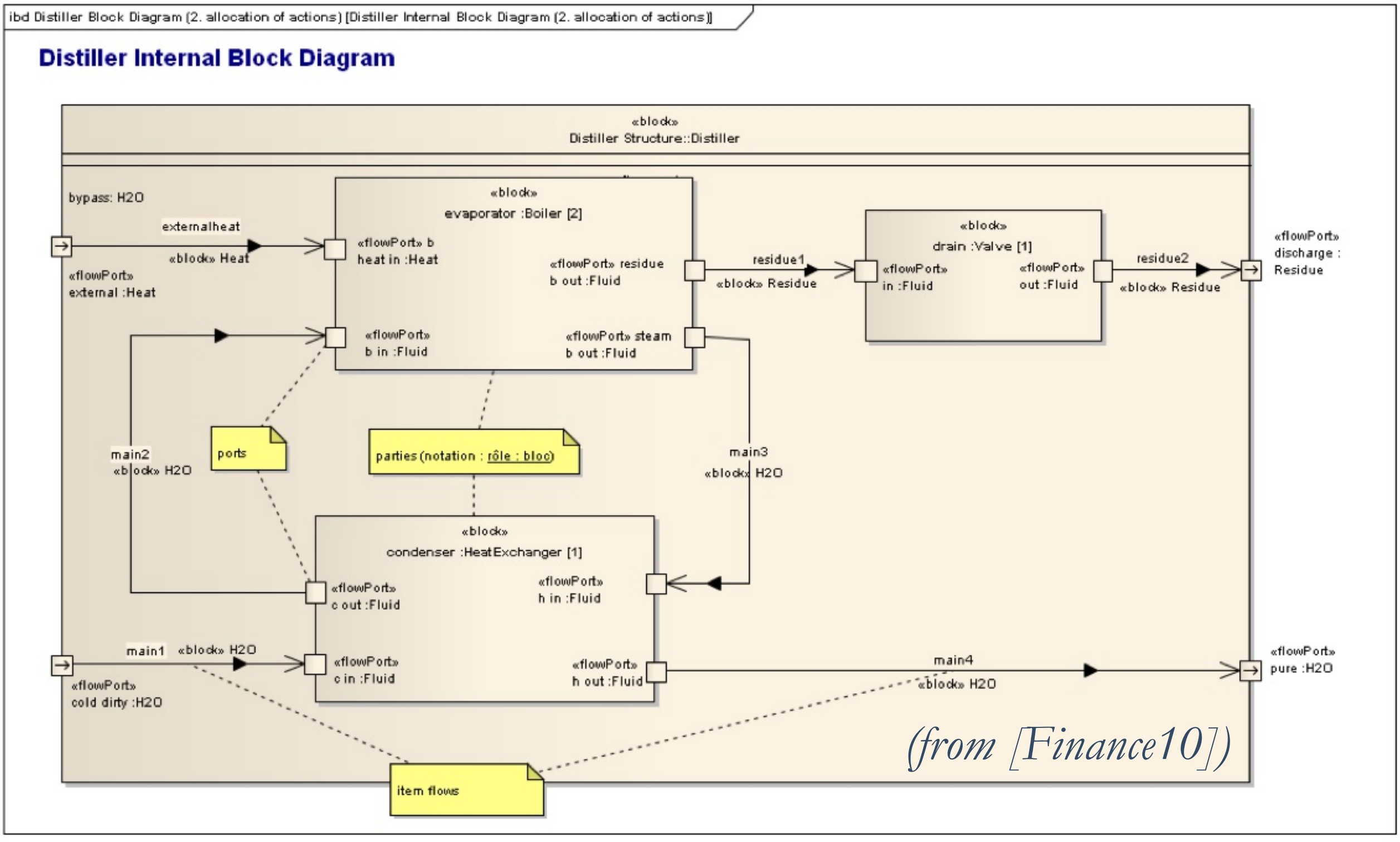


SysML Ports: **delegation**

- 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 boundaries** without requiring ports at each level of nested hierarchy



OMG Distiller Example (ibd)

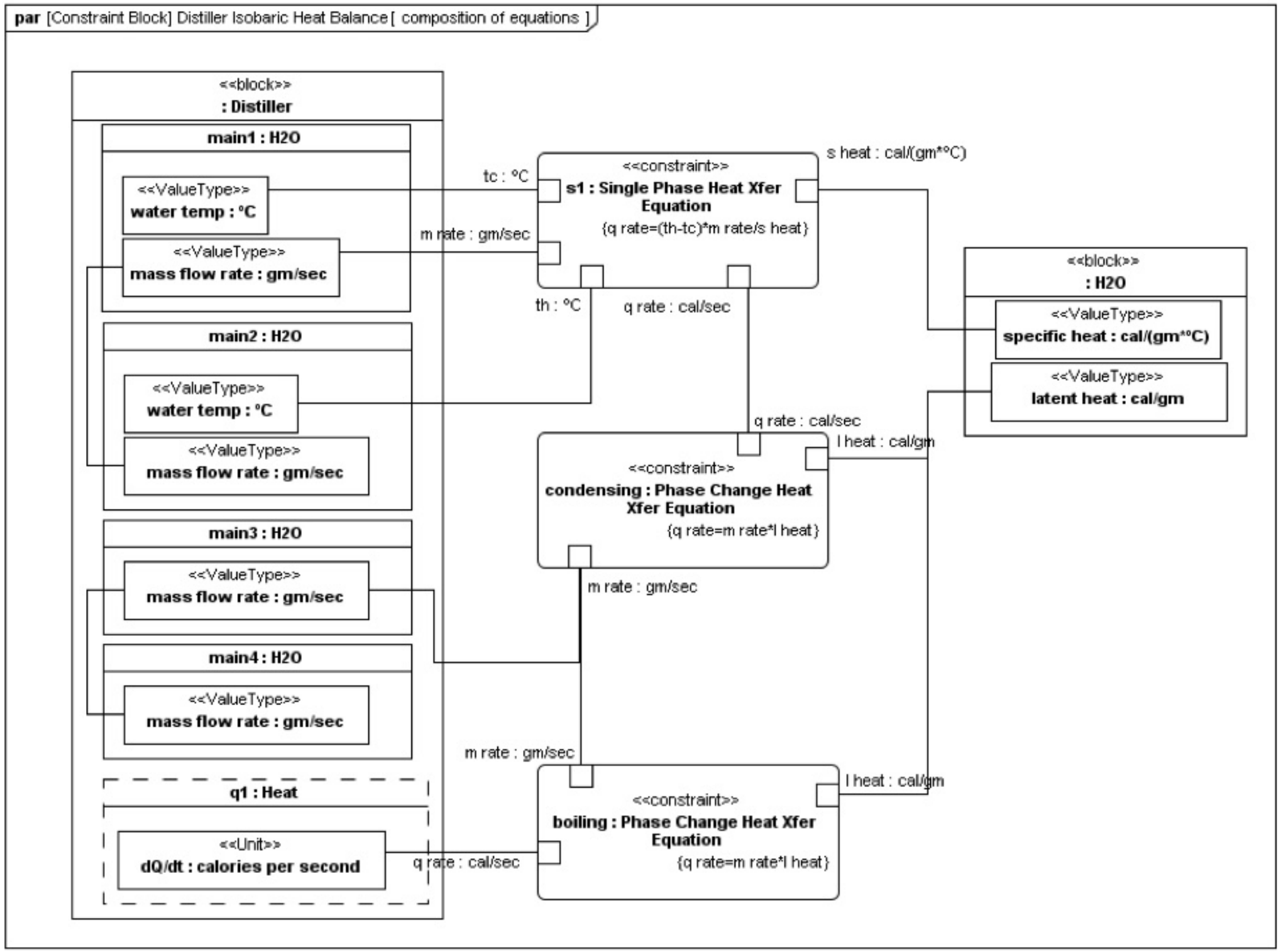


Parametric Diagram (par)

(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

OMG Distiller Example (par)



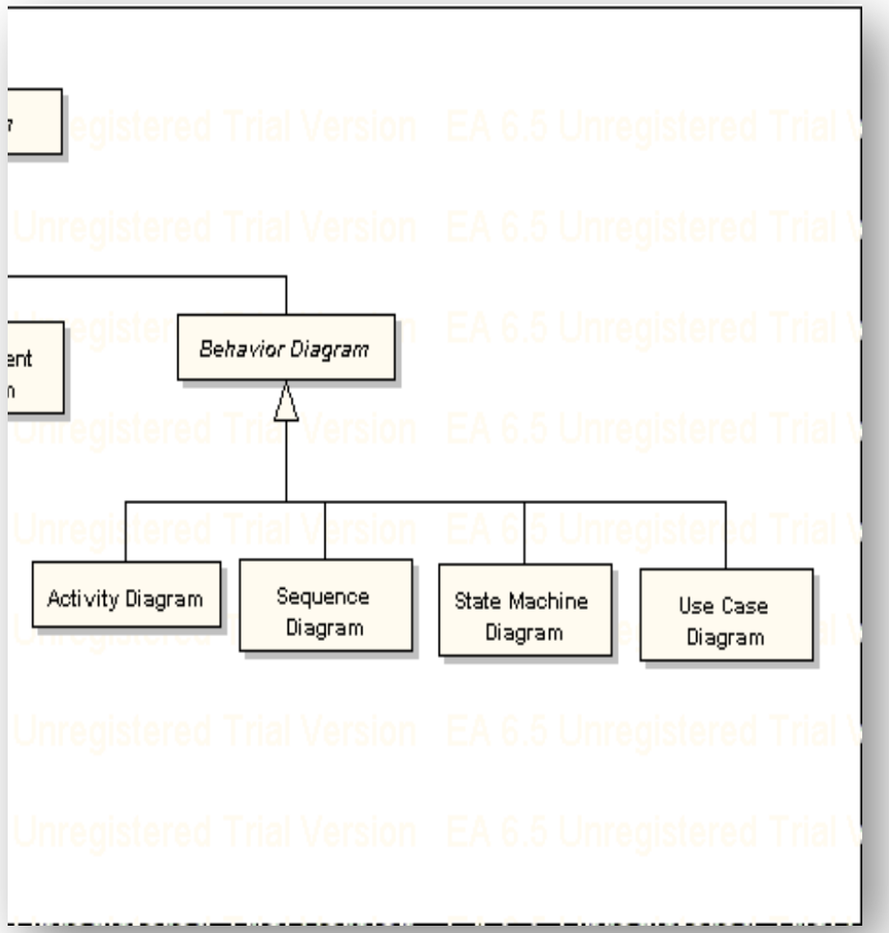
Copyright © 2006-2008 by Object Management Group.

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)

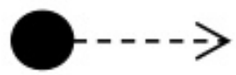
- to specify
  - controlled sequence of actions
  - the **flow** of inputs/outputs
  - **control**, including sequence and conditions for coordinate activities

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

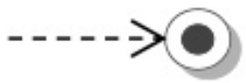
Activity Diagram (act)

- 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

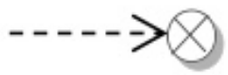
Routing Flow



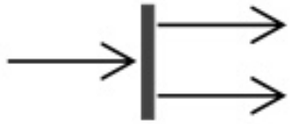
Initial Node – On execution of parent control token placed on outgoing control flows



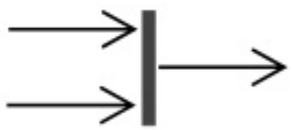
Activity Final Node – Receipt of a control token terminates parent



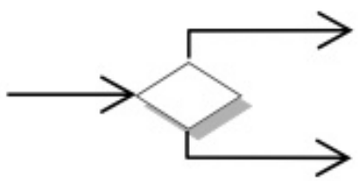
Flow Final Node – Sink for control tokens



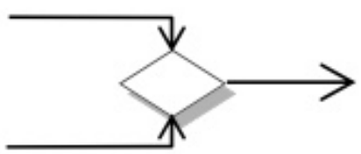
Fork Node – Duplicates input (control or object) tokens from its input flow onto all outgoing flows



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



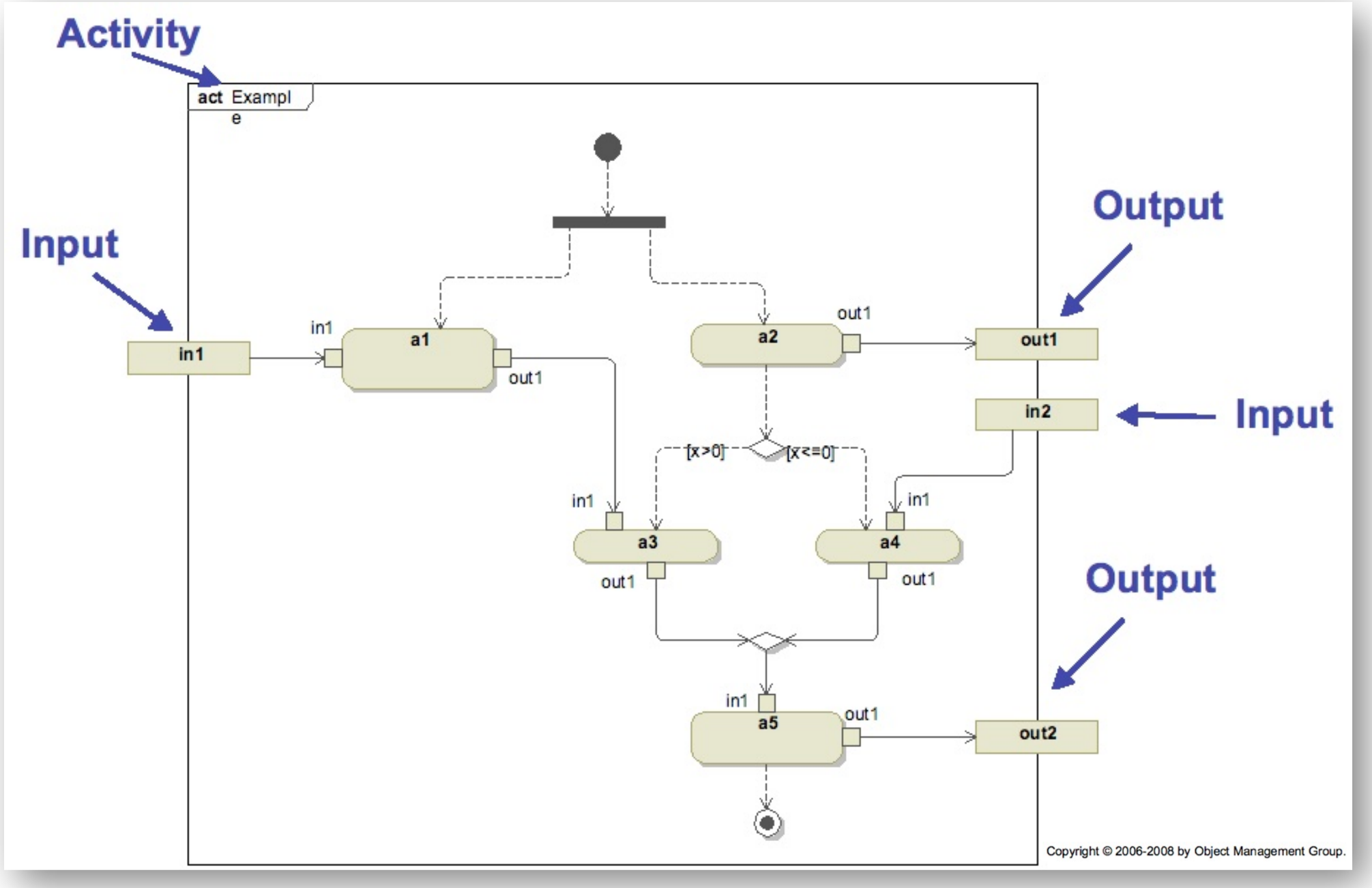
Decision Node – Waits for an input (control or object) token on its input flow and places it on one outgoing flow based on guards



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

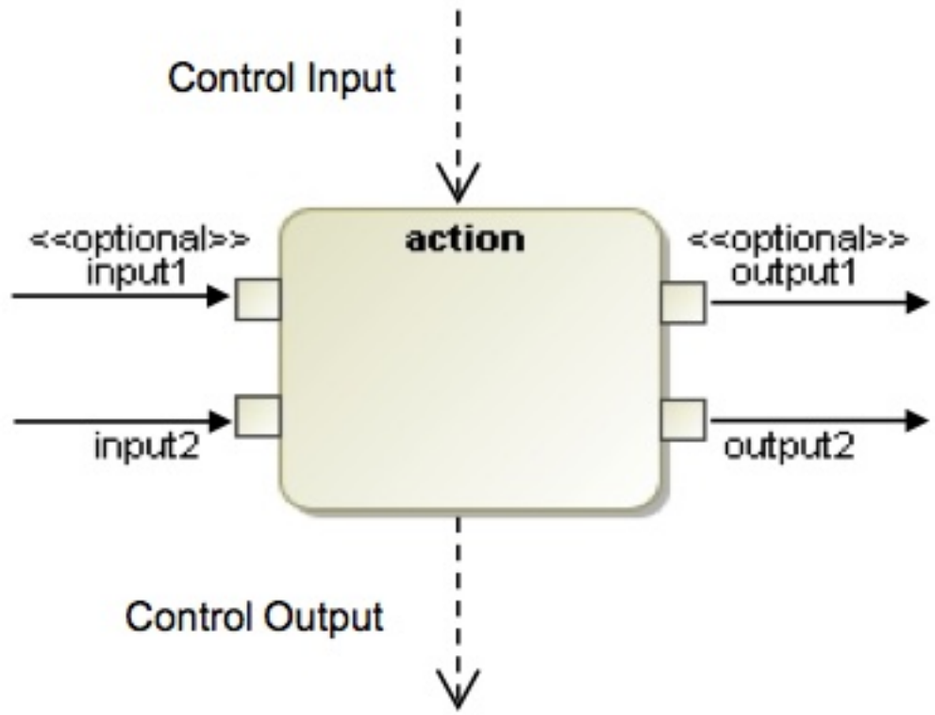
Guard expressions can be applied on all flows

Activity Diagram (act)

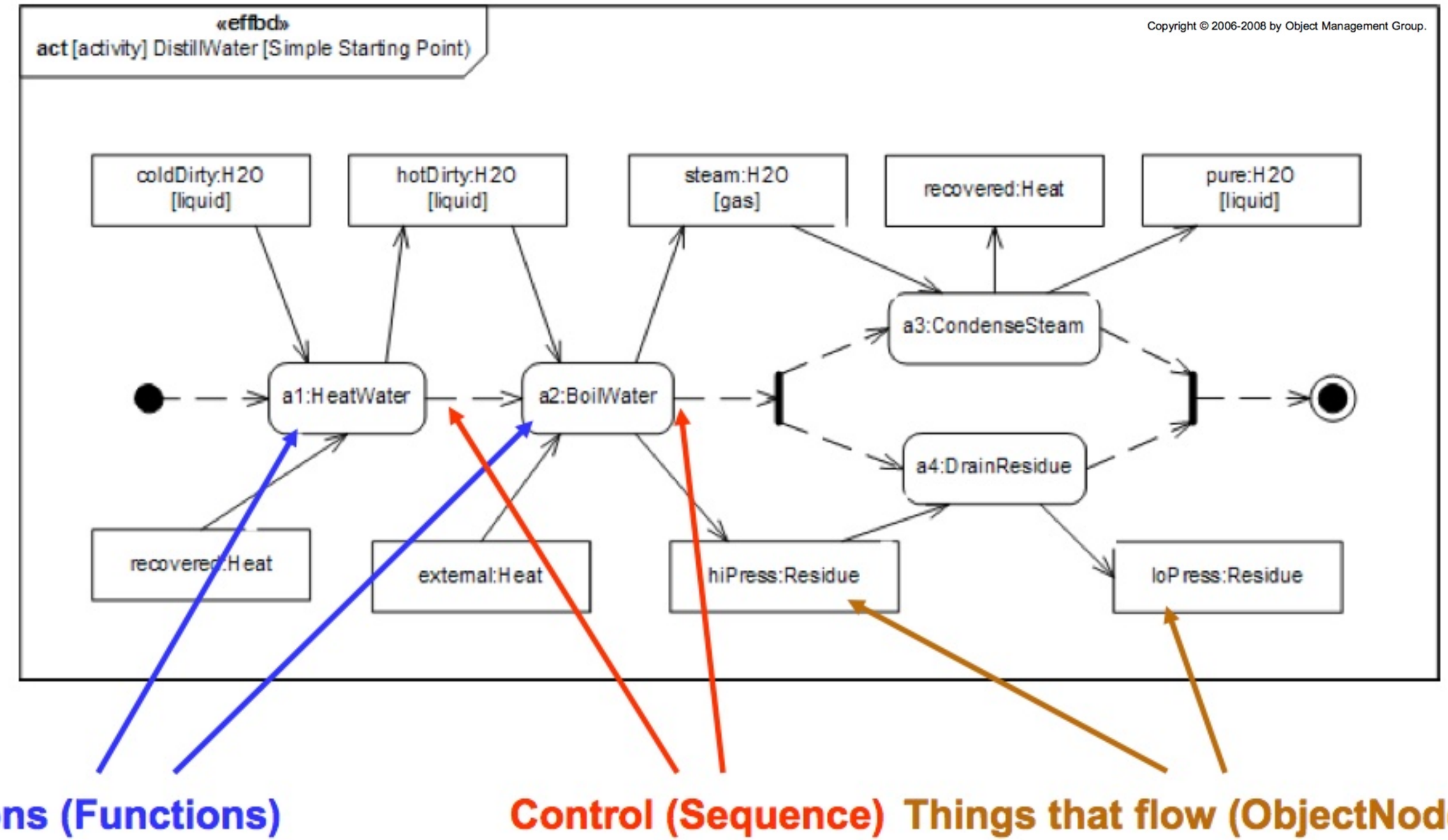


Actions Process Flow of Control and Data

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



OMG Distiller Example (act)

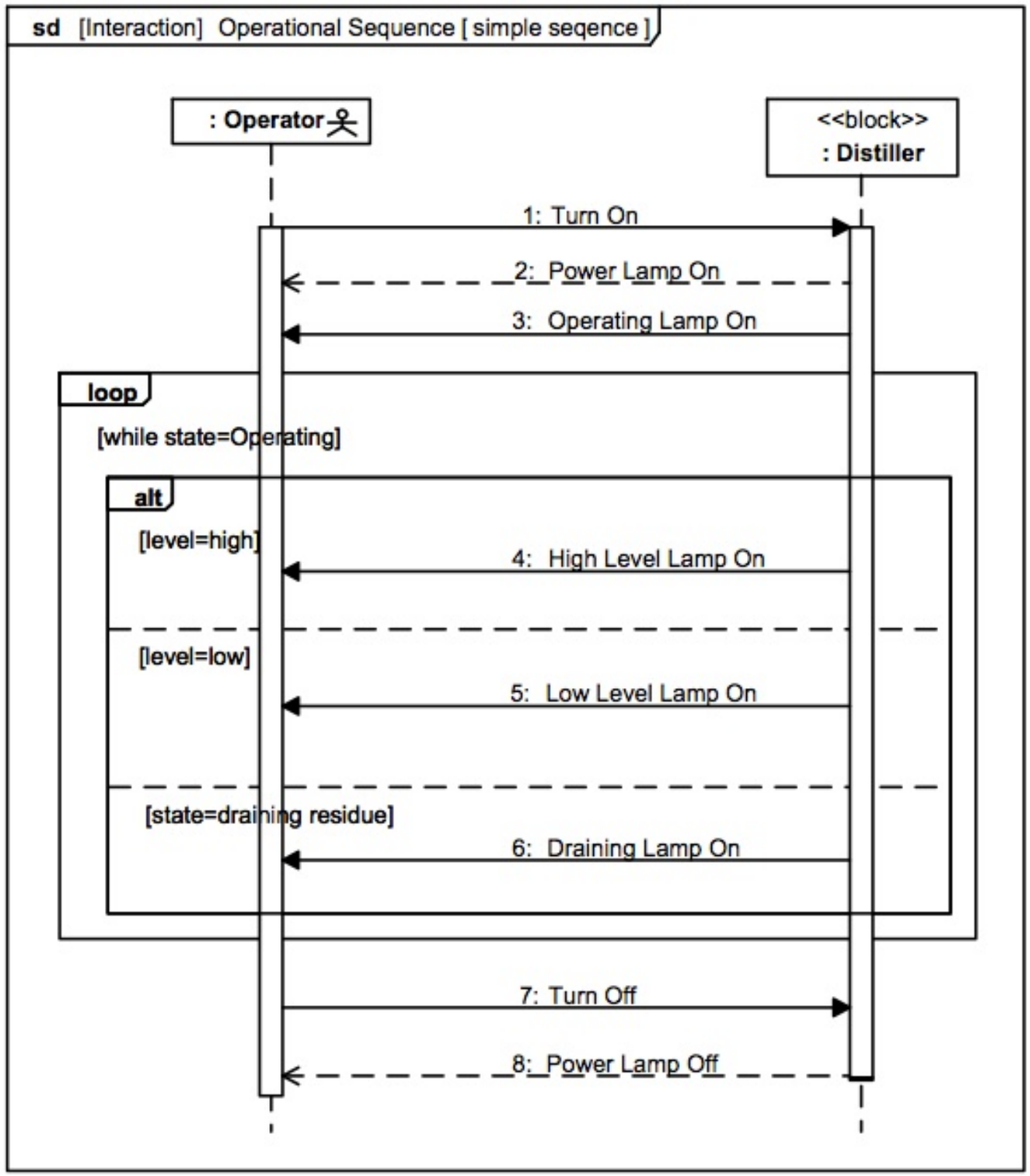


Interaction Diagrams (sdm, sd & uc)

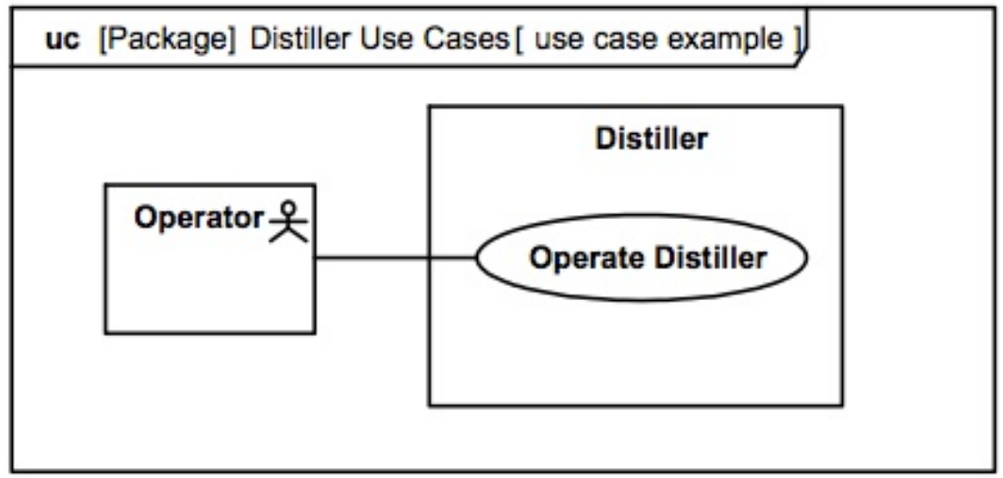
State Machine, Sequence, and Use Case Diagrams:

Like in UML!

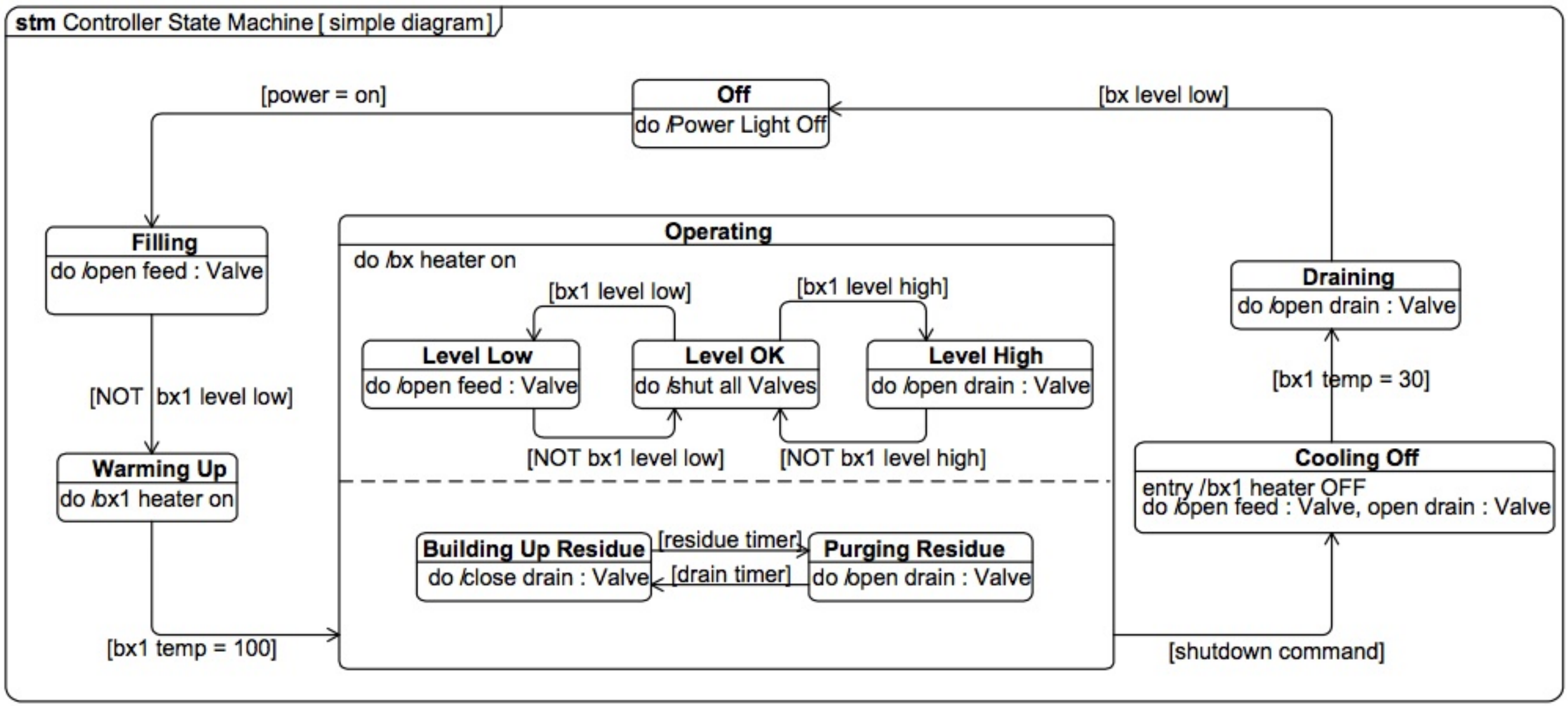
OMG Distiller Example (sd & uc)



Copyright © 2006-2008 by Object Management Group.



OMG Distiller Example (sdm)



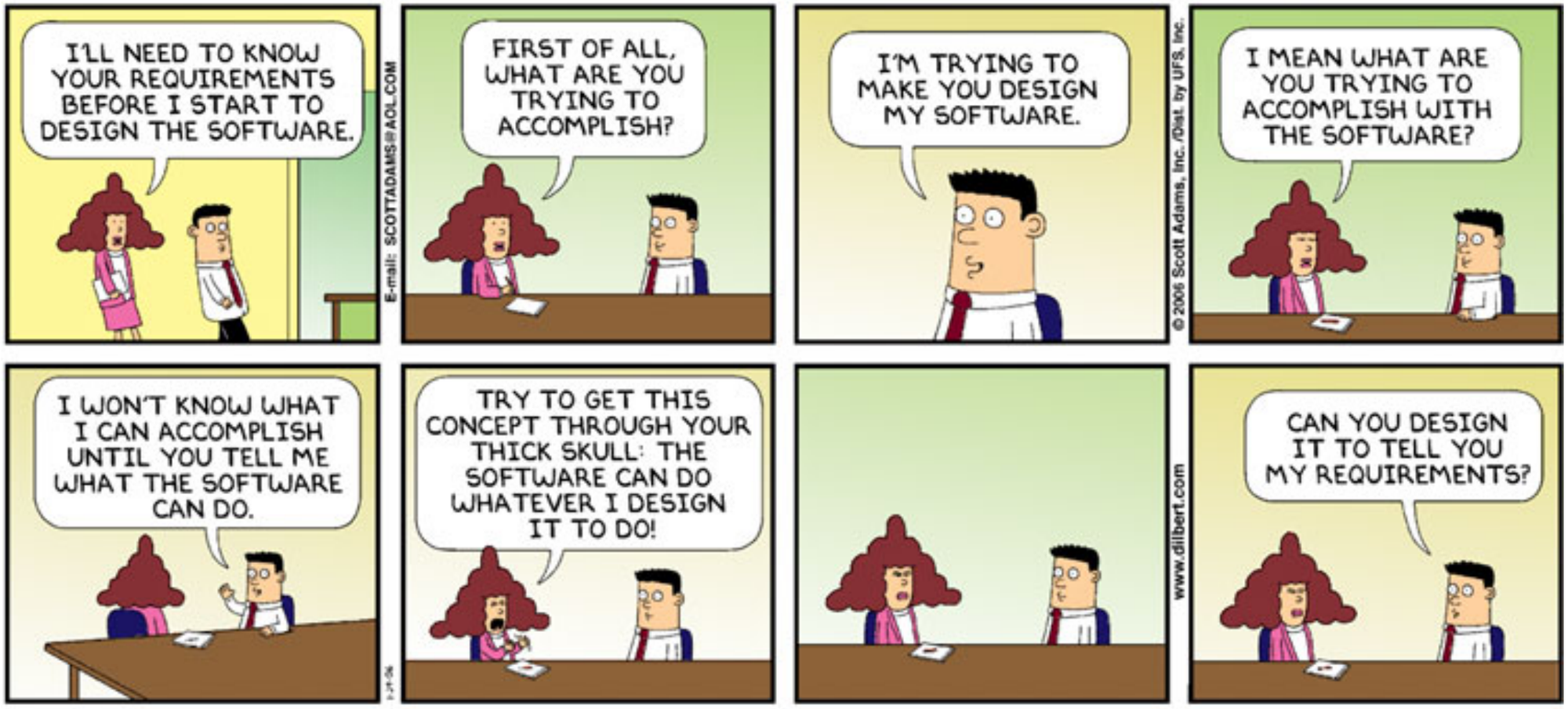
Copyright © 2006-2008 by Object Management Group.

Outline

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

SysML Requirement Diagram

(SysML extension)



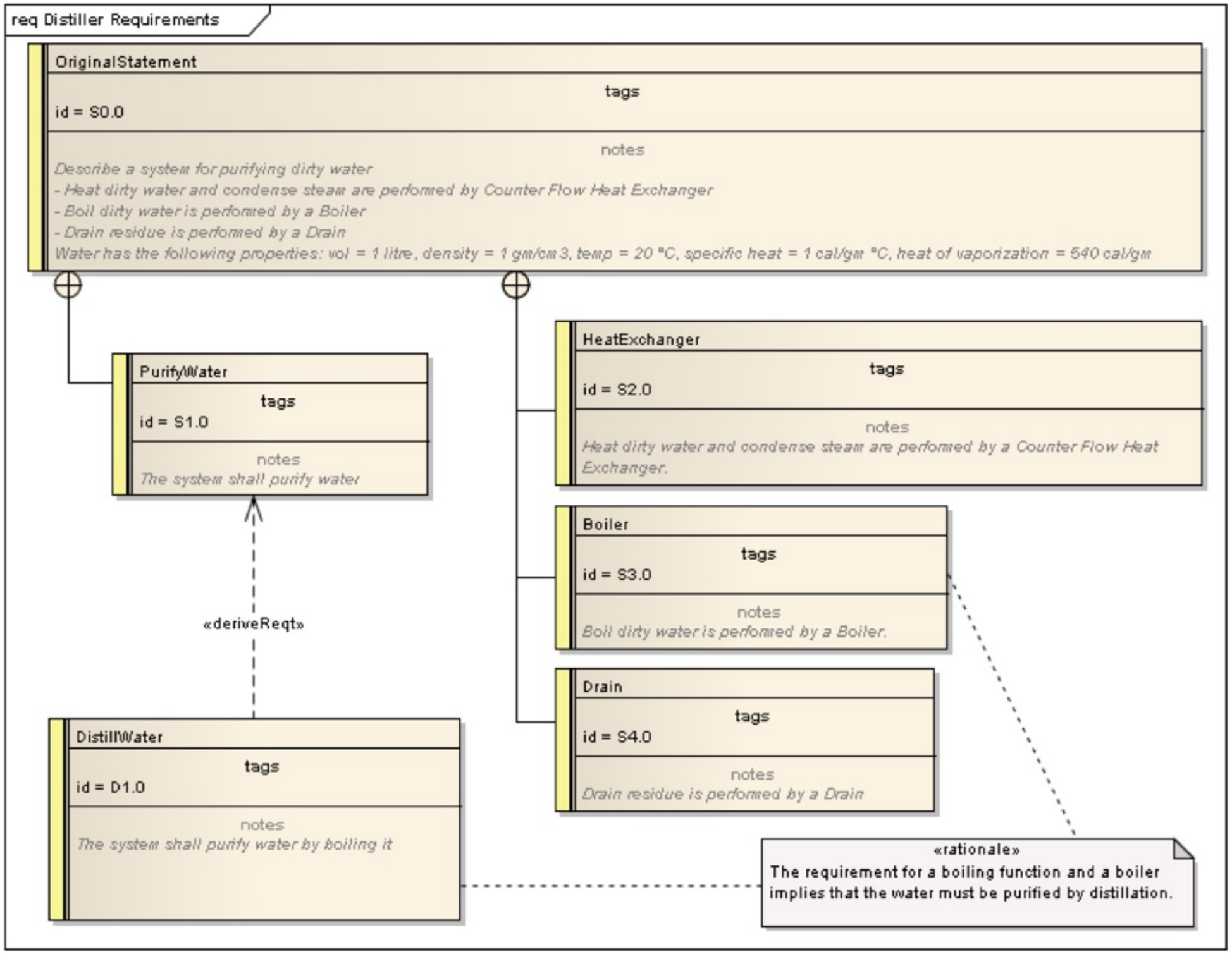
© Scott Adams, Inc./Dist. by UFS, Inc.

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)

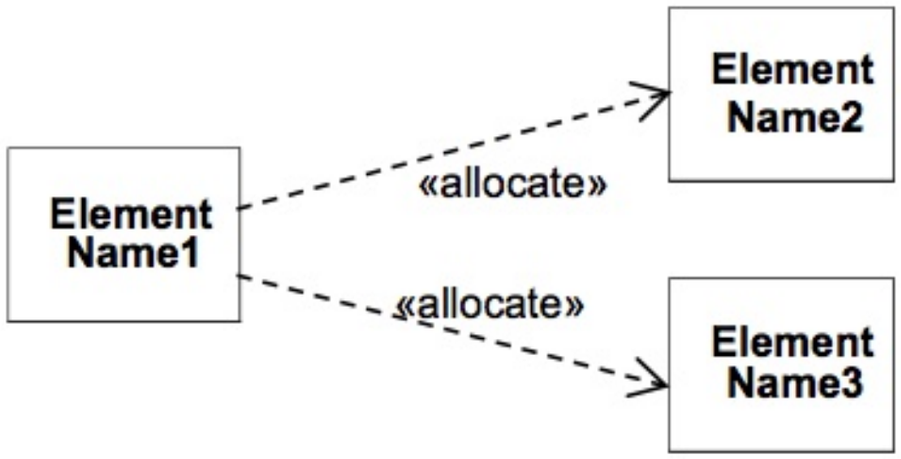


(from [Finance10])

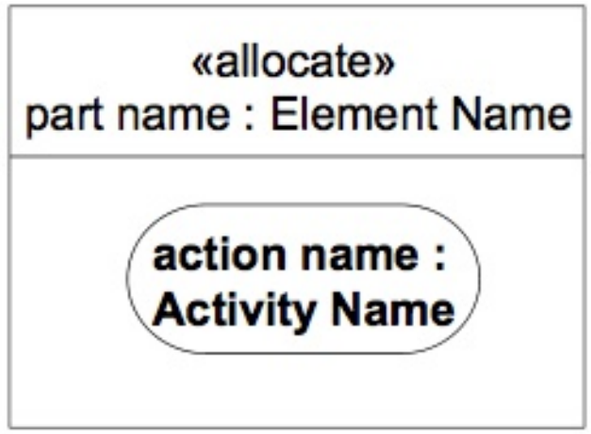
Allocations

- 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).

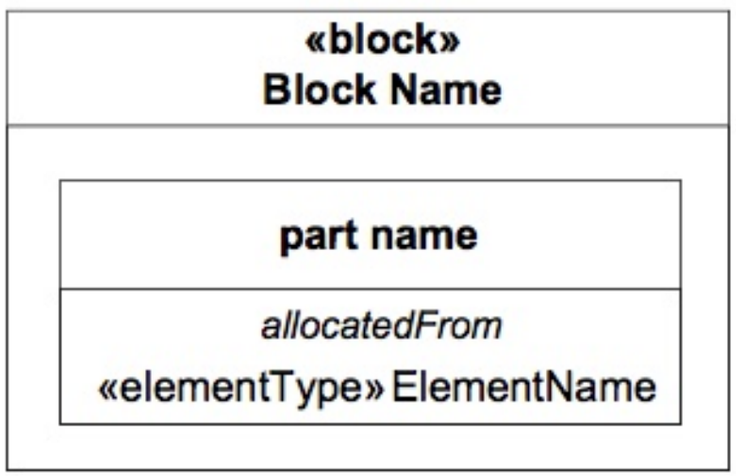
Allocations Representation



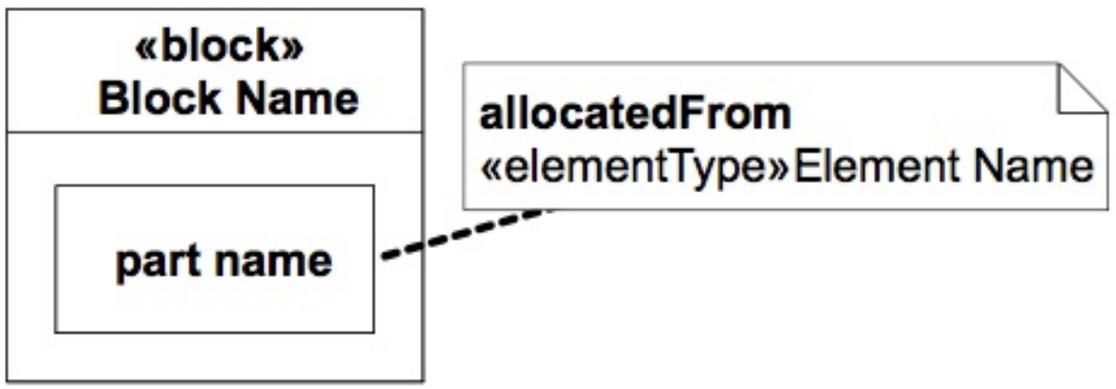
Allocate Relationship



Explicit Allocation of Action to Part Property



Compartment Notation



Callout Notation

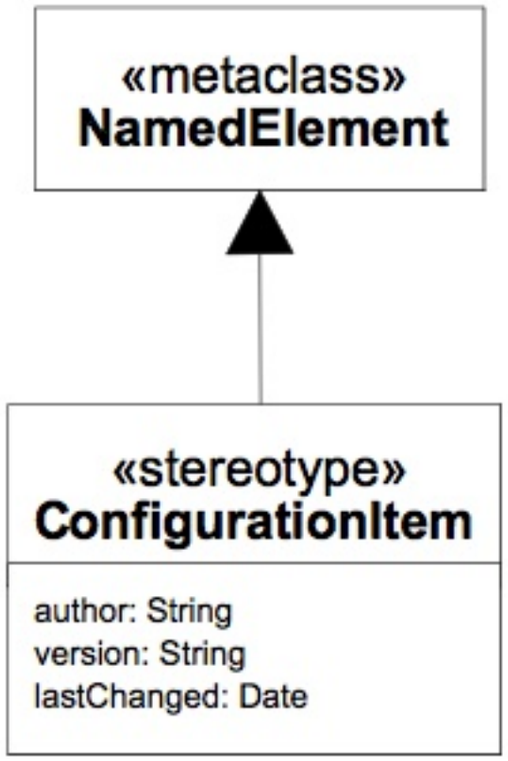
Read as follows: "part name has constraints that are allocated to/from an <<element type>> Element Name"

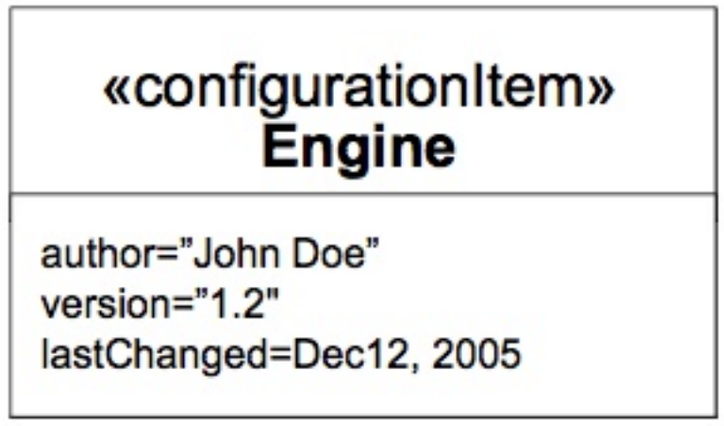
Copyright © 2006-2008 by Object Management Group.

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

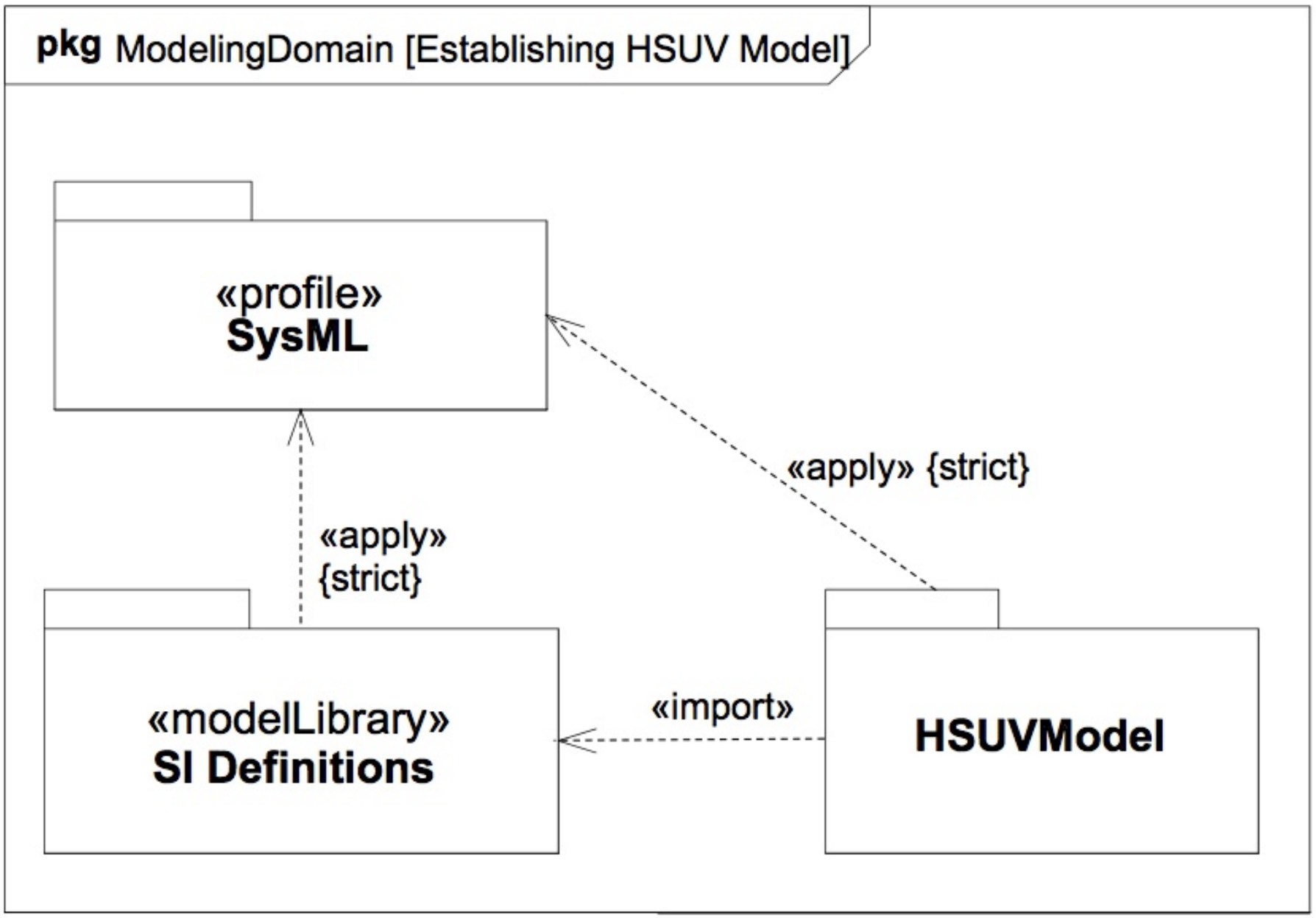




Defining the Stereotype

Applying the Stereotype

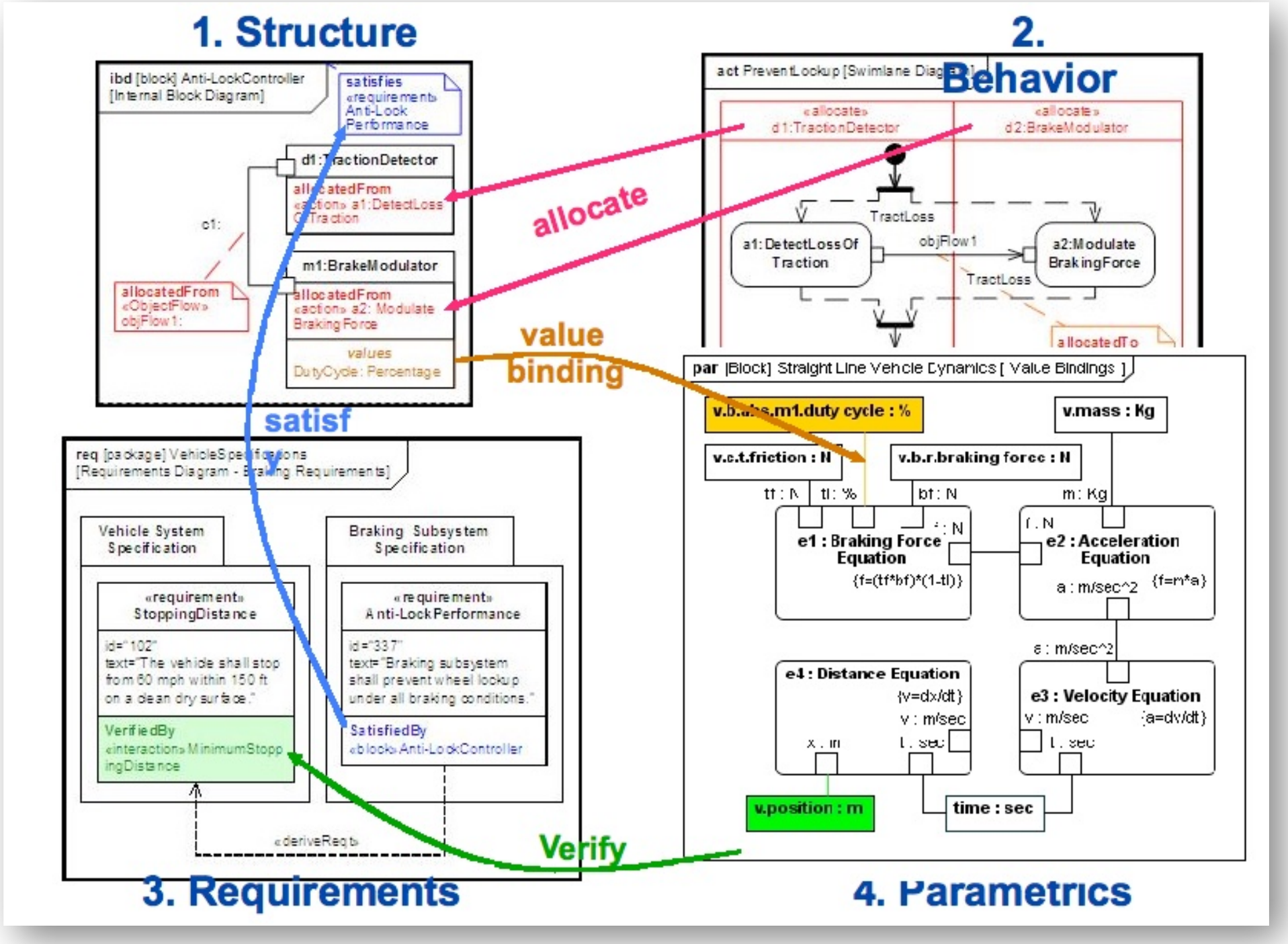
Model Libraries



Outline

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

Cross Connecting Model Elements



Towards SysML v2

- A standalone, formal grounded, layered, and extensible language
- Come first with a textual syntax, and a graphical one as an alternative

- Current proposition: https://drive.google.com/drive/mobile/folders/1L0E3RwO9ch3Ta5Ye4EIdmn15yRPgwS4B

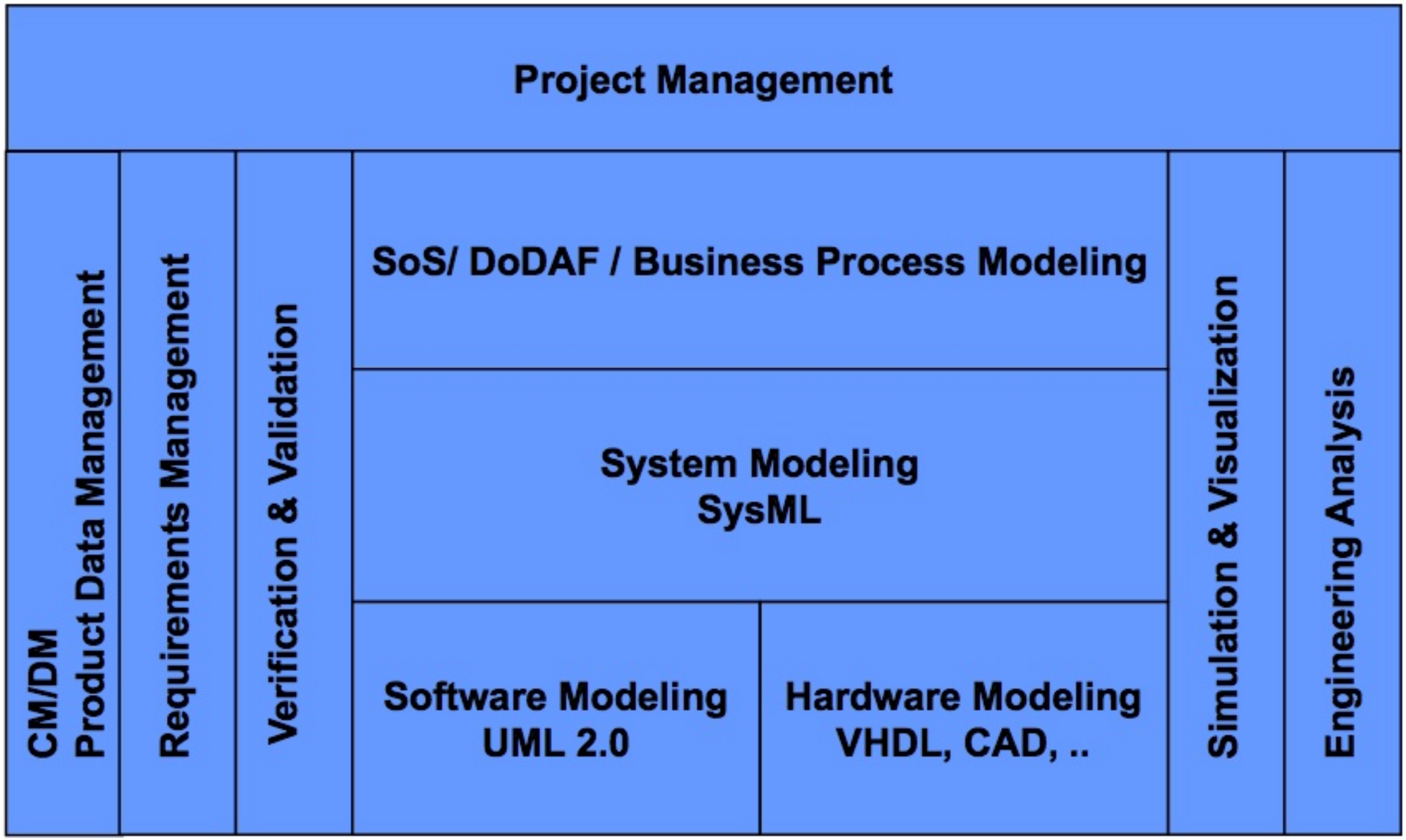
- Complete tutorial on demand

Conclusion

- SysML is:
  - a specific language for complex systems
  - strongly UML-Based
  - focusing on specification, analysis, design and verification

- SysML is not:
  - a method or a tool
  - just a UML profile
  - sufficient in itself

Typical Integrated Tool Environment



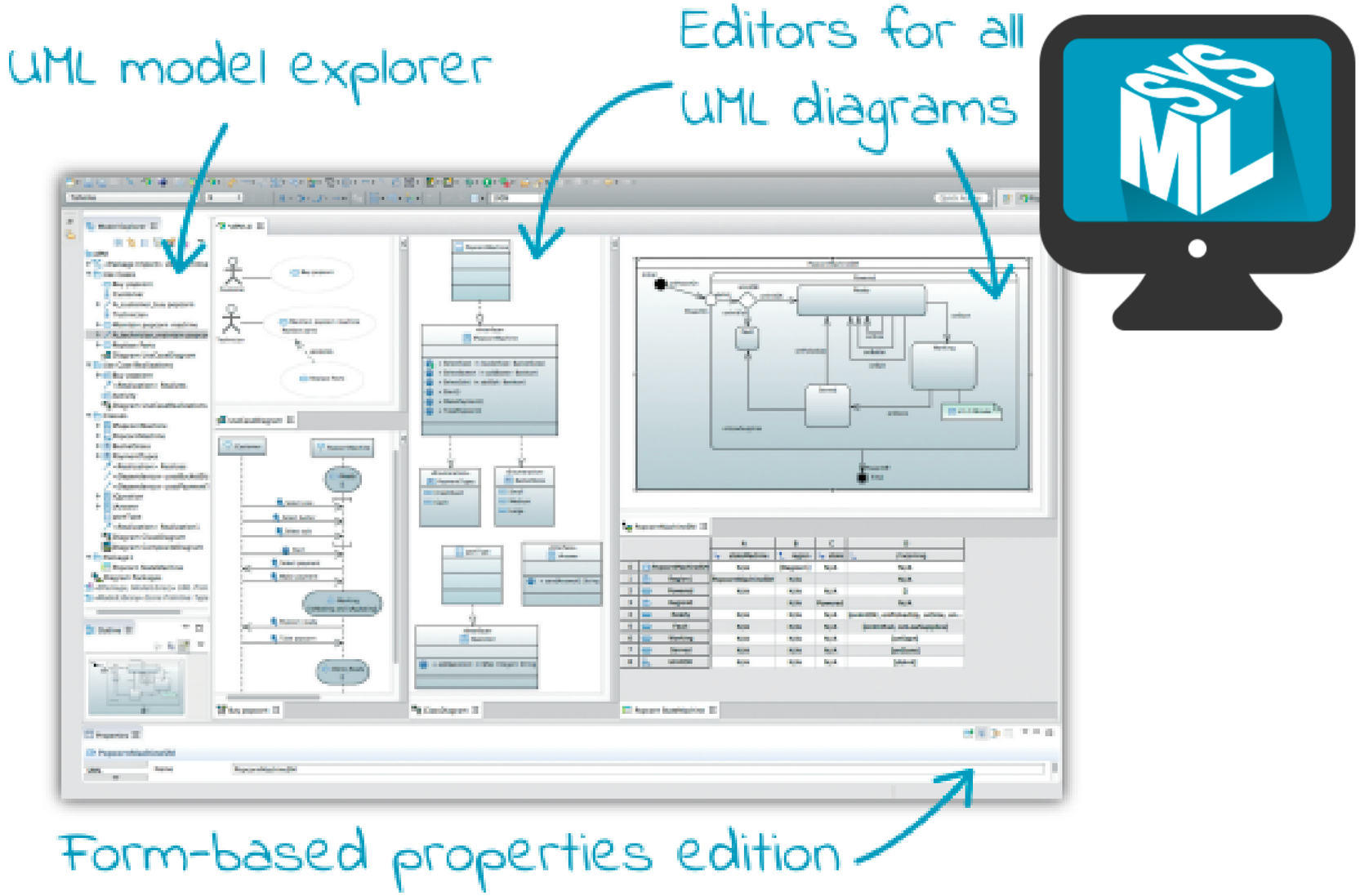
Tools

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

- Capella: https://www.eclipse.org/capella/
- Papyrus: https://www.eclipse.org/papyrus/

- Visio SysML template

Papyrus (SysML)



Official website: https://www.eclipse.org/papyrus/

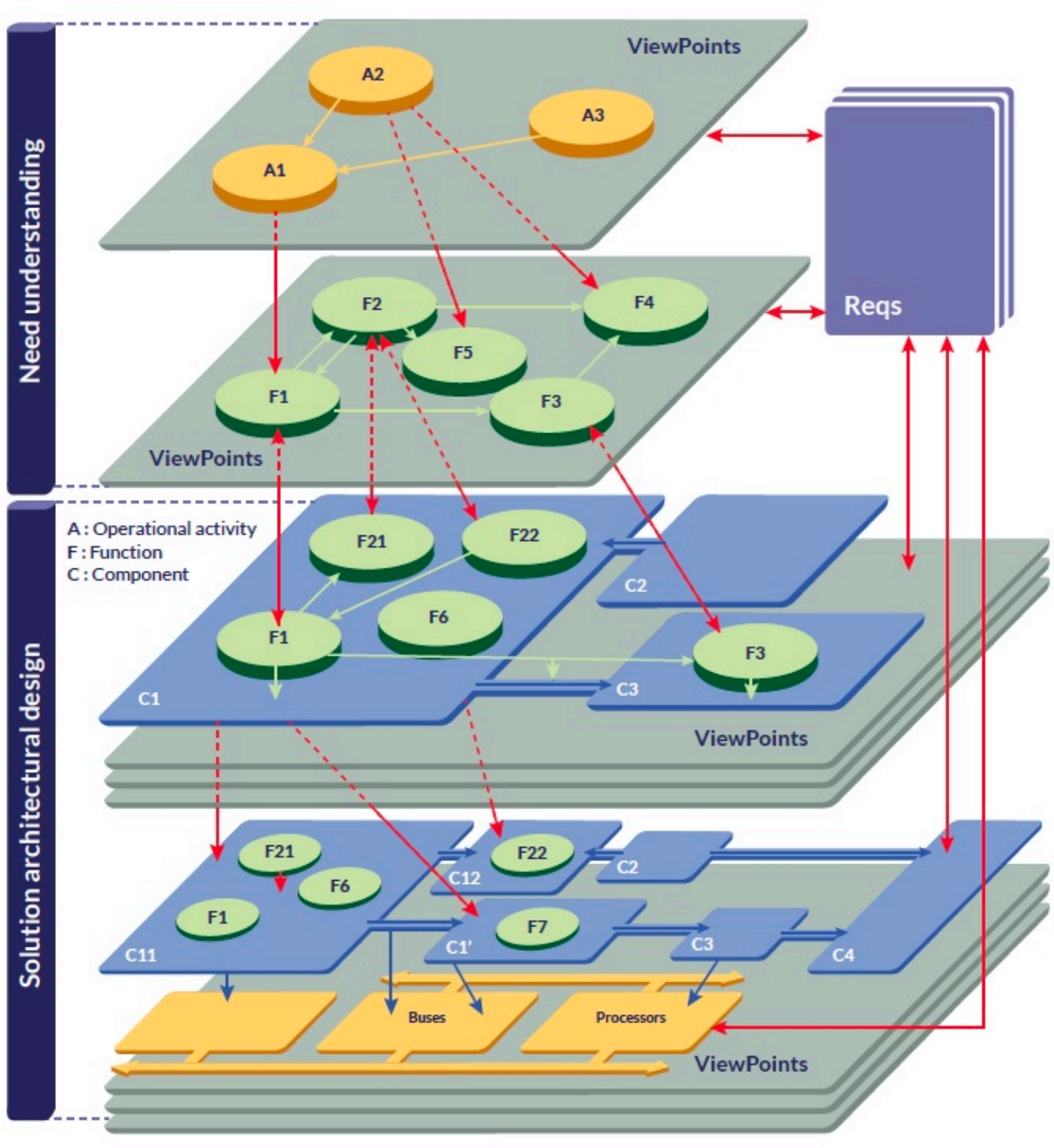
Capella

- An industrial, open-source, systems engineering workbench
  - With its own formalism (~aligned with SysML)
  - With its own method (namely, Arcadia)

- Official website: https://www.eclipse.org/capella/
- The Arcadia method: https://www.eclipse.org/capella/arcadia.html

- Differences with SysML: https://www.eclipse.org/capella/arcadia_capella_sysml_tool.html

Arcadia



Operational Analysis
What the users of the system need to accomplish

Functional & Non Functional Need
What the system has to accomplish for the users

Logical Architecture
How the system will work to fulfill expectations

Physical Architecture
How the system will be developed and built

The Arcadia method: https://www.eclipse.org/capella/arcadia.html

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.): https://fr.scribd.com/document/3103975/sysml-overview-oose