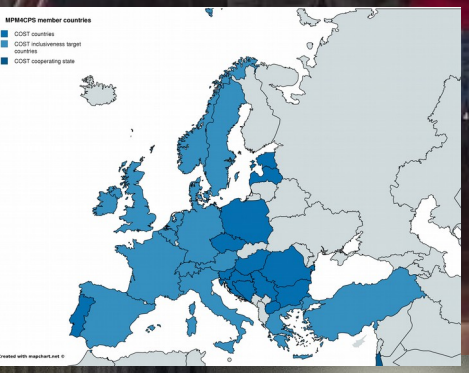
Multi-Paradigm Modelling for Cyber-Physical Systems

AADL meeting

Hans Vangheluwe

http://msdl.cs.mcgill.ca/

27 June 2019



MPM4CPS

www.mpm4cps.eu

cost
EUROPEAN COOPERATION IN SCIENCE AND TECHNOLOGY

Multi-Paradigm Modelling
for Cyber-Physical Systems

Cyber-Physical Systems (CPS)

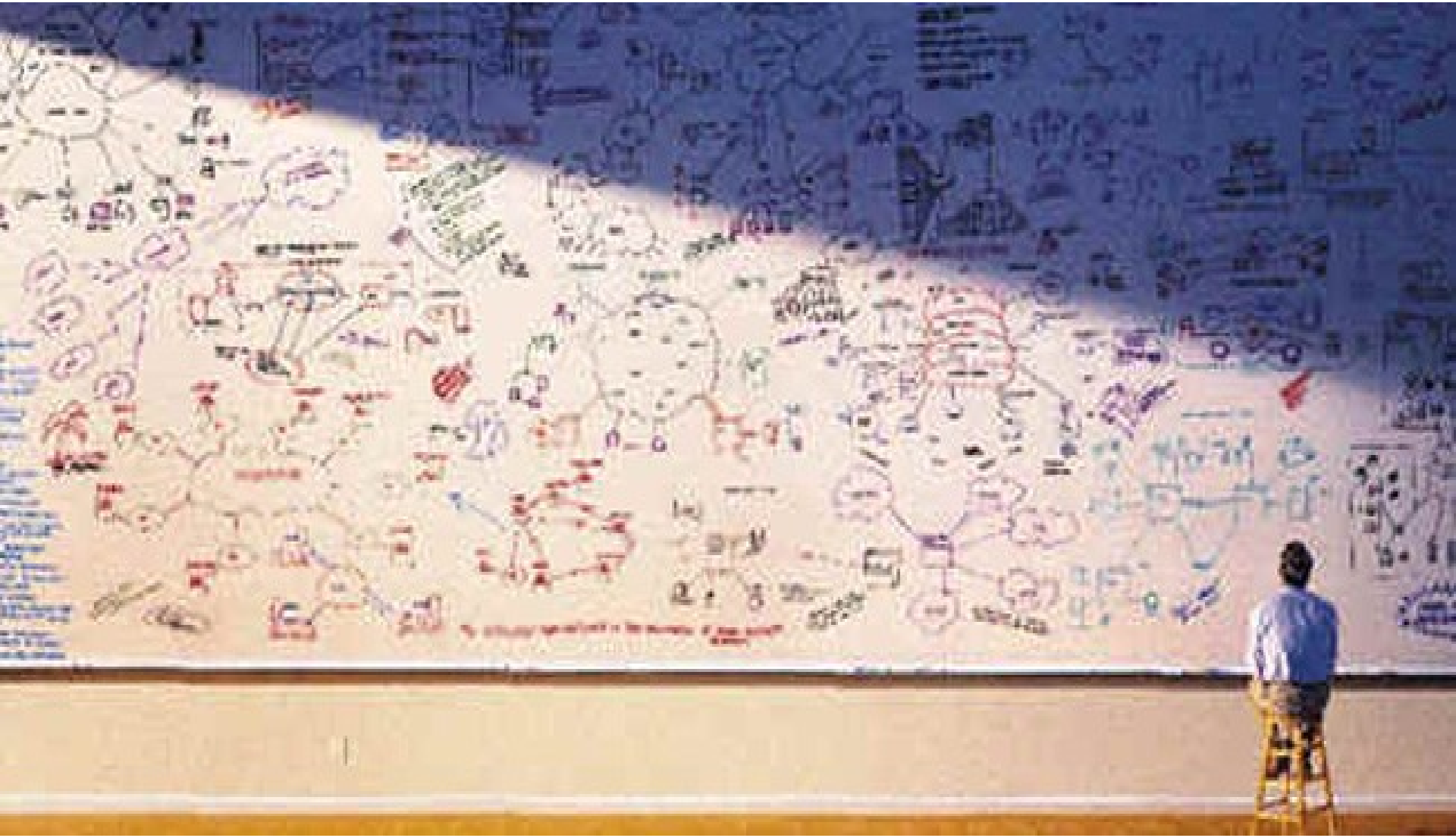
CPS emerge from the **networking** of **multi-physical** (mechanical, electrical, hydraulic, biochemical, ….) and **computational** (control, signal processing, logical inference, planning, …) processes, often interacting with a highly uncertain environment, including human actors, in a socio-economic context.

Multi-Paradigm Modelling
for Cyber-Physical Systems

Cyber-Physical Systems (CPS)

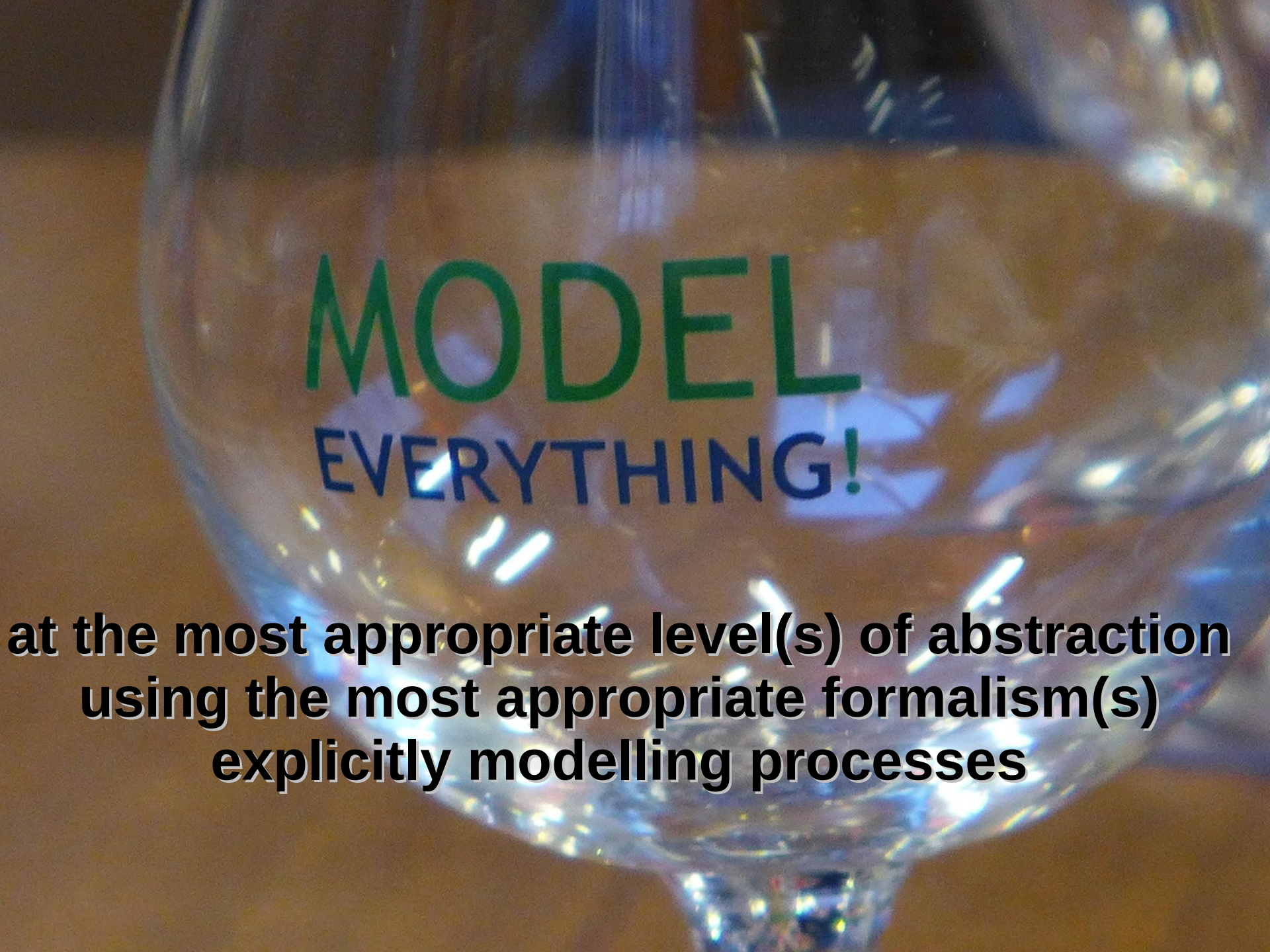
CPS emerge from the **networking** of **multi-physical** (mechanical, electrical, hydraulic, biochemical, ….) and **computational** (control, signal processing, logical inference, planning, …) processes, often interacting with a highly uncertain environment, including human actors, in a socio-economic context.

engineered systems



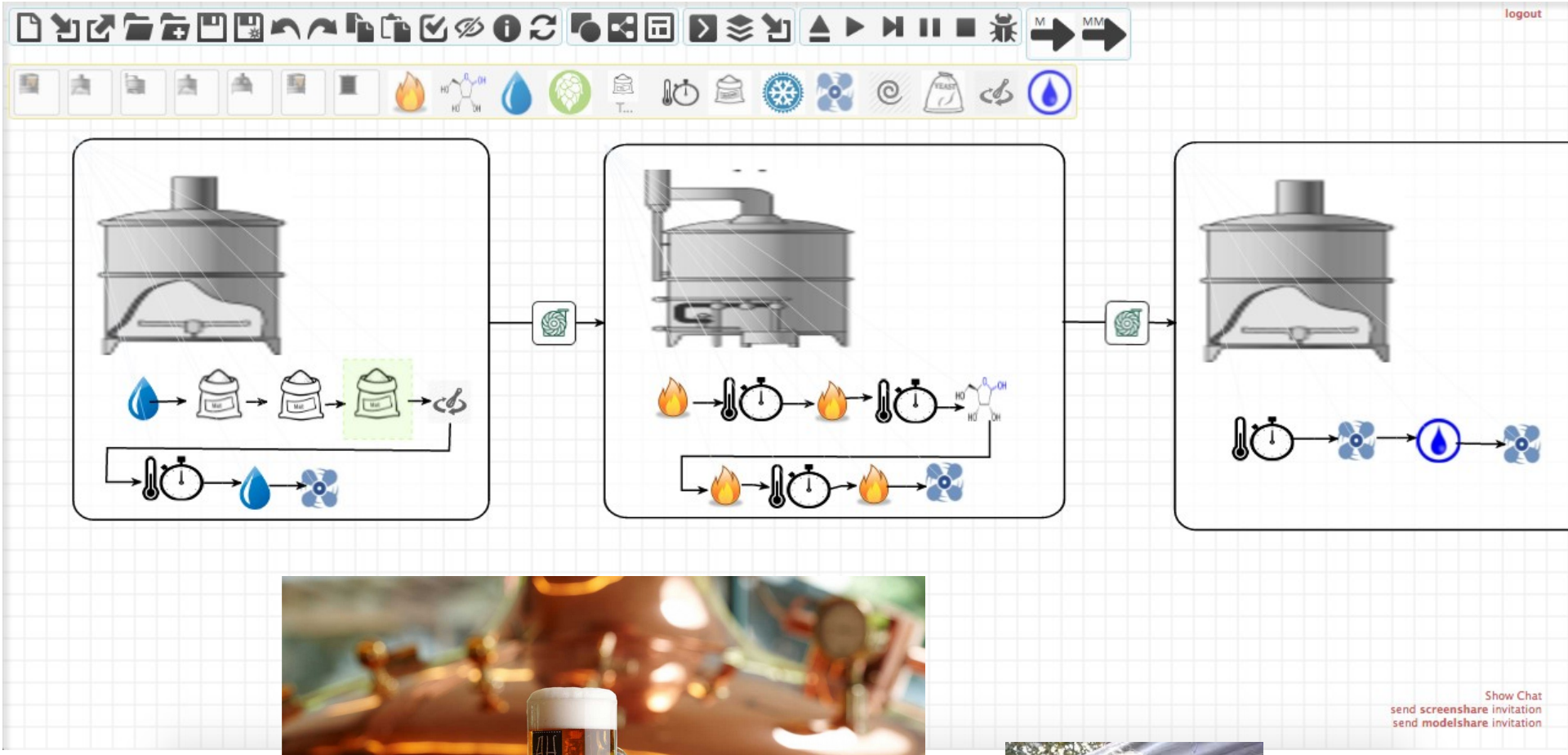
How to deal with **Complexity?**

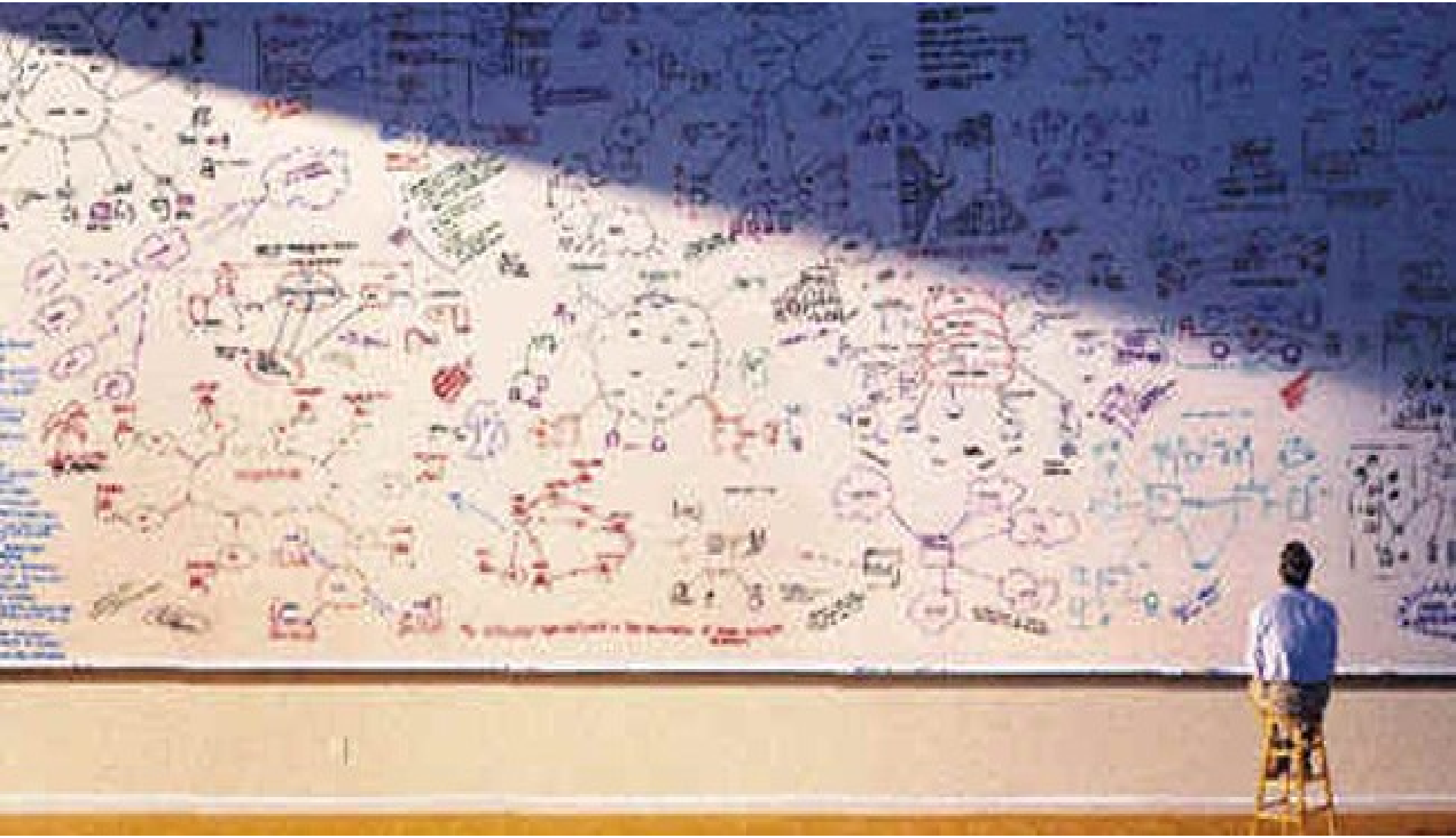
(in engineered systems)



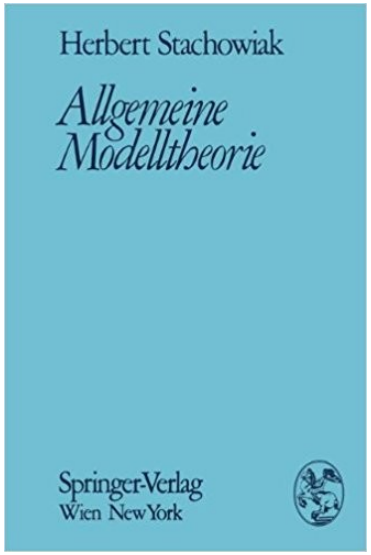
at the most appropriate level(s) of abstraction
using the most appropriate formalism(s)
explicitly modelling processes

MODEL
EVERYTHING!
DSM TP 2014
Theory and Practice
5th International Summer School
on Domain Specific Modeling
Antwerp, Belgium
25 - 29 August
Thomas Kühne


logout
Show Chat
send screenshare invitation
send modelshare invitation

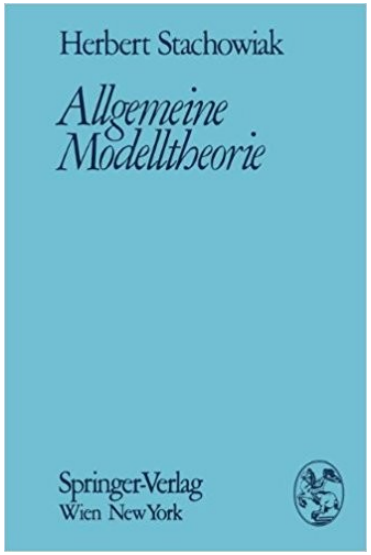


What is a **Model?**



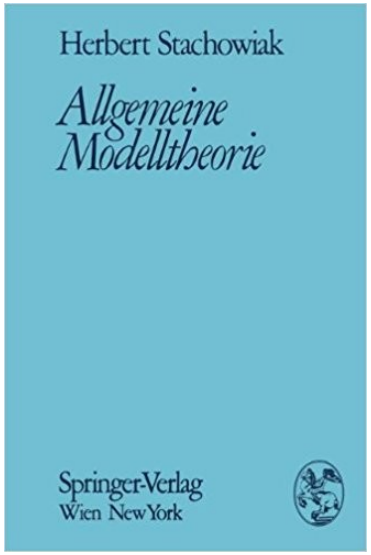
1973

mapping feature	A model is based on an original.[4]
reduction feature	A model only reflects a (relevant) selection of an original's properties.
pragmatic feature	A model needs to be usable in place of an original with respect to some purpose.



1973

mapping feature	A model is based on an original.[4]
reduction feature	A model only reflects a (relevant) selection of an original's properties.
pragmatic feature	A model needs to be usable in place of an original with respect to some purpose.

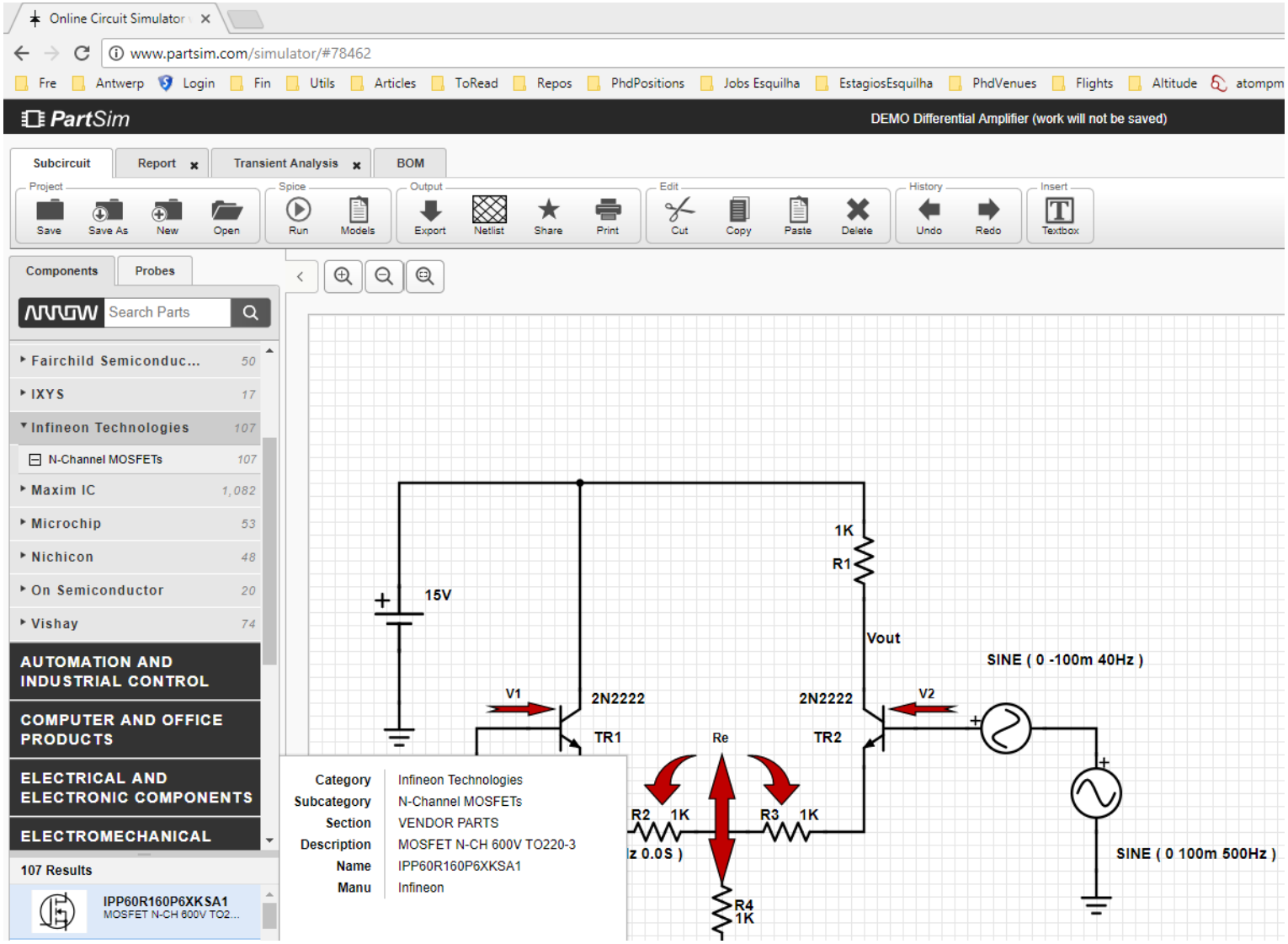


1973

mapping feature	A model is based on an original.[4]
reduction feature	A model only reflects a (relevant) selection of an original's properties.
pragmatic feature	A model needs to be usable in place of an original with respect to some purpose.

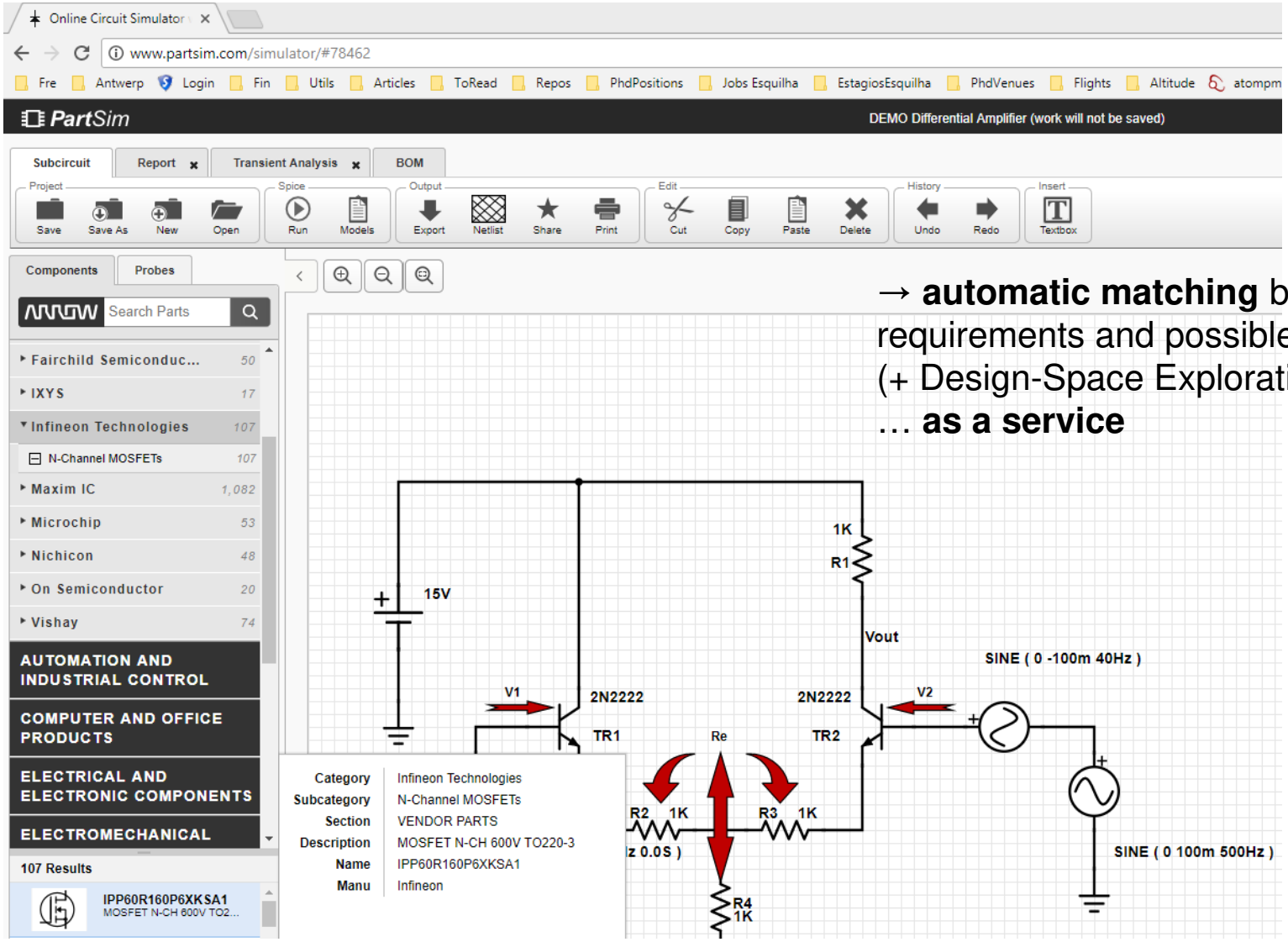
Virtual Build

http://www.partsim.com/

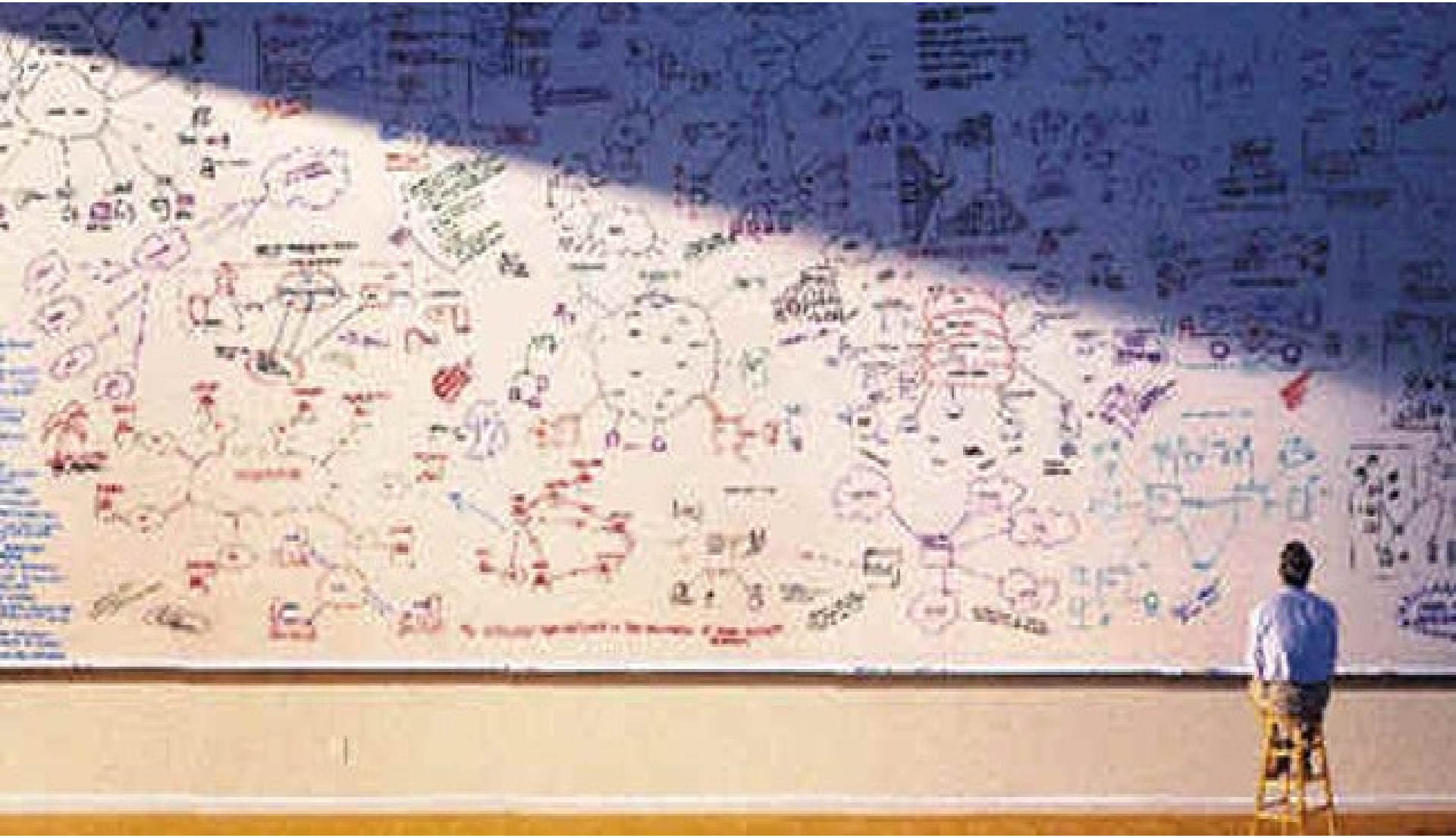


Virtual Build

http://www.partsim.com/

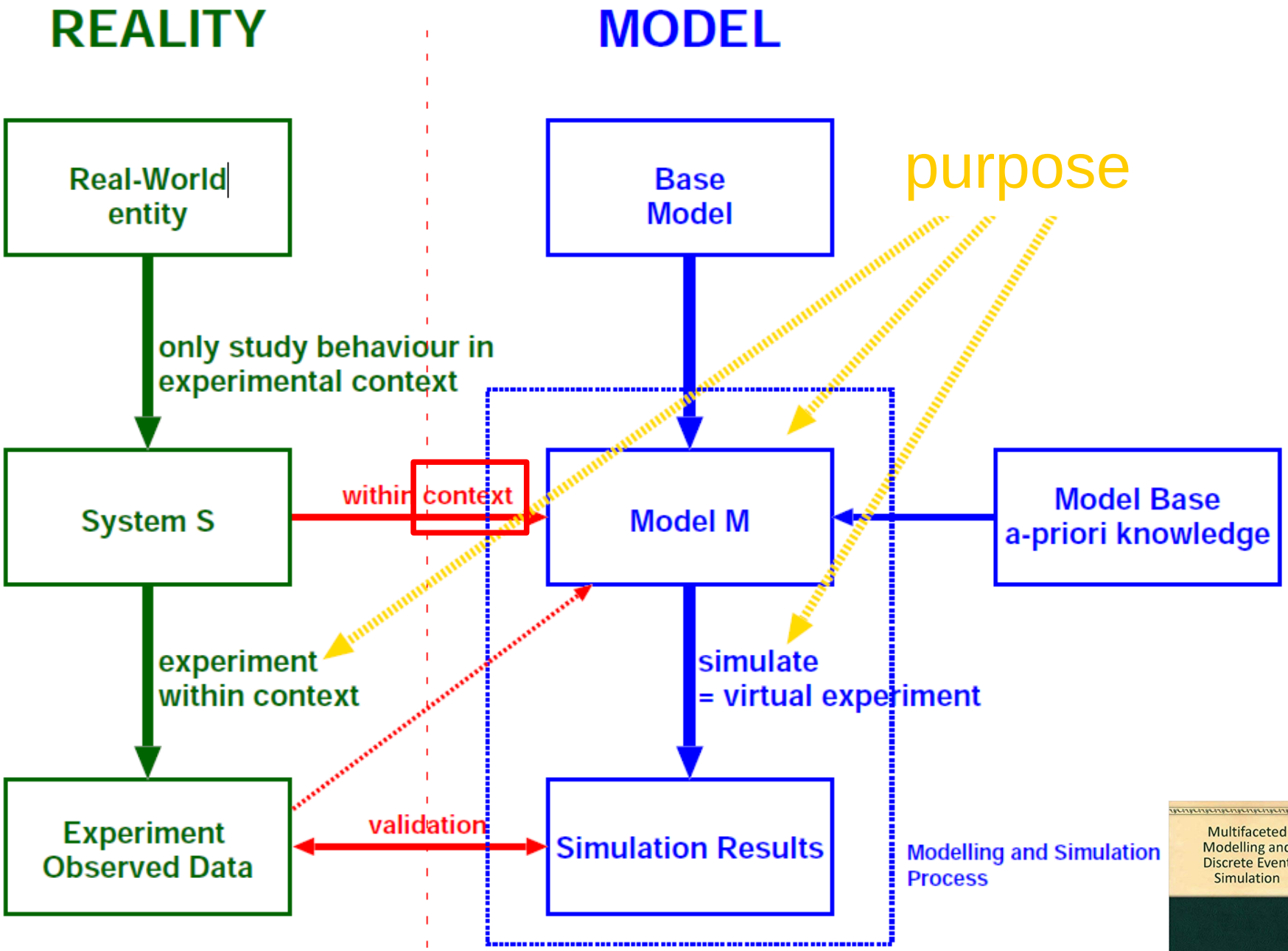


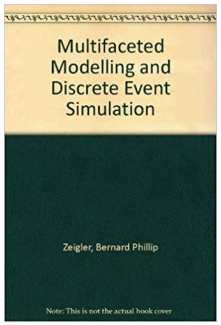
→ **automatic matching** between requirements and possible designs (+ Design-Space Exploration) … **as a service**



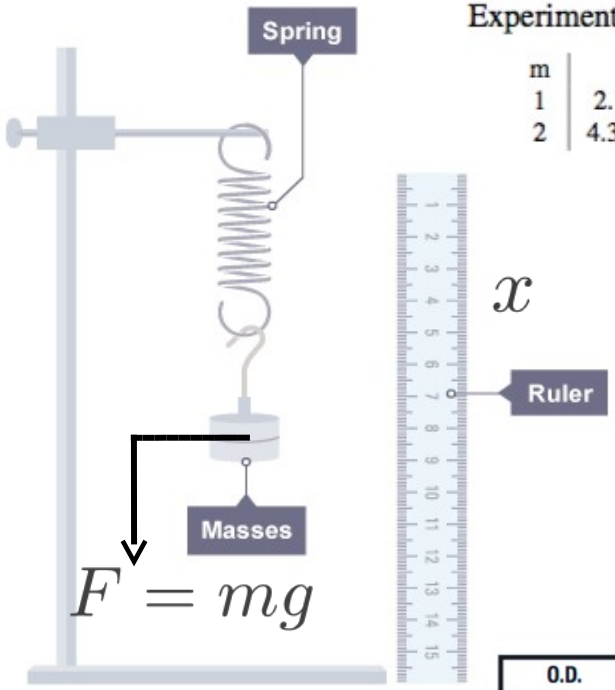
How to deal with **Complexity?**

(in engineered systems)



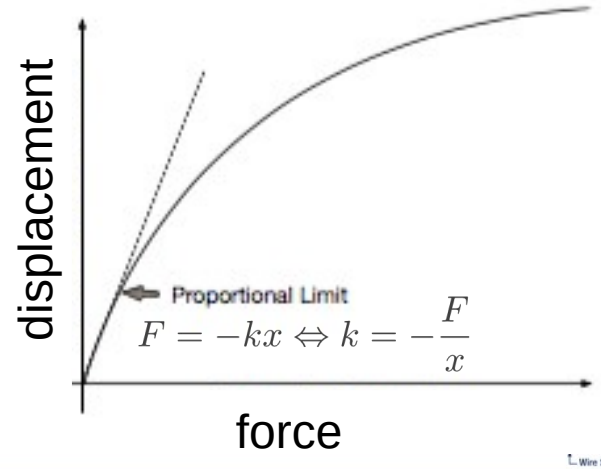


Bernard P. Zeigler. *Multi-faceted Modelling and Discrete-Event Simulation*. Academic Press, 1984.



Experimental spring results, with mass m in kg and displacement x (± 0.0001) in cm

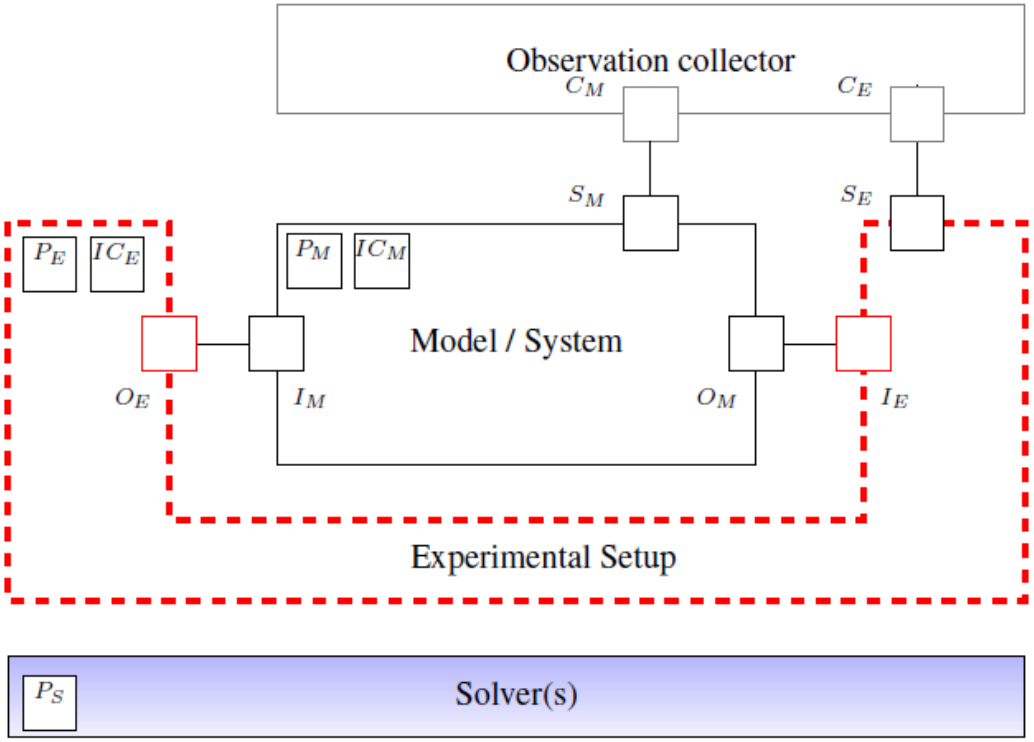
m	x	m	x	m	x	m	x	m	x
1	2.100	3	6.3749	5	10.4915	7	14.6081	9	19.0012
2	4.3166	4	8.4332	6	12.5489	8	16.7774		

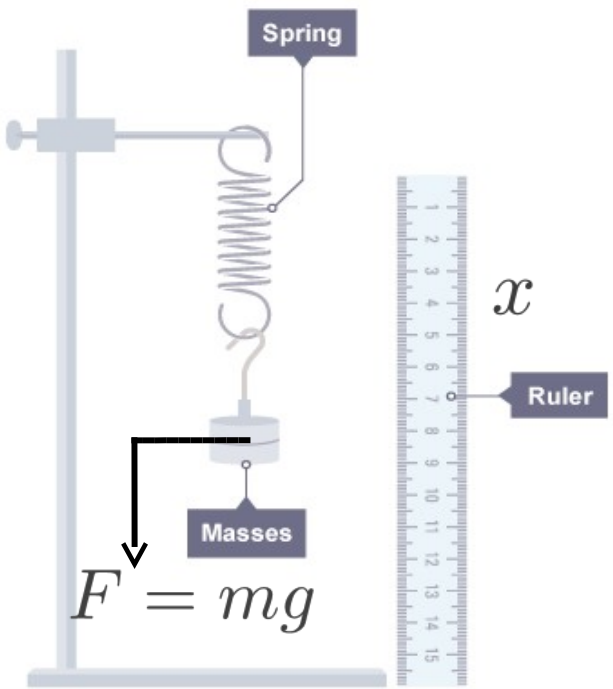


O.D. Inches	O.D. mm	CENTURY STOCK NUMBER	FREE LENGTH Inches	FREE LENGTH mm	I.D. Inches	I.D. mm	RATE Lbs./In.	RATE N/mm	SUGG. MAX. DEFL. Inches	SUGG. MAX. DEFL. mm	SUGG. MAX. LOAD Lbs.	SUGG. MAX. LOAD N	SOLID LENGTH Inches	SOLID LENGTH mm	WIRE DIA. Inches	WIRE DIA. mm	TOTAL COILS	MAT'L	ENDS	FINISH
0.036	.91	10075	.59	15.1	.022	.6	2.6	.46	.15	3.8	.39	1.7	.35	8.9	0.007	0.2	49.0	SST	C	N
0.036	.91	JJ-7	.63	15.9	.024	.6	1.6	.28	.16	4.1	.25	1.1	.25	6.2	0.006	0.2	40.0	SST	C	N
0.040	1.02	2924	.66	16.8	.020	.5	11	2.0	.13	3.2	1.4	6.4	.50	12.6	0.010	0.3	48.5	MW	C	N
0.040	1.02	10778	.69	17.5	.028	.7	1.0	.17	.35	8.9	.35	1.6	.30	7.7	0.006	0.2	49.5	MW	C	N
0.054	1.37	RR-6	.25	6.4	.036	.9	6.2	1.1	.09	2.2	.56	2.5	.16	4.1	0.009	0.2	16.5	SST	C	N
0.054	1.37	10619	.72	18.3	.038	1.0	1.6	.29	.37	9.3	.60	2.7	.32	8.1	0.008	0.2	39.0	MW	C	N
0.057	1.45	70000	.13	3.3	.045	1.1	3.7	.66	.07	1.7	.25	1.1	.04	1.0	0.006	0.2	5.75	MW	C	N
0.057	1.45	70000S	.13	3.3	.045	1.1	3.3	.57	.05	1.3	.17	.74	.04	1.0	0.006	0.2	5.75	SST	C	N
0.057	1.45	70009	.13	3.3	.043	1.1	6.9	1.2	.06	1.5	.40	1.8	.05	1.2	0.007	0.2	6.00	MW	C	N
0.057	1.45	70009S	.13	3.3	.043	1.1	6.0	1.1	.04	1.1	.26	1.2	.05	1.2	0.007	0.2	6.00	SST	C	N
0.057	1.45	70018	.13	3.3	.041	1.0	12	2.1	.05	1.2	.57	2.5	.06	1.4	0.008	0.2	6.13	MW	C	N
0.057	1.45	70018S	.13	3.3	.041	1.0	11	1.8	.03	.88	.37	1.6	.06	1.4	0.008	0.2	6.13	SST	C	N
0.057	1.45	70001	.19	4.8	.045	1.1	2.3	.40	.11	2.8	.25	1.1	.06	1.4	0.006	0.2	8.13	MW	C	N
0.057	1.45	70001S	.19	4.8	.045	1.1	2.0	.35	.08	2.1	.17	.74	.06	1.4	0.006	0.2	8.13	SST	C	N
0.057	1.45	70010	.19	4.8	.043	1.1	4.0	.70	.10	2.5	.40	1.8	.07	1.8	0.007	0.2	8.88	MW	C	N
0.057	1.45	70010S	.19	4.8	.043	1.1	3.5	.61	.07	1.9	.26	1.2	.07	1.8	0.007	0.2	8.88	SST	C	N
0.057	1.45	70019	.19	4.8	.041	1.0	7.4	1.3	.08	2.0	.57	2.5	.08	2.0	0.008	0.2	8.75	MW	C	N
0.057	1.45	70019S	.19	4.8	.041	1.0	6.4	1.1	.06	1.4	.37	1.6	.08	2.0	0.008	0.2	8.75	SST	C	N
0.057	1.45	70002	.25	6.4	.045	1.1	1.7	.30	.15	3.8	.25	1.1	.07	1.7	0.006	0.2	10.3	MW	C	N
0.057	1.45	70002S	.25	6.4	.045	1.1	1.5	.26	.11	2.8	.17	.74	.07	1.7	0.006	0.2	10.3	SST	C	N
0.057	1.45	70011	.25	6.4	.043	1.1	3.1	.54	.13	3.3	.40	1.8	.08	2.1	0.007	0.2	11.0	MW	C	N
0.057	1.45	70011S	.25	6.4	.043	1.1	2.7	.47	.10	2.5	.26	1.2	.08	2.1	0.007	0.2	11.0	SST	C	N
0.057	1.45	70020	.25	6.4	.041	1.0	5.3	.92	.11	2.8	.57	2.5	.10	2.5	0.008	0.2	11.5	MW	C	N
0.057	1.45	70020S	.25	6.4	.041	1.0	4.6	.80	.08	2.0	.37	1.6	.10	2.5	0.008	0.2	11.5	SST	C	N
0.057	1.45	70003	.31	7.9	.045	1.1	1.4	.24	.19	4.7	.25	1.1	.08	2.0	0.006	0.2	12.4	MW	C	N
0.057	1.45	70003S	.31	7.9	.045	1.1	1.2	.21	.14	3.6	.17	.74	.08	2.0	0.006	0.2	12.4	SST	C	N
0.057	1.45	70012	.31	7.9	.043	1.1	2.4	.42	.17	4.2	.40	1.8	.10	2.6	0.007	0.2	13.5	MW	C	N
0.057	1.45	70012S	.31	7.9	.043	1.1	2.1	.37	.12	3.2	.26	1.2	.10	2.6	0.007	0.2	13.5	SST	C	N
0.057	1.45	70021	.31	7.9	.041	1.0	4.1	.72	.14	3.6	.57	2.5	.12	3.1	0.008	0.2	14.3	MW	C	N
0.057	1.45	70021S	.31	7.9	.041	1.0	3.6	.63	.10	2.6	.37	1.6	.12	3.1	0.008	0.2	14.3	SST	C	N

www.centuryspring.com

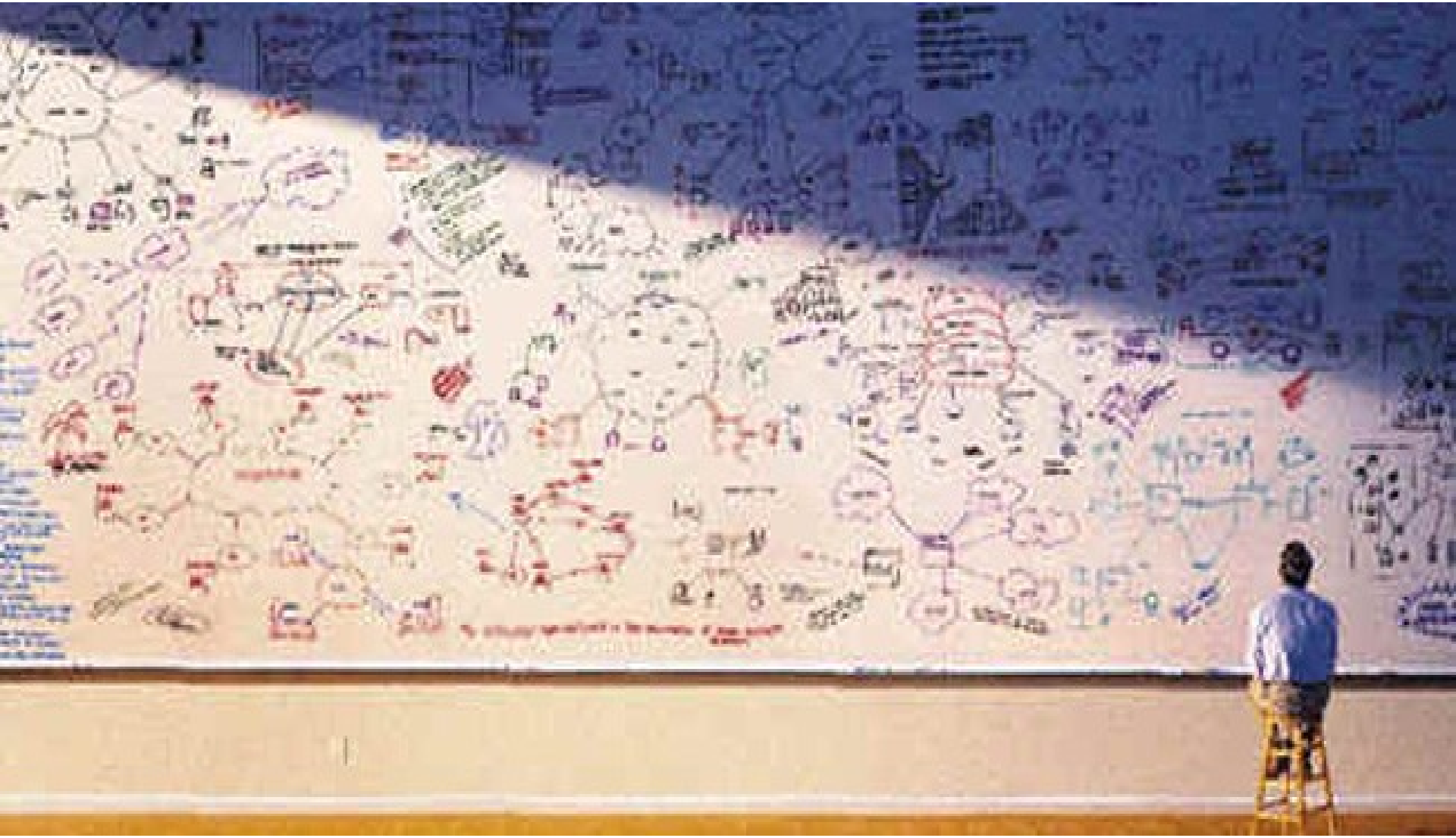
Validity **“Frame”** ~ reproducibility of experiments





Denil, J., Klikovits, S., Mosterman, P. J., Vallecillo, A., & Vangheluwe, H. (2017).
The experiment model and validity frame in M&S.
In *Proceedings of the Symposium on Theory of Modeling & Simulation* (Vol. 49).

Vanherpen, K., Denil, J., De Meulenaere, P., & Vangheluwe, H. (2016).
Ontological Reasoning as an Enabler of Contract-Based Co-design.
In C. Berger, M. R. Mousavi, & R. Wisniewski (Eds.), *Cyber Physical Systems. Design, Modeling, and Evaluation: 6th International Workshop, CyPhy 2016, Pittsburgh, PA, USA, October 6, 2016, Revised Selected Papers* (pp. 101–115). Cham: Springer International Publishing.
http://doi.org/10.1007/978-3-319-51738-4_8

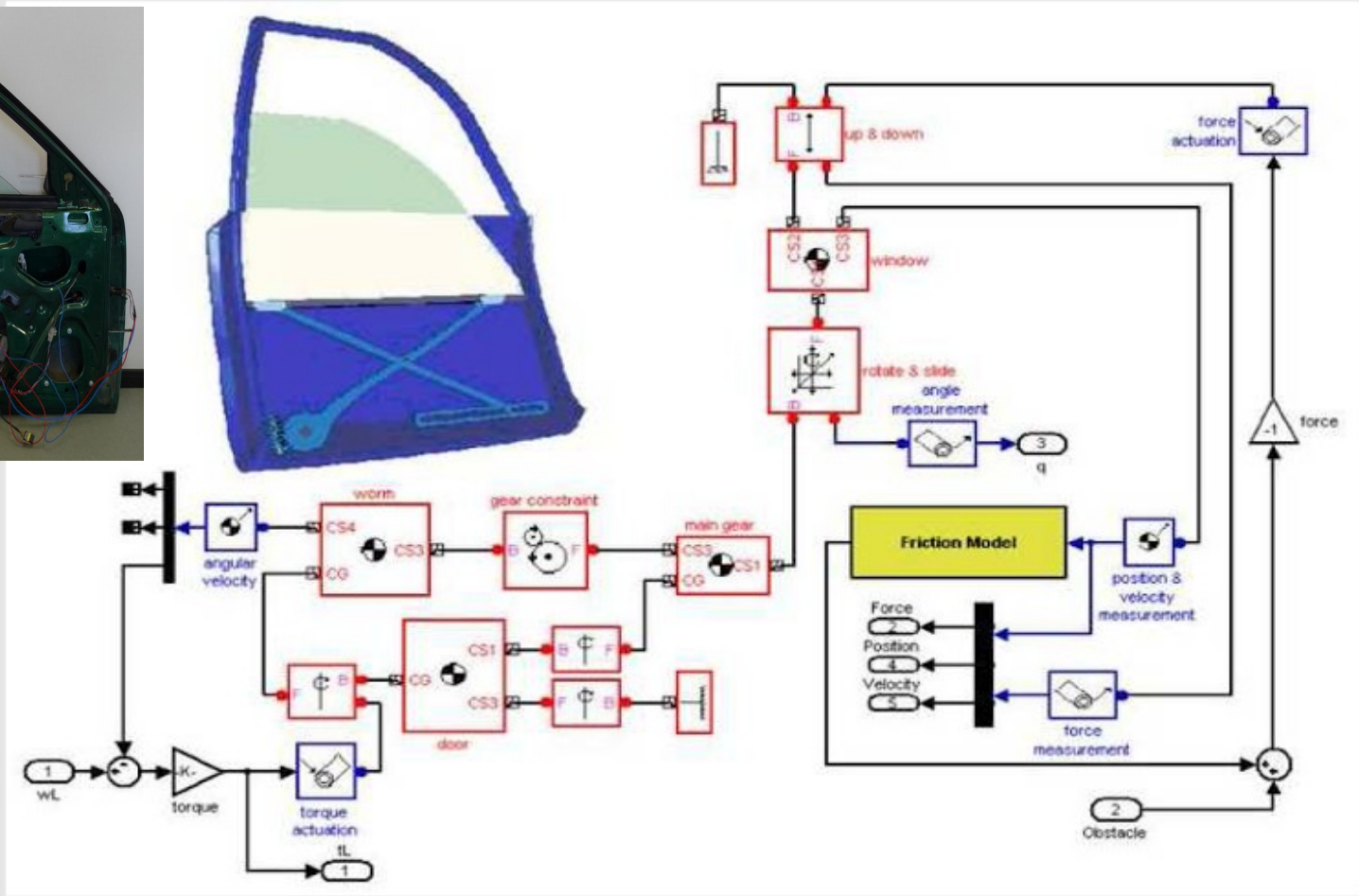


How to deal with **Complexity?**

(in engineered systems)

Most Appropriate Formalism

Mechanics subsystem

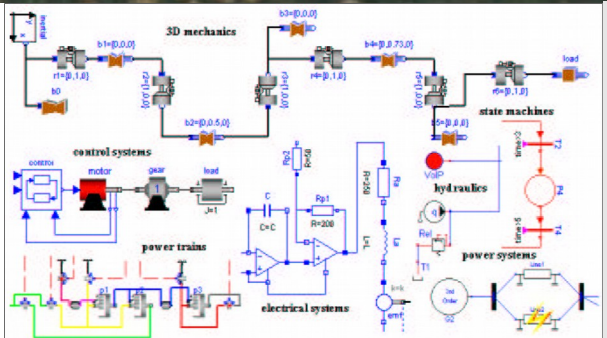


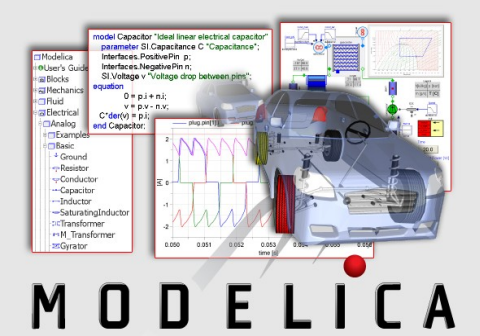
Simulation in Europe

SiE

ESPRIT Basic Research Working Group 8467
Simulation for the Future: New Concepts, Tools and Applications

Keywords:
simulation technologies, multi-paradigm modelling, solvers, standards, interoperability, industrial deployment, demonstrators, user-simulator interfaces

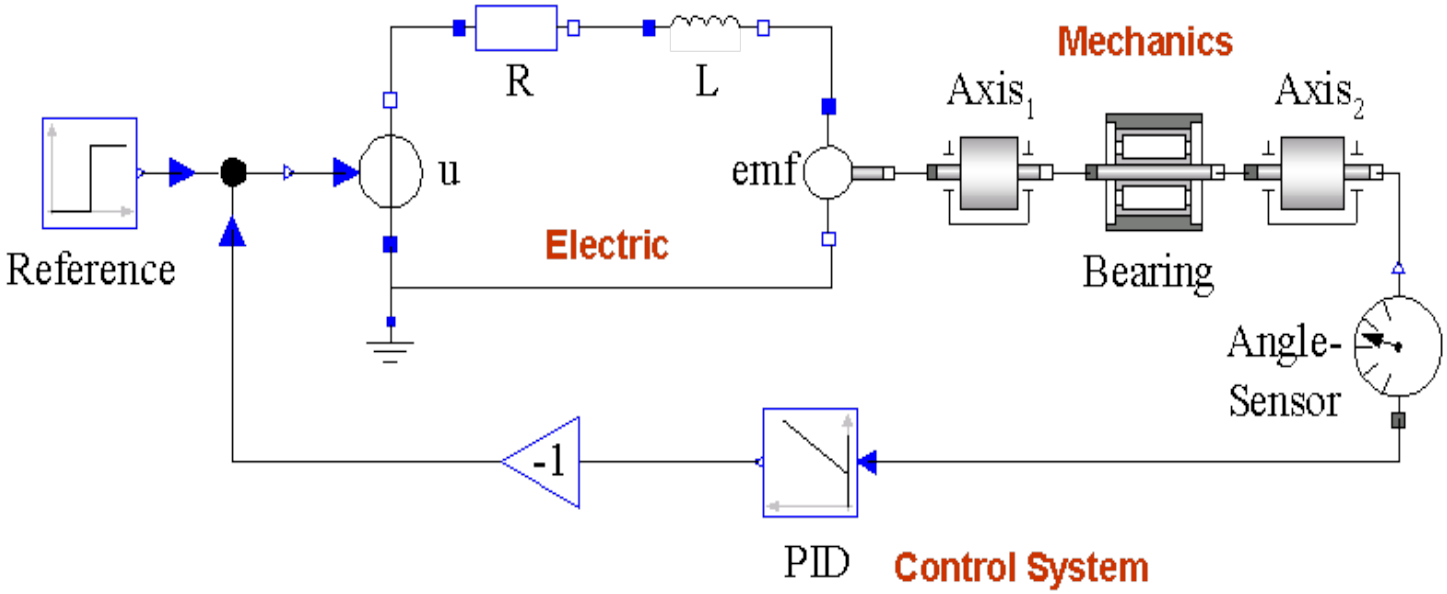




Multi-Domain Modeling

```
model Resistor "Ideal linear electrical resistor"
  extends Interfaces.OnePort;
  parameter SI.Resistance R=1 "Resistance";
equation
  R*i = v;
end Resistor;
```

http://www.modelica.org



http://www.eoolt.org

common underlying semantic domain: hybrid differential Algebraic Equations ++

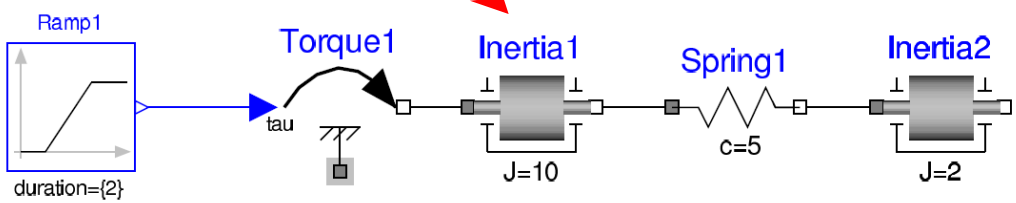
this slide from Peter Fritzson's Modelica tutorial

Multi-Domain Modeling

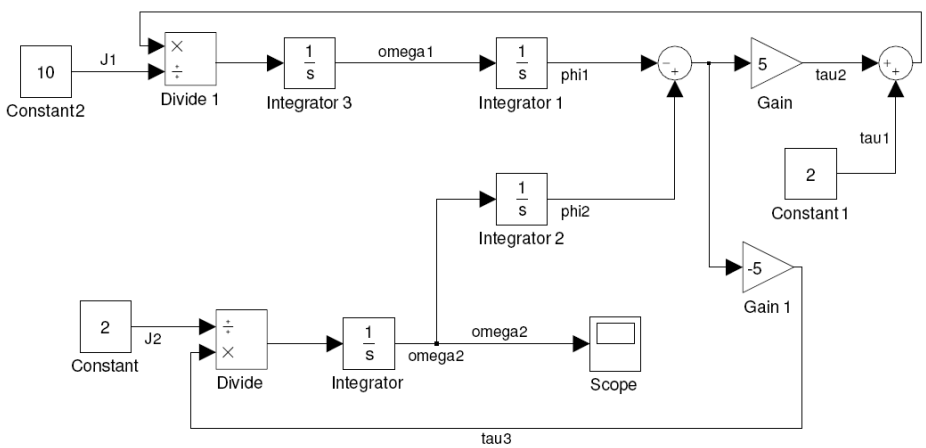
Visual Acausal Hierarchical Component Modeling

Keeps the physical structure

Acausal model (Modelica)



Causal block-based model (Simulink)



this slide from Peter Fritzson's Modelica tutorial

UNIVERSITÉ DE GENÈVE
Centre Universitaire d'Informatique

FACULTÉ DES SCIENCES
Professeur D. Buchs, directeur
Professeur G. Falquet, codirecteur

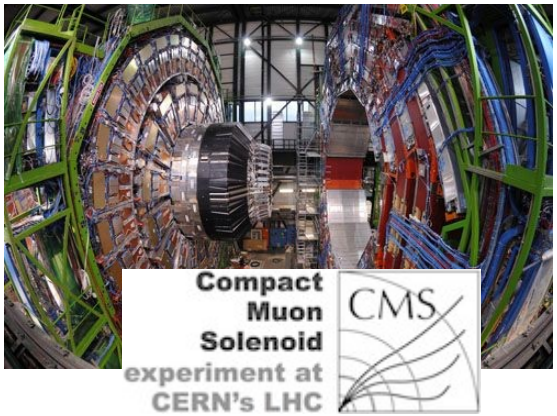
A Methodology For The Development Of Complex Domain Specific Languages

THÈSE

présentée à la Faculté des sciences de l'Université de Genève
pour obtenir le grade de Docteur ès sciences, mention informatique

2010

Matteo Risoldi



CMS Tracker Cosmic Rack

Design

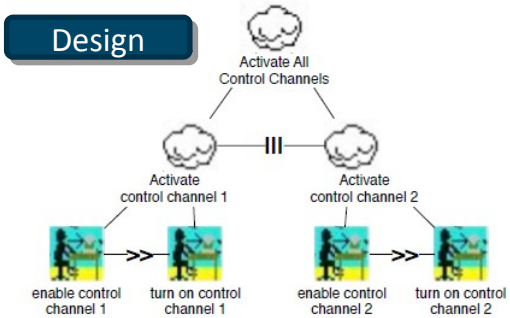


Figure 4.8. CTT for the turn on control channels *task*

Property

```
*PG-Layer-4-Rod-2.apnmm_diagram    properties.prop
import 'PG-Layer-4-Rod-2.apnmm'
import 'blackToken.adt'

Expressions

  MUTUAL_EXCLUSION : (((card($on in ON) + card($onlv in ONLV)) + card($off in OFF
  NOSTATE : (((card($on in ON) + card($onlv in ONLV)) + card($off in OFF)) + card

   TEMP : card($tmp in temp)=1;
   TEMP1 : card($tmp in temp1)=1;
   TEMP2 : card($tmp in temp2)=1;
   TEMP3 : card($tmp in temp3)=1;
   TEMP4 : card($tmp in temp4)=1;
   TEMP5 : card($tmp in temp5)=1;
   TEMP6 : card($tmp in temp6)=1;
   TEMP7 : card($tmp in temp7)=1;
   TEMP8 : card($tmp in temp8)=1;
   TEMP9 : card($tmp in temp9)=1;
   TEMP10 : card($tmp in temp10)=1;
   INTERMEDIATE_STATE : (((((((((@TEMP | @TEMP1) | @TEMP2) | @TEMP3) | @TEMP4) |

Check
  (!(@INTERMEDIATE_STATE) => @MUTUAL_EXCLUSION);
```

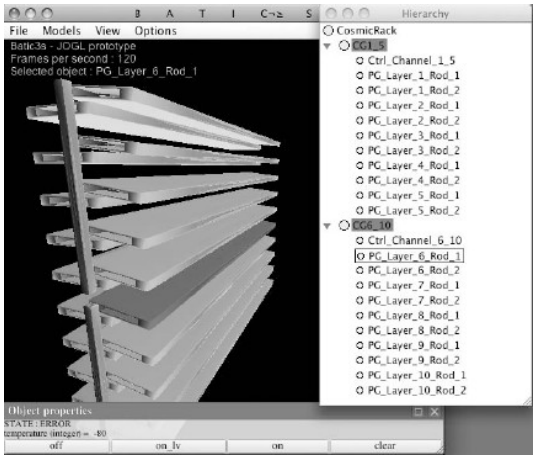
Properties | Specification Imports | Variables | Console | Problems

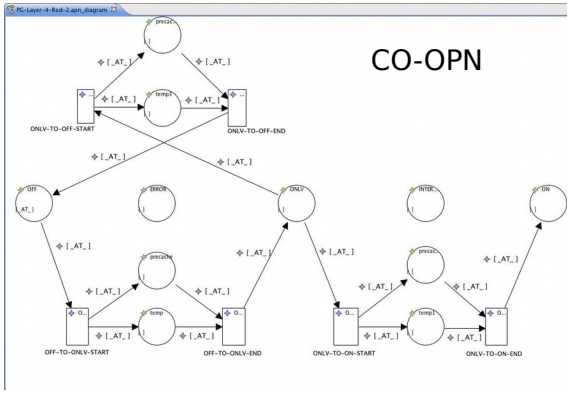
```
AlPiNA Model Checker Engine; [Java Application] /System/Library/Frameworks/Java
*****************************************************************
Compute State Space...
Reachability Time : 8 ms
State Space has been fully generated.
*****************************************************************
Check the properties...
Check property : [(!((((((((((((Card(tmp in temp:TRUE) EQUALS 1) or (Card(tmp in
 temp1:TRUE) EQUALS 1)) or (Card(tmp in temp2:TRUE) EQUALS 1)) or (Card(tmp in t
emp3:TRUE) EQUALS 1)) or (Card(tmp in temp4:TRUE) EQUALS 1)) or (Card(tmp in tem
p5:TRUE) EQUALS 1)) or (Card(tmp in temp6:TRUE) EQUALS 1)) or (Card(tmp in temp7
:TRUE) EQUALS 1)) or (Card(tmp in temp8:TRUE) EQUALS 1)) or (Card(tmp in temp9:T
RUE) EQUALS 1)) or (Card(tmp in temp10:TRUE) EQUALS 1))) implies (((((Card(on in
 ON:TRUE) plus Card(onlv in ONLV:TRUE)) plus Card(off in OFF:TRUE)) plus Card(er
ror in ERROR:TRUE)) plus Card(int in INTERLOCKED:TRUE)) EQUALS 1))]
Property holds : OK
*****************************************************************
Property Check is finished.
```

Alpina

Application

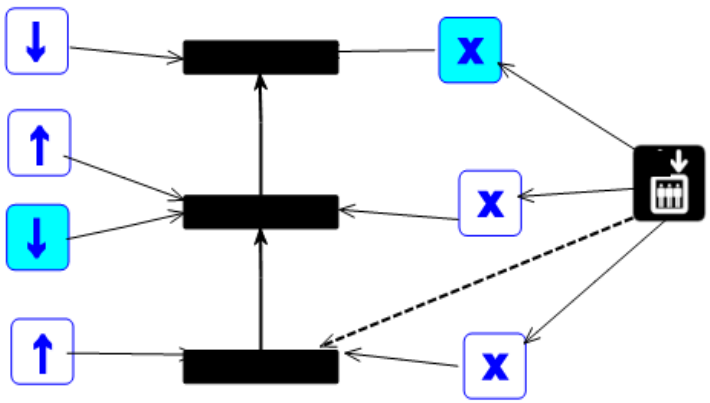
UI prototype





Designing Requirements/Property Languages

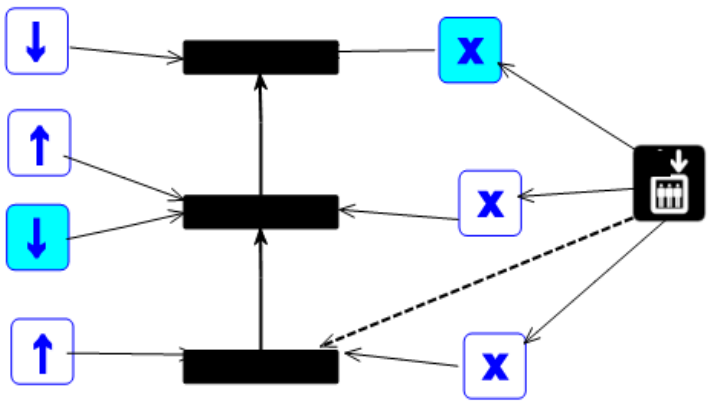
Design $\models$ Property

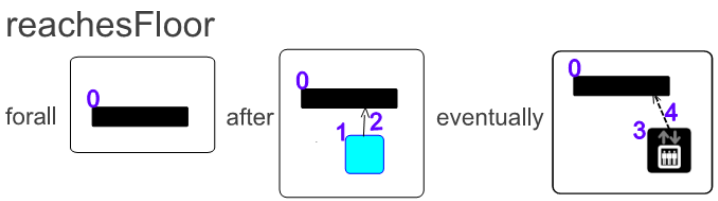


$$\Box(((go0 \wedge up0) \vee \Diamond(floor0 \vee idle)) \rightarrow ((\neg(floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge (((floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge ((\neg(floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge (((floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge (\neg(floor0)\mathcal{U}(floor0 \vee idle)))))))))) \vee \Box(((go1 \wedge up1 \wedge down1) \vee \Diamond(floor1 \vee idle)) \rightarrow ((\neg(floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge (((floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge ((\neg(floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge (((floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge (\neg(floor1)\mathcal{U}(floor1 \vee idle)))))))))) \vee \Box(((go2 \wedge down2) \vee \Diamond(floor2 \vee idle)) \rightarrow ((\neg(floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge (((floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge ((\neg(floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge (((floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge (\neg(floor2)\mathcal{U}(floor2 \vee idle)))))))))))$$

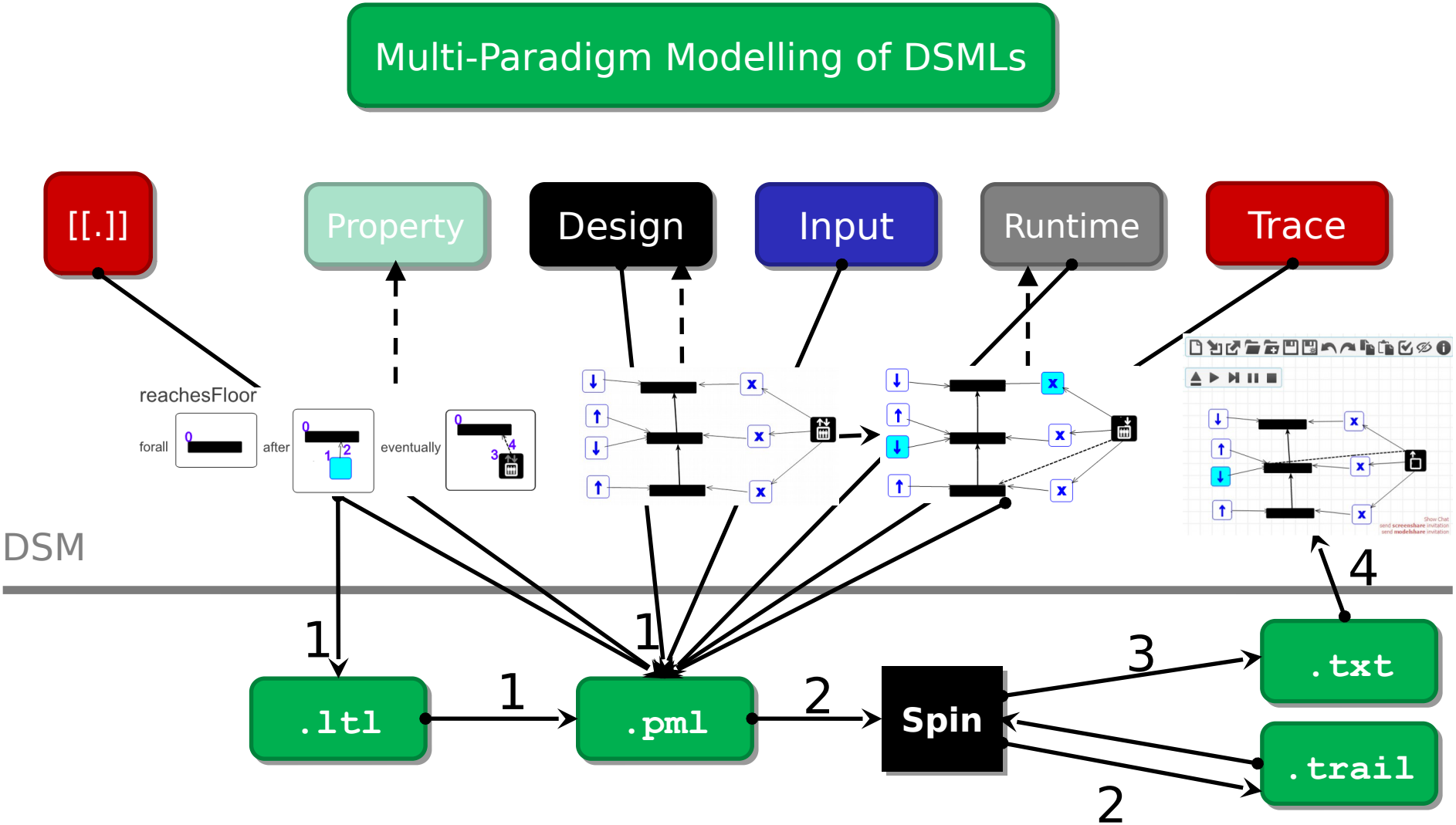
Designing Requirements/Property Languages

Design ⊨ Property



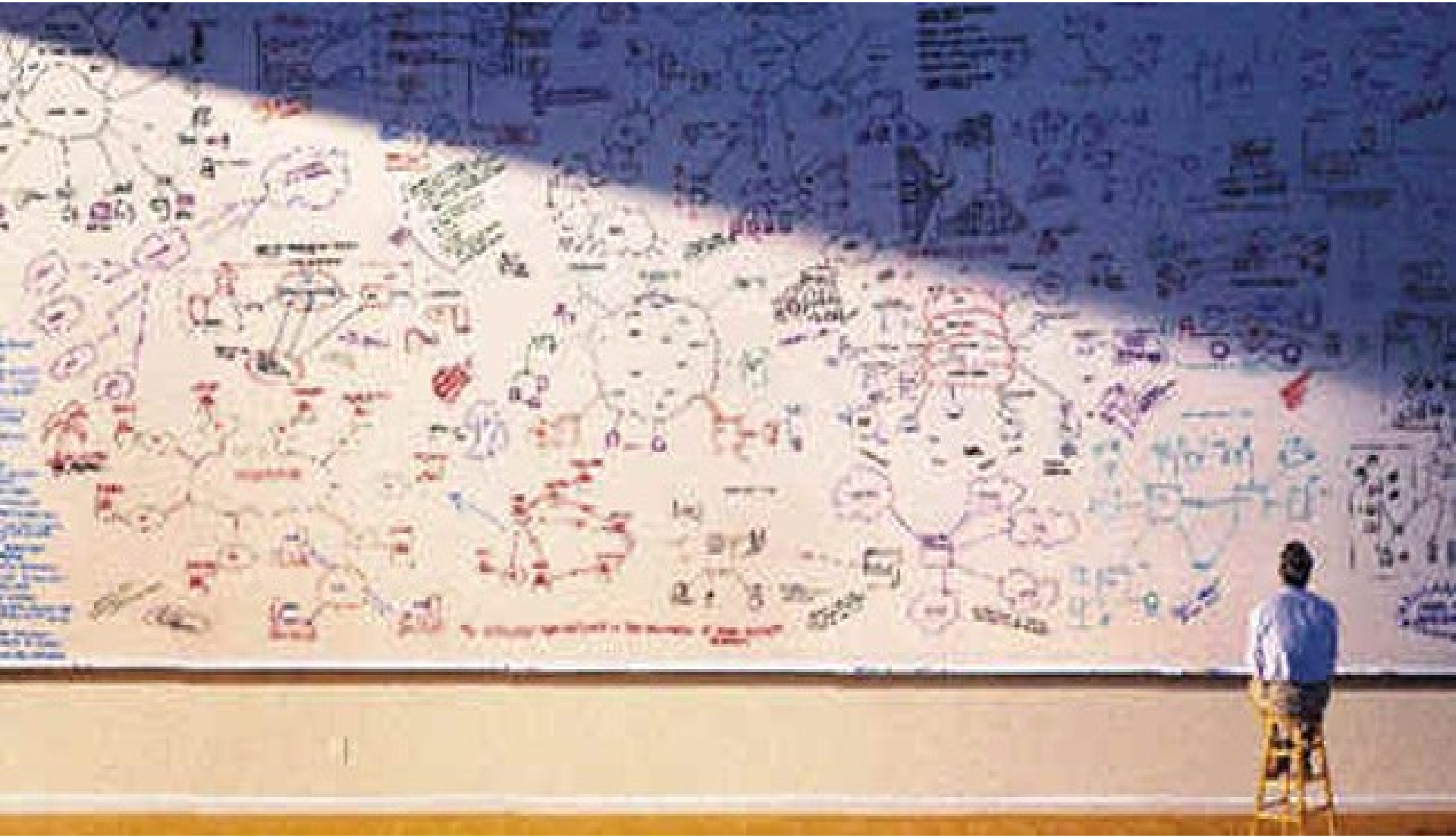


Designing DS Requirements/Property Languages



B. Meyers, R. Deshayes, L. Lucio, E. Syriani, H. Vangheluwe, and M. Wimmer. ProMoBox: A Framework for Generating Domain-Specic Property Languages. In Software Language Engineering (SLE), Vasteras, Sweden, LNCS vol. 8706, pp. 1- 20. Springer. September 2014.

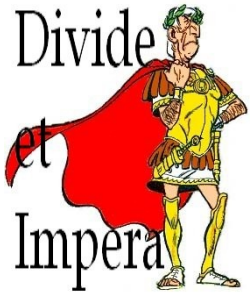
B. Meyers, H. Vangheluwe, J. Denil and R. Salay, "A Framework for Temporal Verification Support in Domain-Specific Modelling," in IEEE Transactions on Software Engineering. doi:10.1109/TSE.2018.2859946

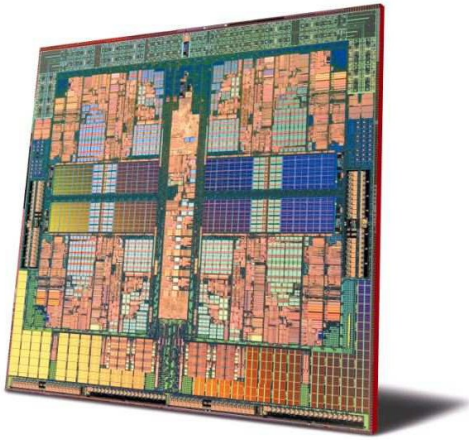


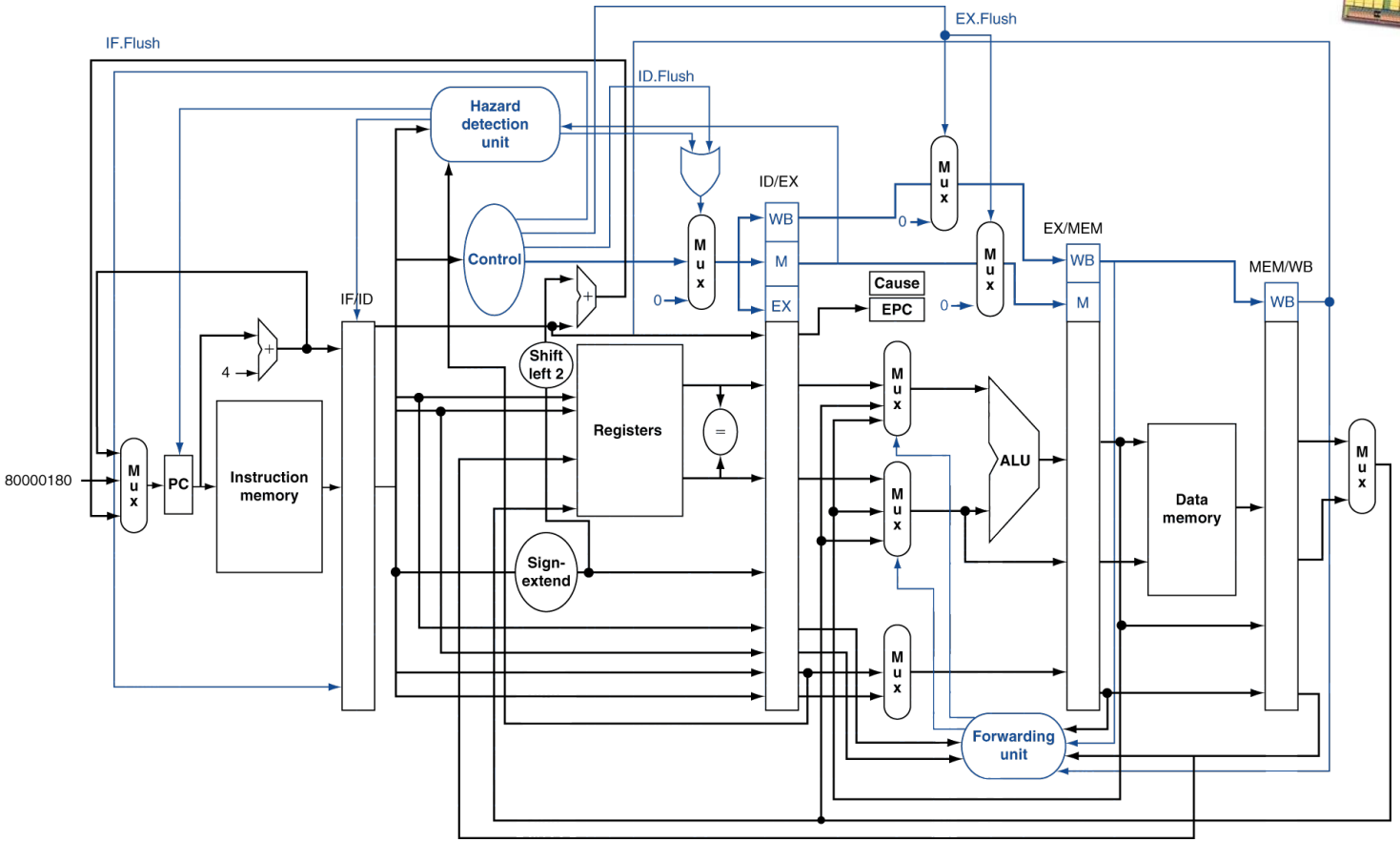
How to deal with **Complexity?**

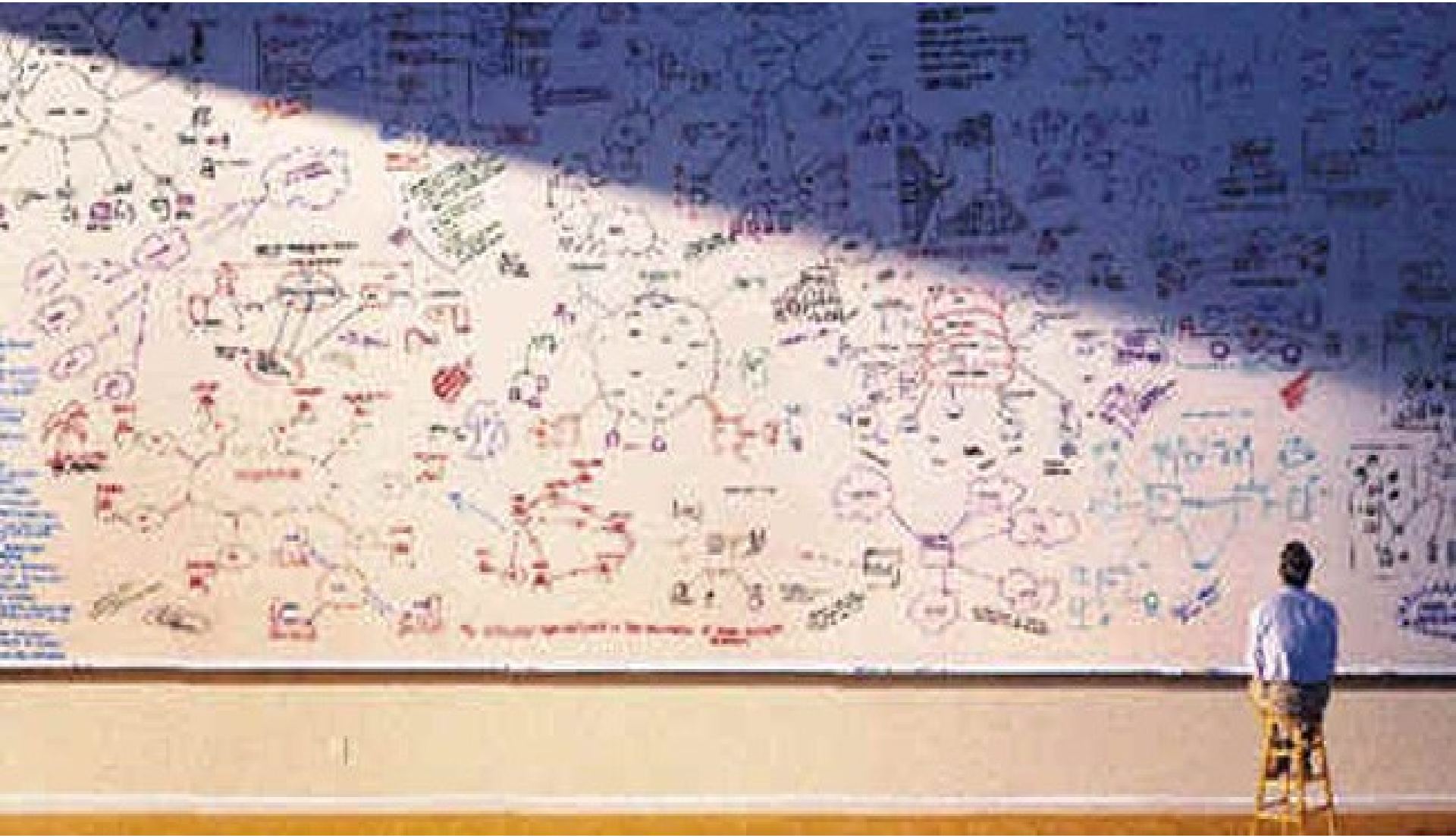
(in engineered systems)

“architectural” (hierarchical) **(de-)composition**



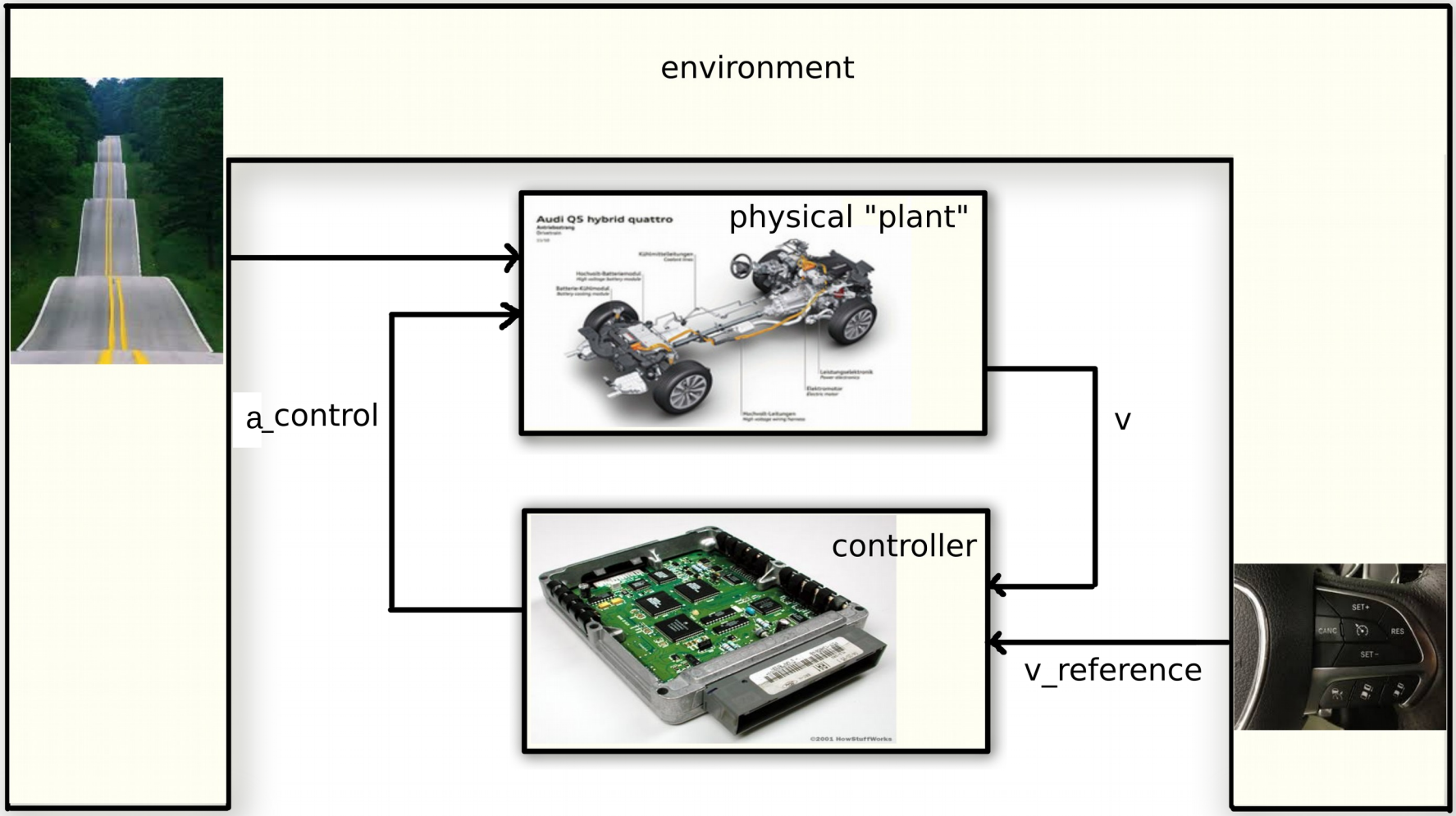




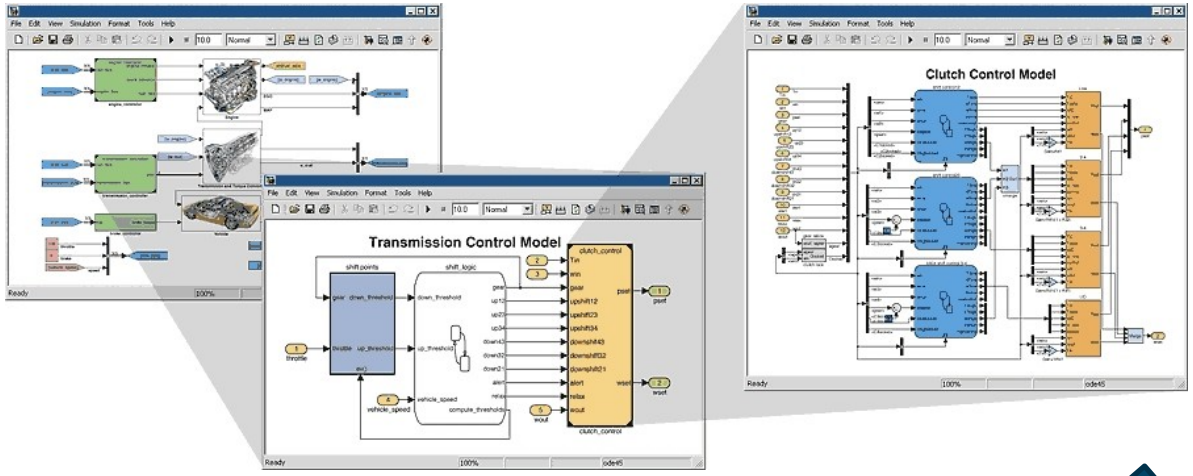


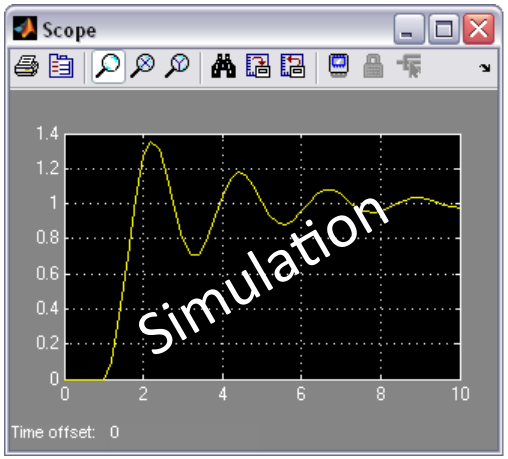
How to deal with **Complexity?**

(in engineered systems)

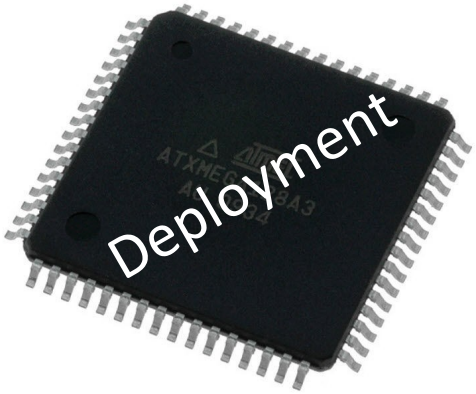

environment
physical "plant"
Audi Q5 hybrid quattro
a_control
v
controller
v_reference
SET+
CANC
RES
SET-

Model-Based System Design



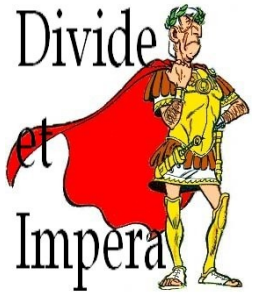


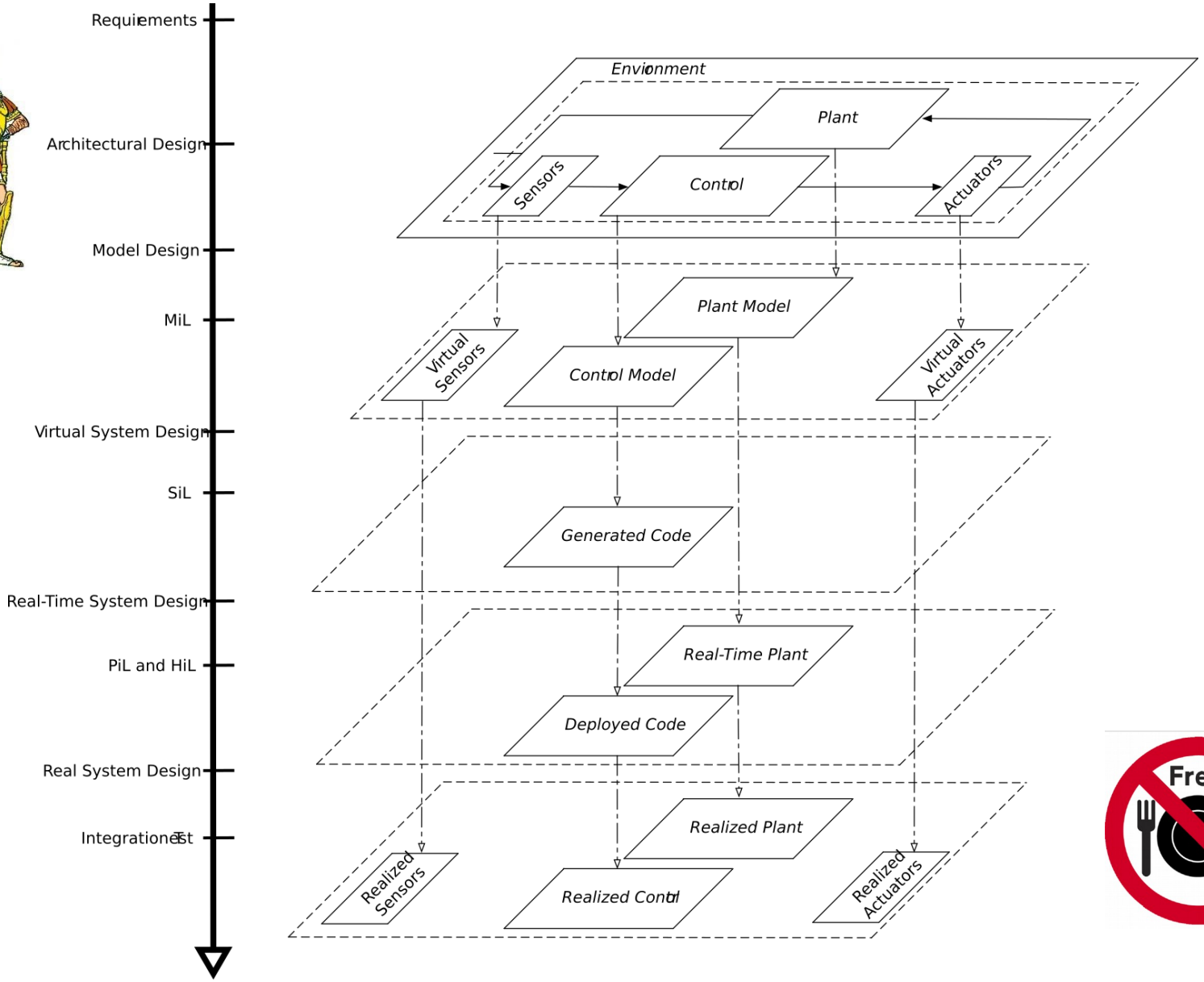
MiL, HiL, SiL, ...



XiL: X = Model, Software, Processor, Hardware

CoSys-Lab
Constrained Systems Lab
University of Antwerp

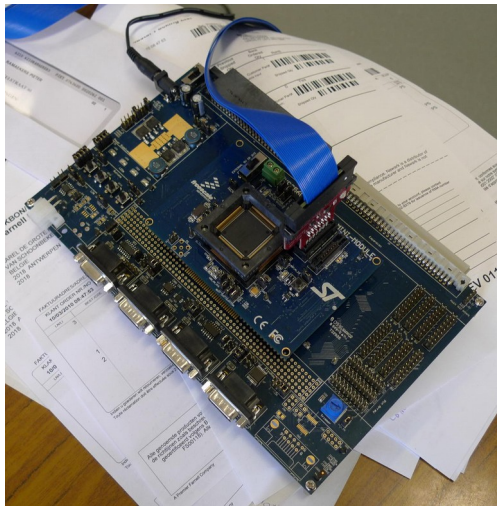


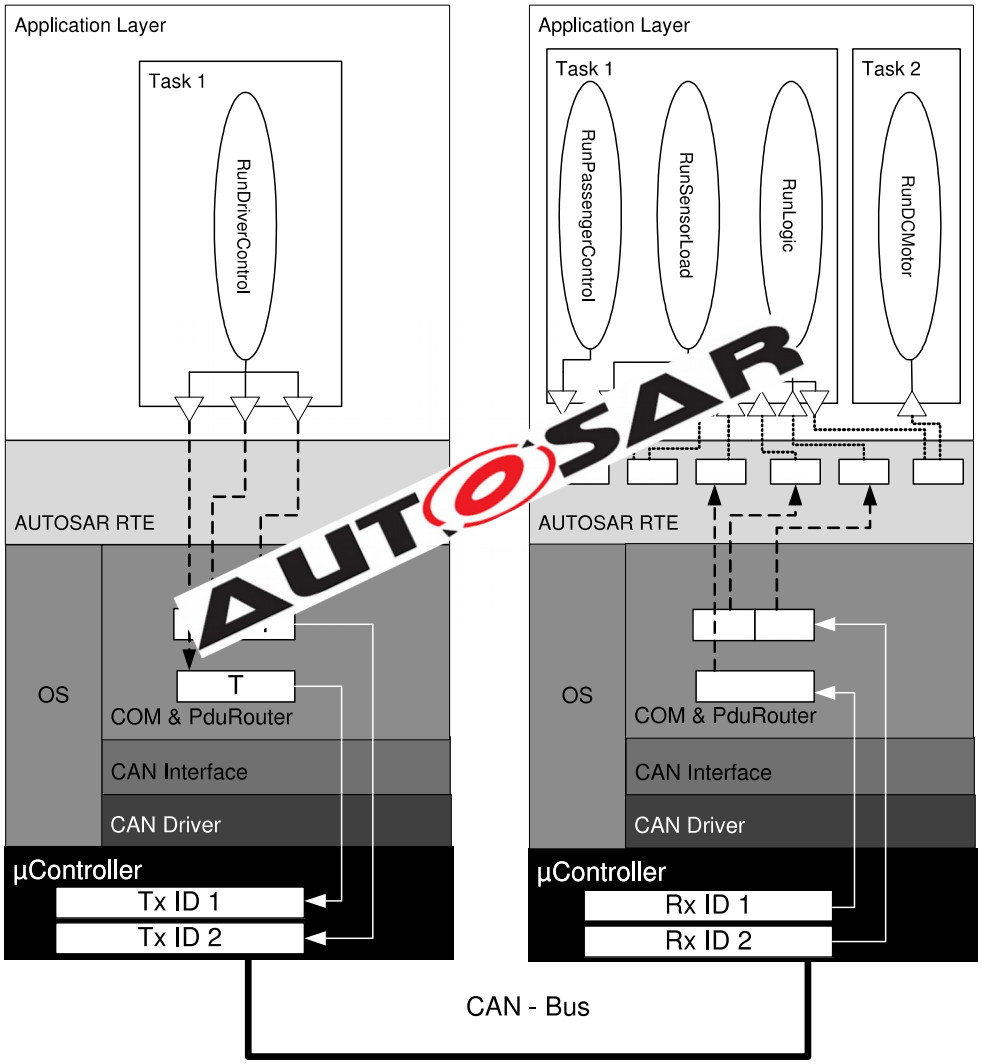


vertical consistency!

Ken Vanherpen. A contract-based approach for multi-viewpoint consistency in the concurrent design of cyber-physical systems. PhD thesis University of Antwerp. 2018.

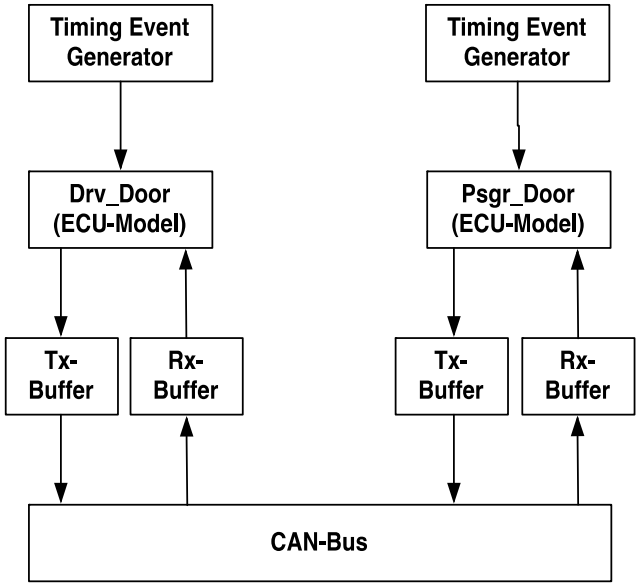
Deployment and Resource-Optimized Execution

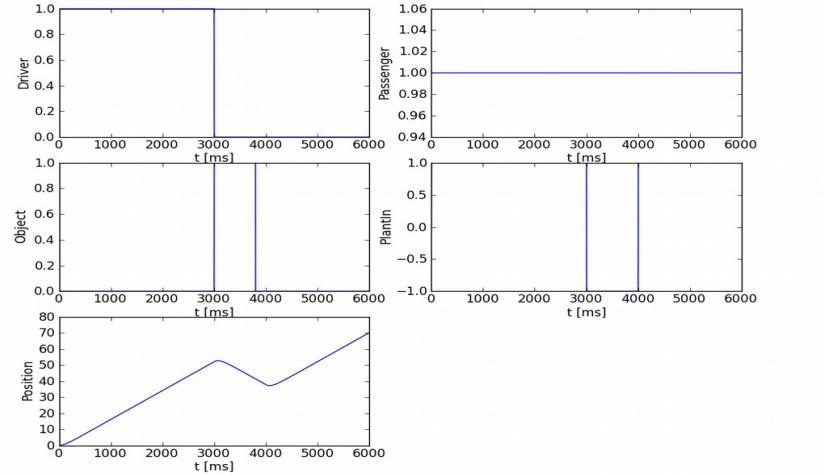


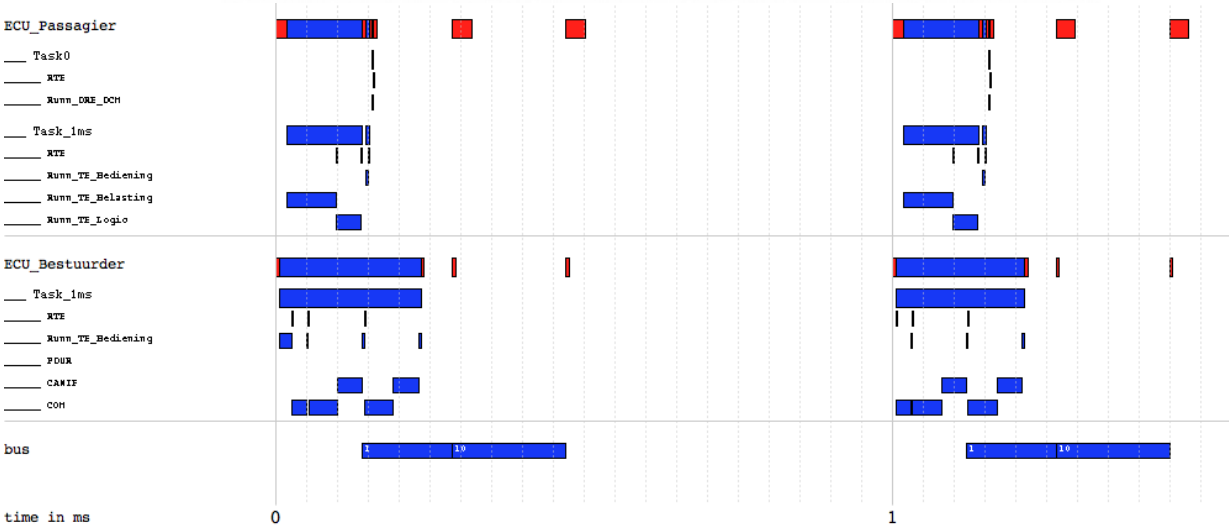


Joachim Denil, Paul De Meulenaere, Serge Demeyer, and Hans Vangheluwe. DEVS for AUTOSAR-based system deployment modeling and simulation. SIMULATION: Transactions of the Society for Modeling and Simulation International , 93(6):489 – 513, 2017.

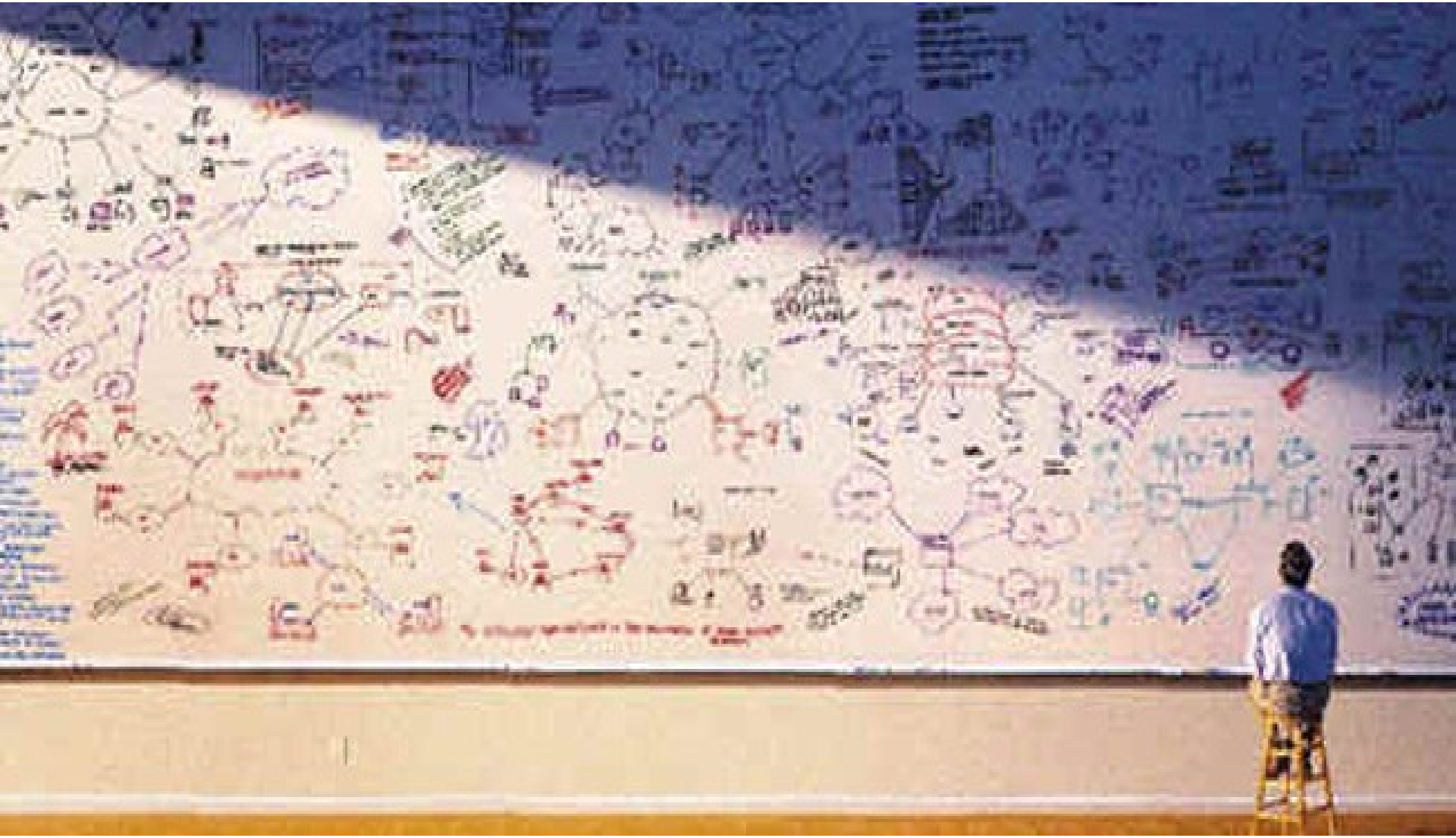
Deployment/Design-Space Exploration (trsf. To MILP, trsf. based)







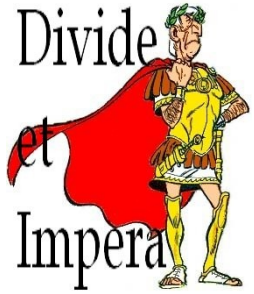
Joachim Denil, Hans Vangheluwe, Pieter Ramaekers, Paul De Meulenaere, and Serge Demeyer. DEVS for AUTOSAR platform modelling. In Spring Simulation Multiconference , pages 67 - 74. Society for Computer Simulation International (SCS), April 2011. Boston, MA, USA.



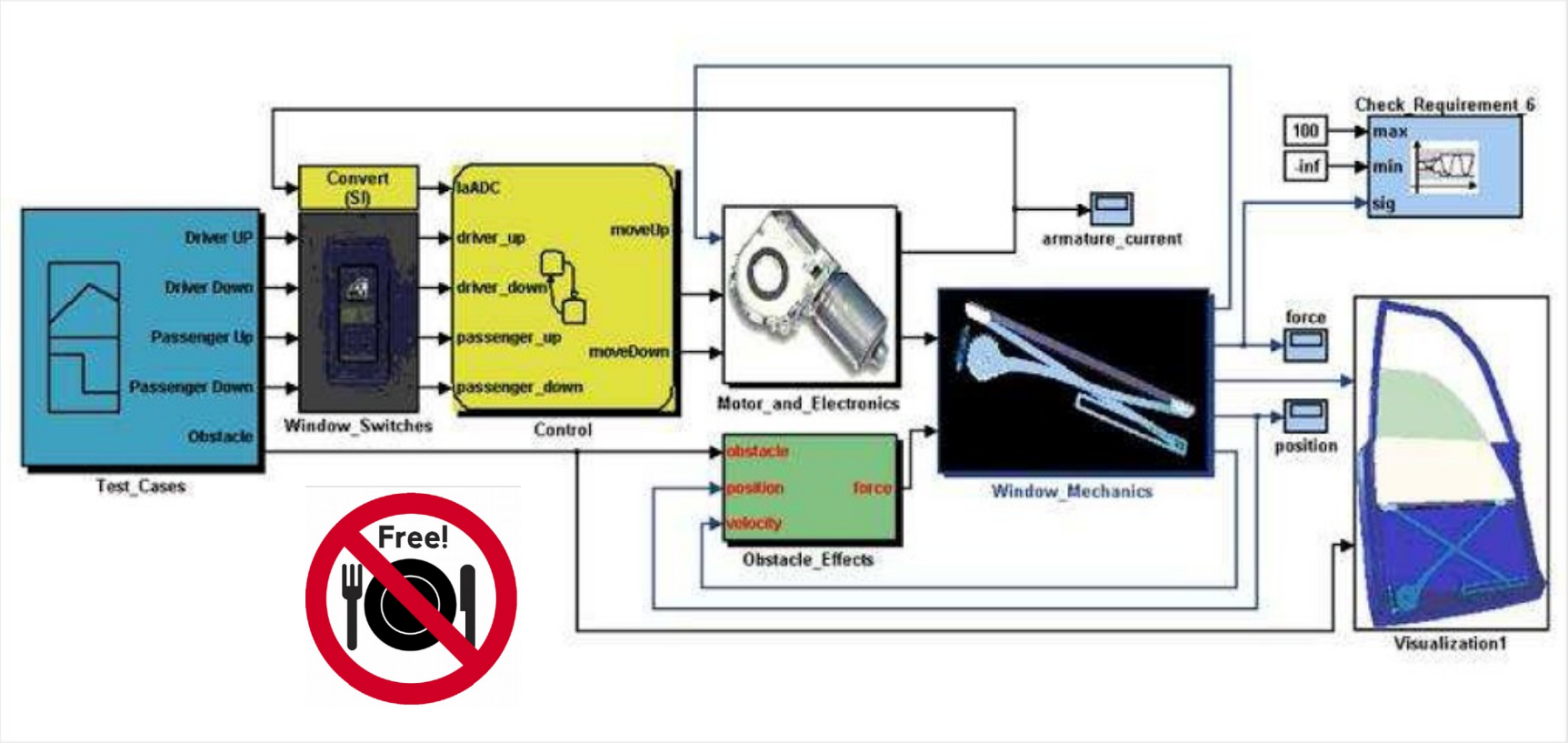
How to deal with **Complexity?**

(in engineered systems)

Most Appropriate Formalisms



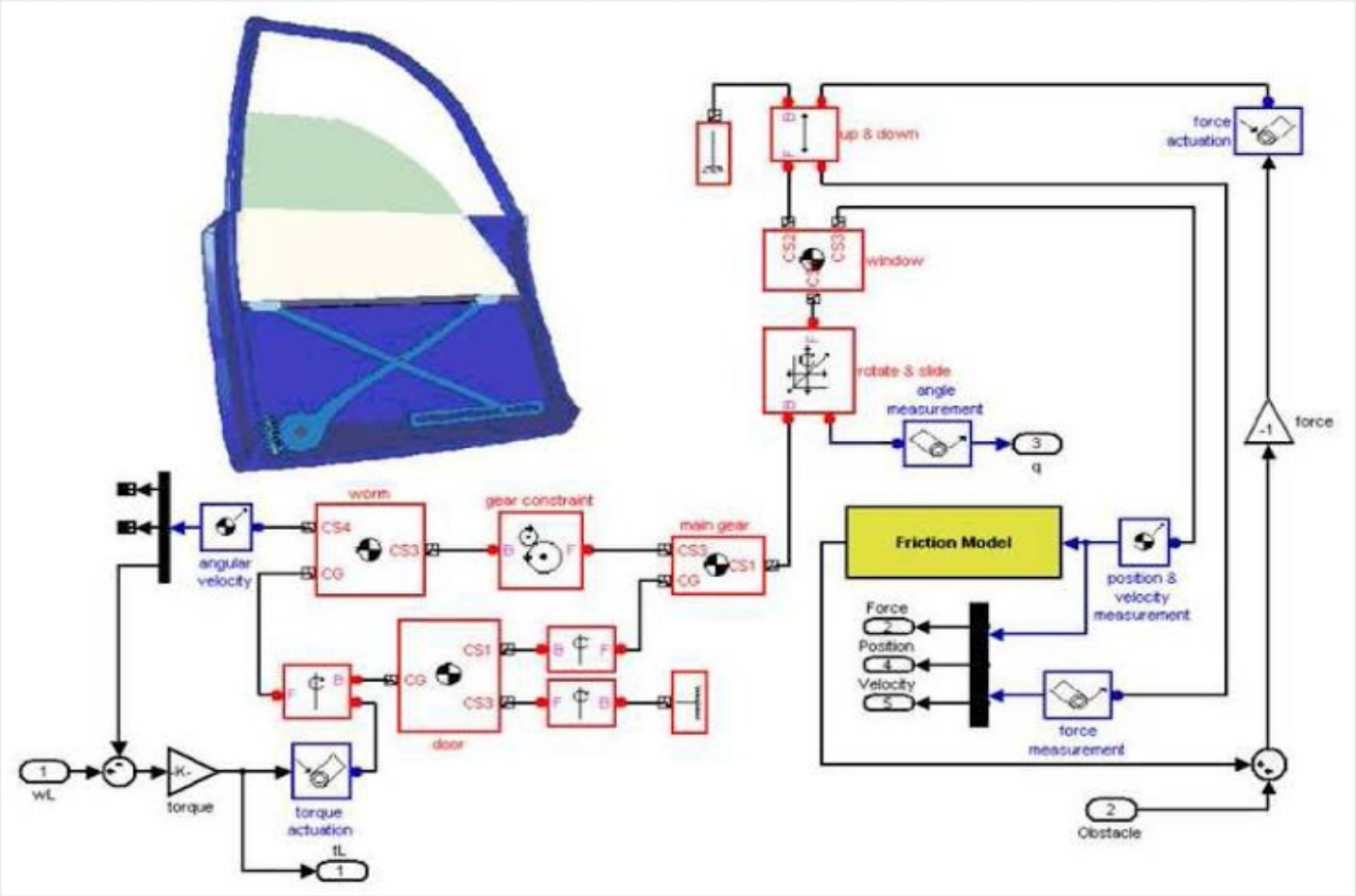
Components in Different Formalisms



www.mathworks.com/products/demos/simulink/PowerWindow/html/PowerWindow1.html

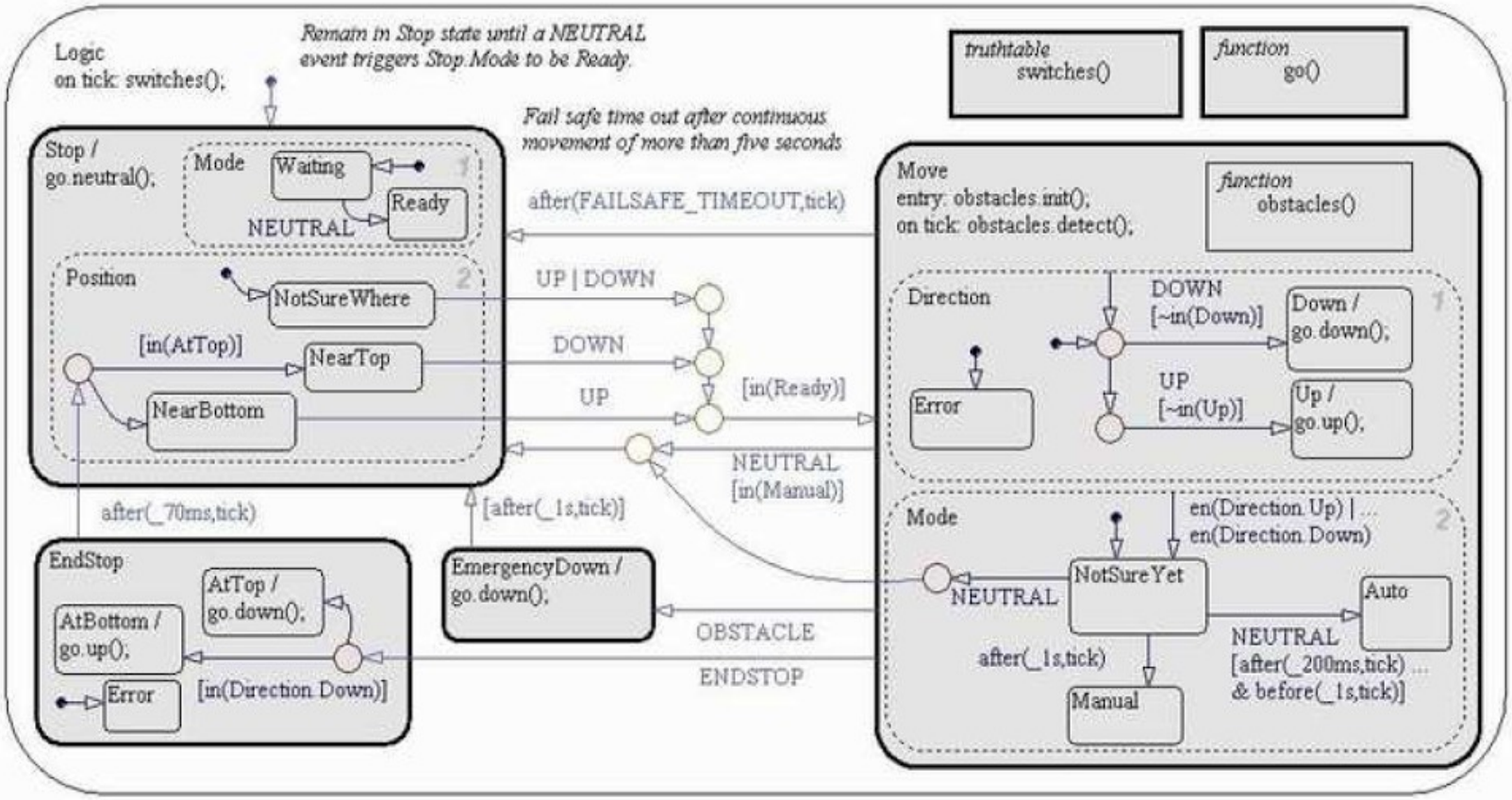
Most Appropriate Formalisms

Mechanics subsystem

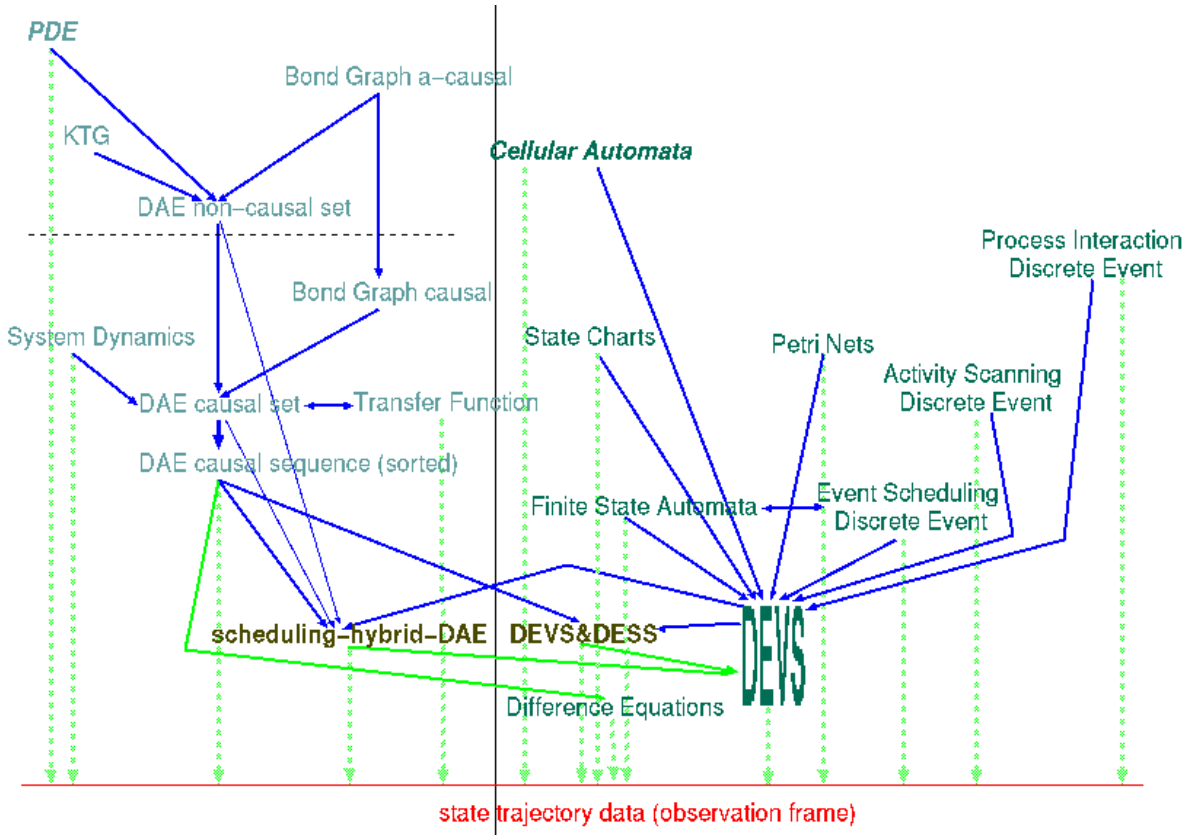


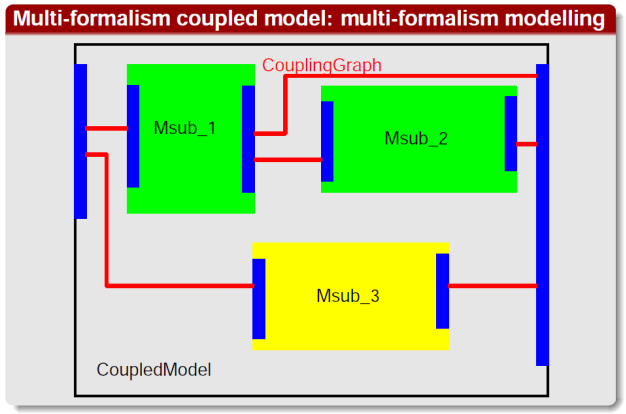
Most Appropriate Formalisms

Controller, using Statechart(StateFlow) formalism



problem : **analysis** of multi-formalism models
solution : transform to **common formalism**

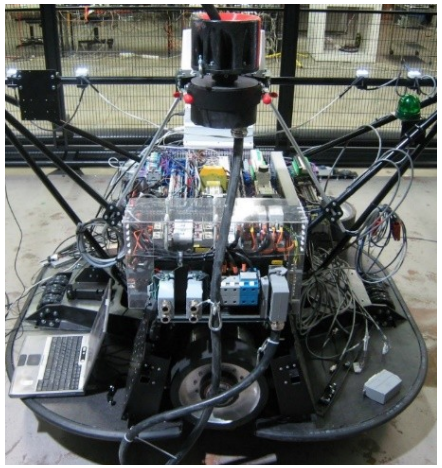


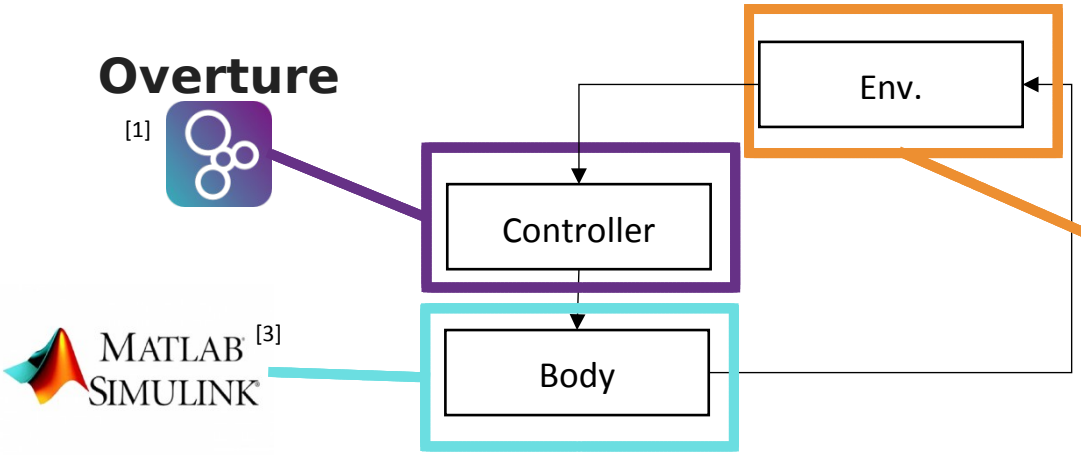


Hans Vangheluwe and Ghislain C. Vansteenkiste. A multi-paradigm modeling and simulation methodology: Formalisms and languages. In European Simulation Symposium (ESS), pp. 168-172. Society for Computer Simulation International (SCS), October 1996. Genoa, Italy.

Hans Vangheluwe and Juan de Lara. Meta-models are models too. Winter Simulation Conference, pp. 597 - 605. IEEE, December 2002.

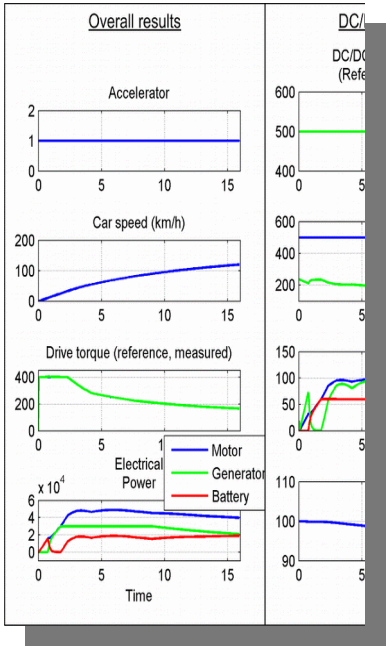
problem: **full-system analysis**
(also when IP protected)





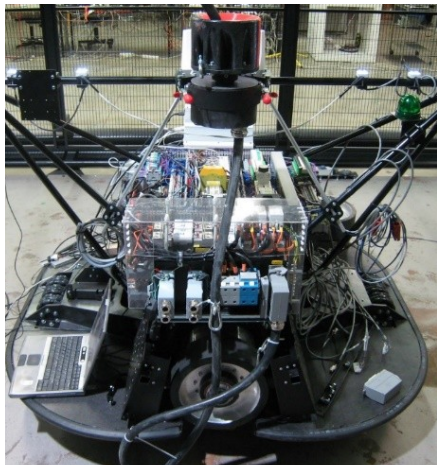
solution: combine
sub-system simulators

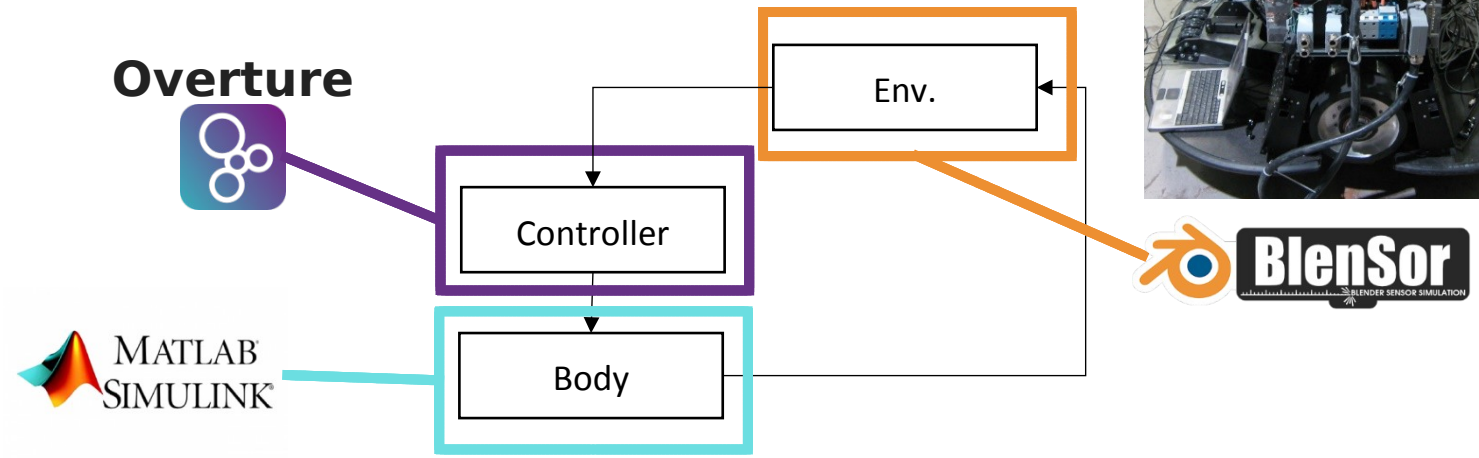
aka **co-simulation**

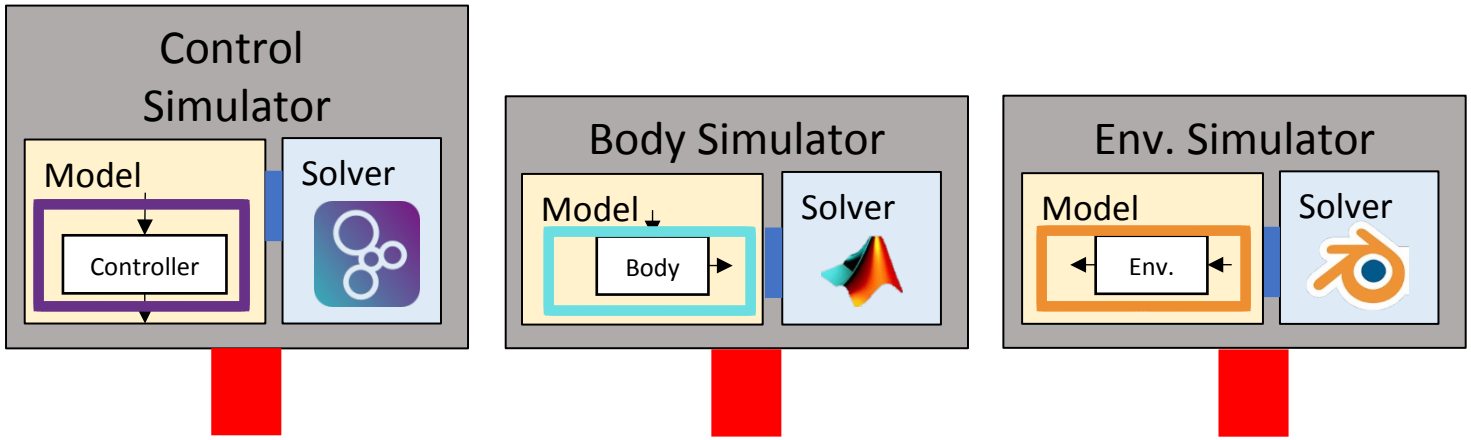


Cláudio Gomes, Casper Thule, David Broman, Peter Gorm Larsen, Hans Vangheluwe
Co-Simulation: A Survey. ACM Comput. Surv.51(3): 49:1-49:33 (2018)

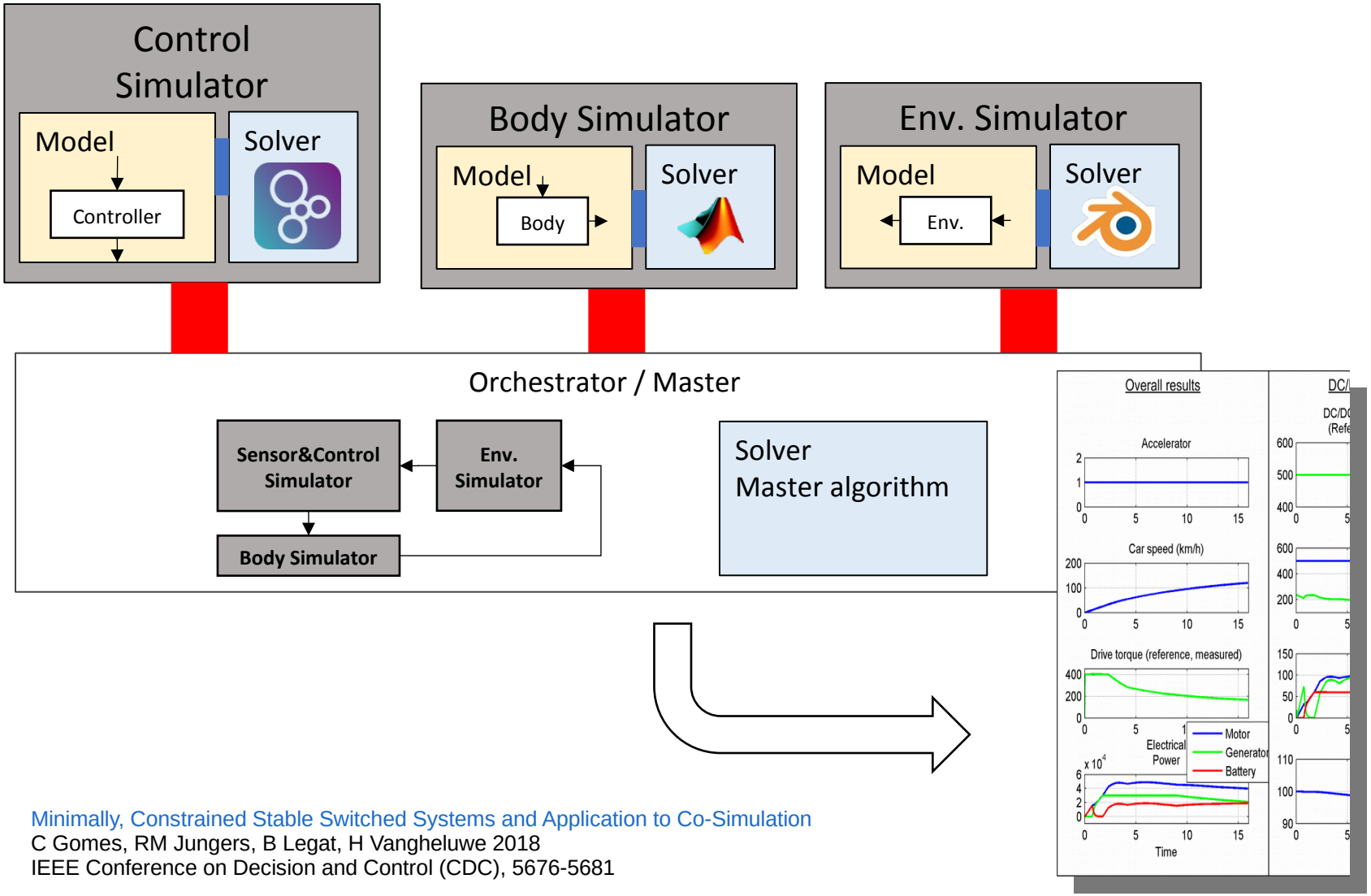
co-simulation: how?
(when IP protected)



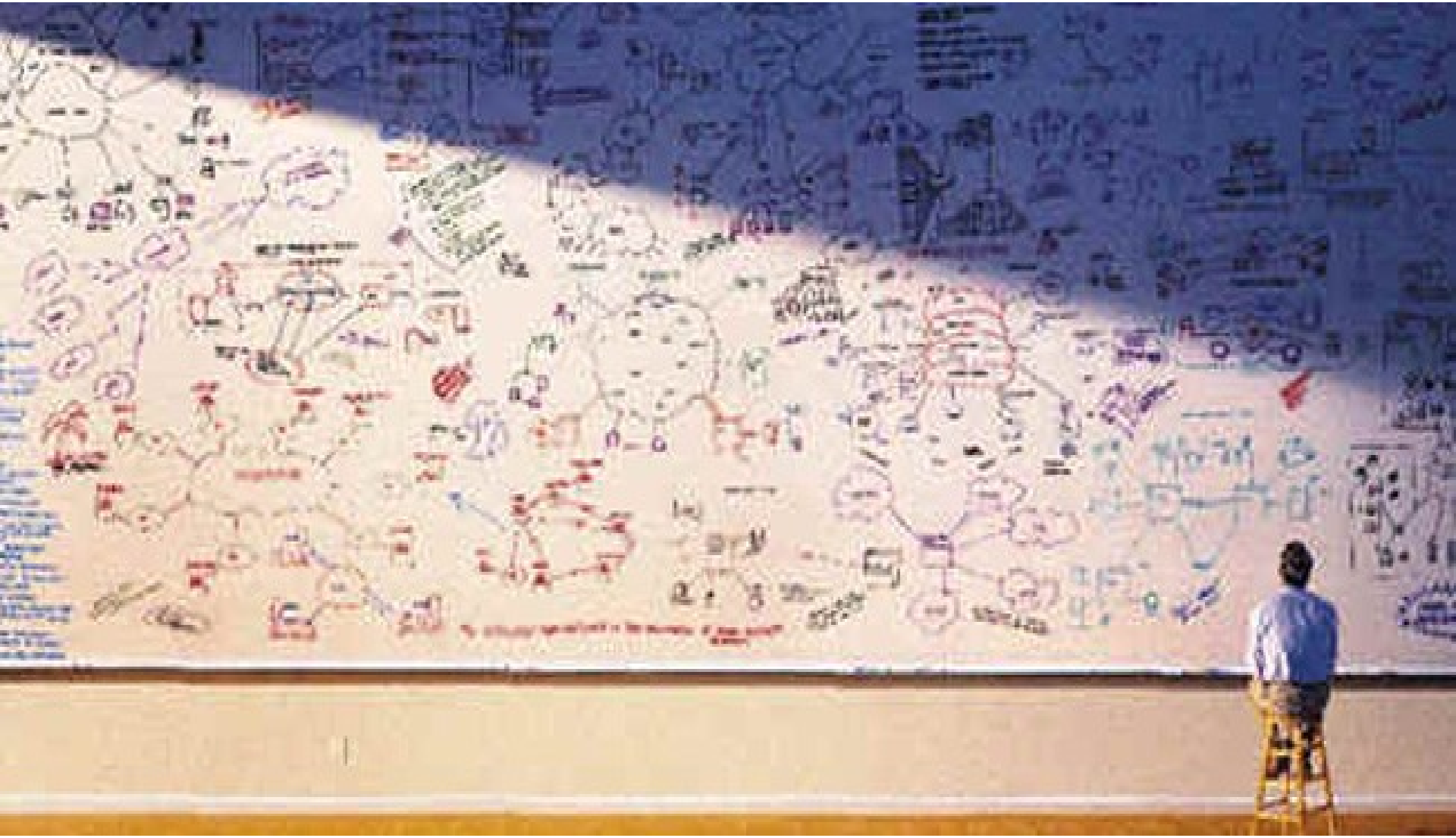




co-simulation: how?
(when IP protected)



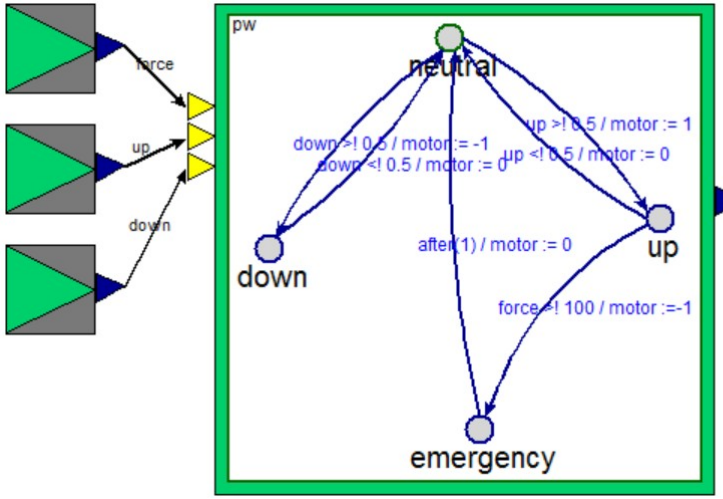
Minimally, Constrained Stable Switched Systems and Application to Co-Simulation
C Gomes, RM Jungers, B Legat, H Vangheluwe 2018
IEEE Conference on Decision and Control (CDC), 5676-5681

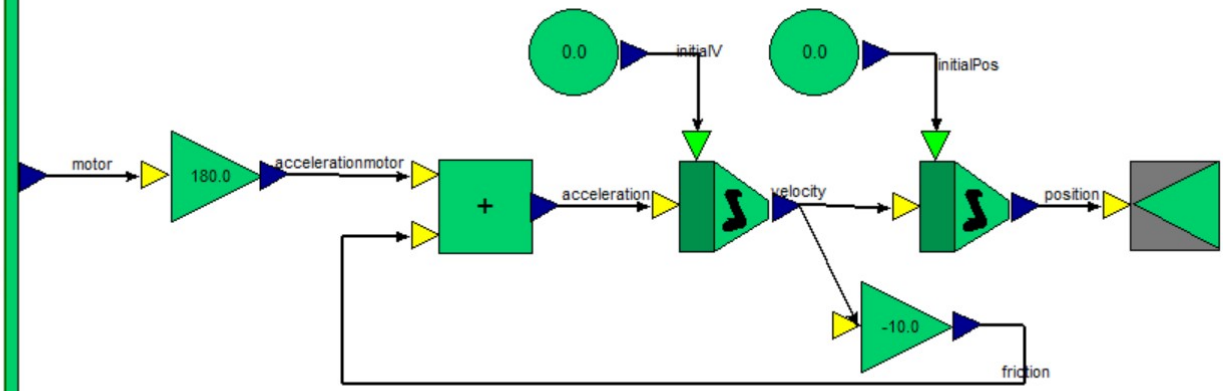


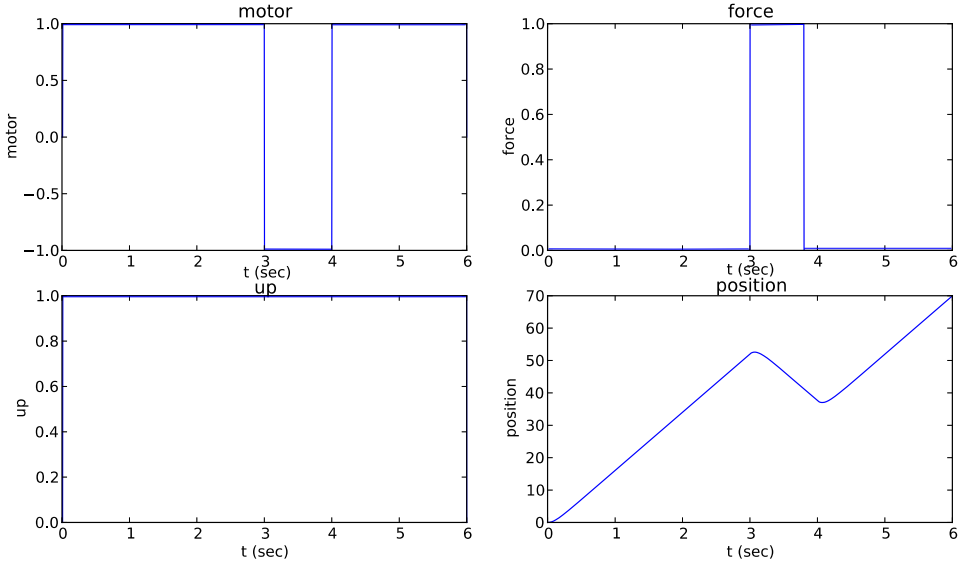
How to deal with **Complexity?**

(in engineered systems)

hybrid formalism (CBD + FSA)



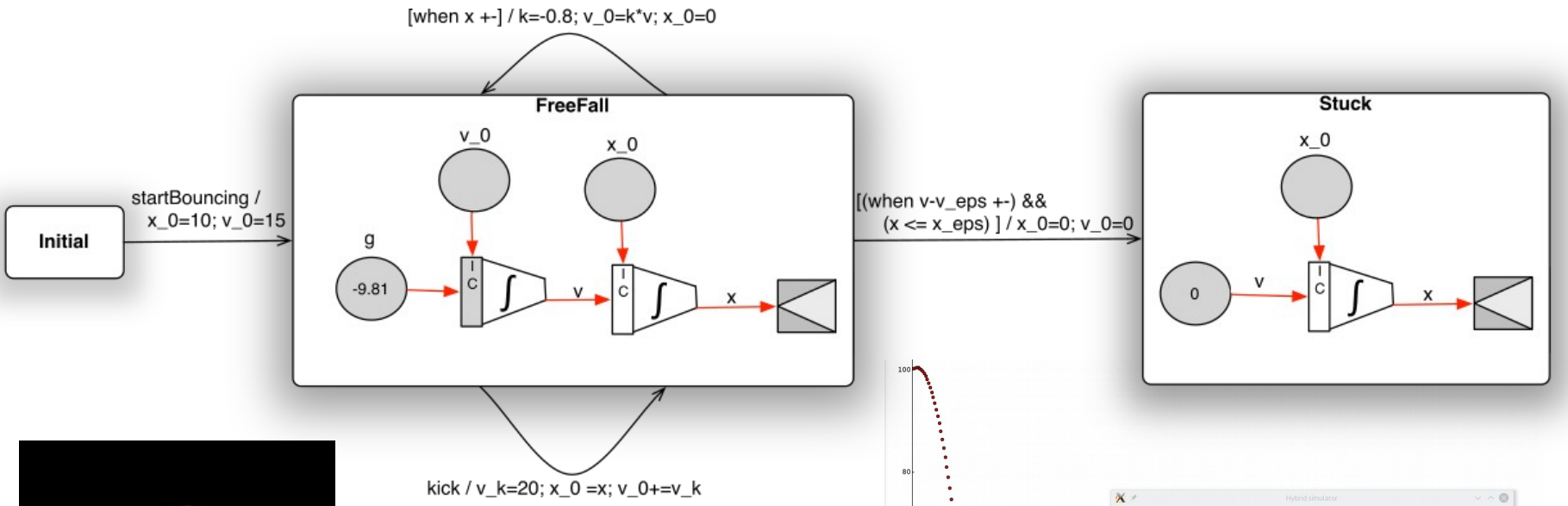


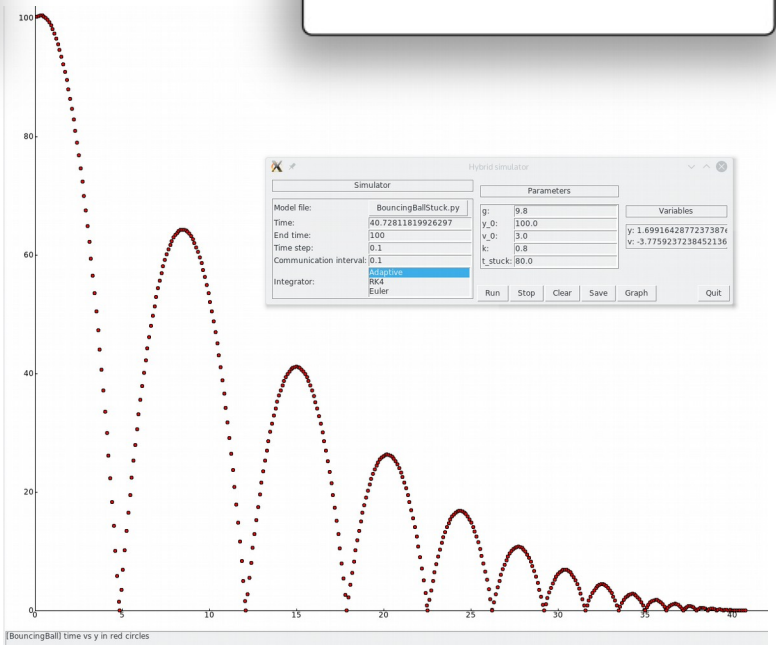


Bart Meyers, Joachim Denil, Frederic Boulanger, Cecile Hardebolle, Christophe Jacquet, Hans Vangheluwe. A DSL for Explicit Semantic Adaptation. MPM@MoDELS 2013:47-56.

Joachim Denil, Bart Meyers, Paul De Meulenaere, and Hans Vangheluwe. Explicit **semantic adaptation** of hybrid formalisms for **FMI co-simulation**. In Proceedings of the 2015 Spring Simulation Multi-Conference, pages 852 - 859. SCS, April 2015.

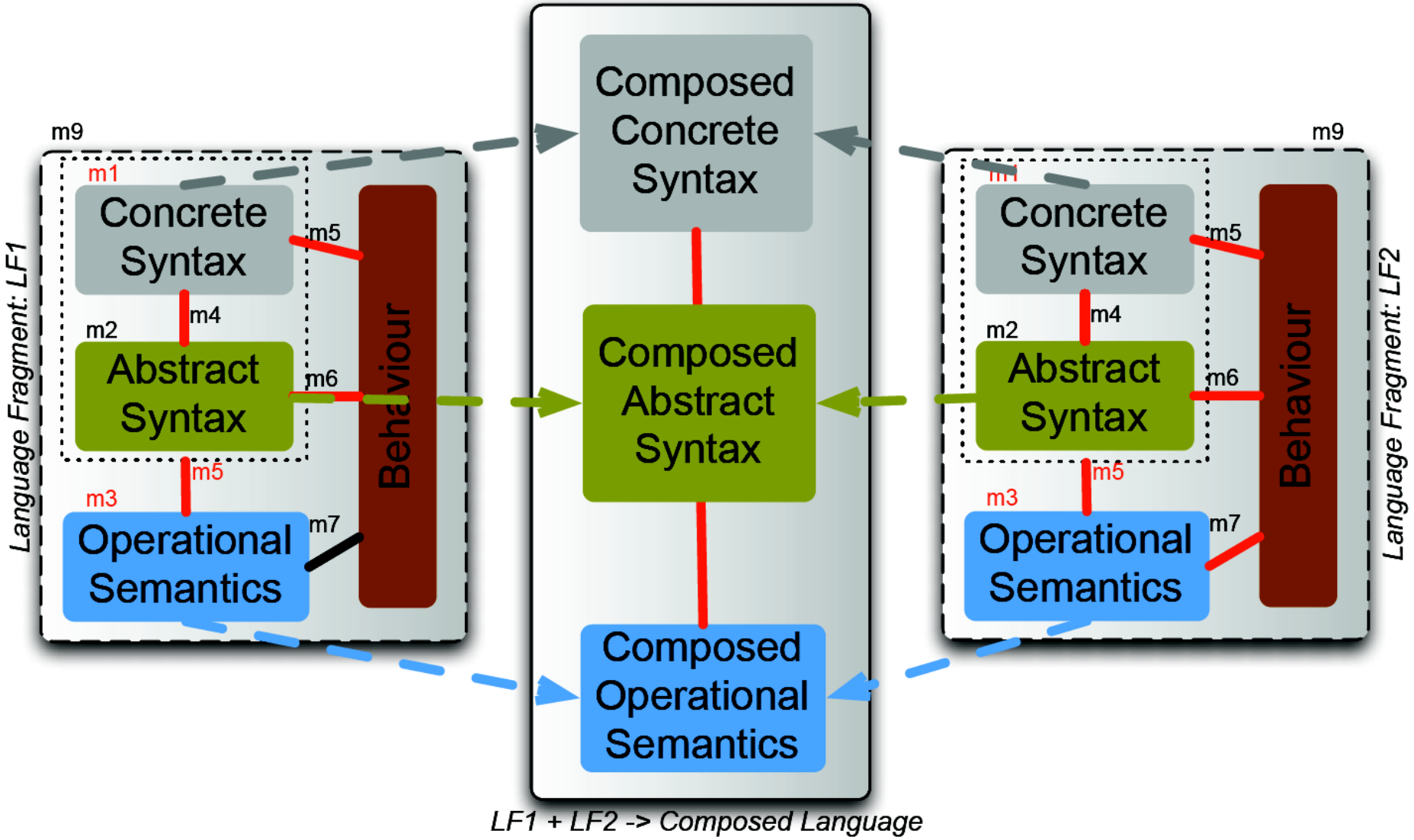
hybrid formalism (FSA+CBD)



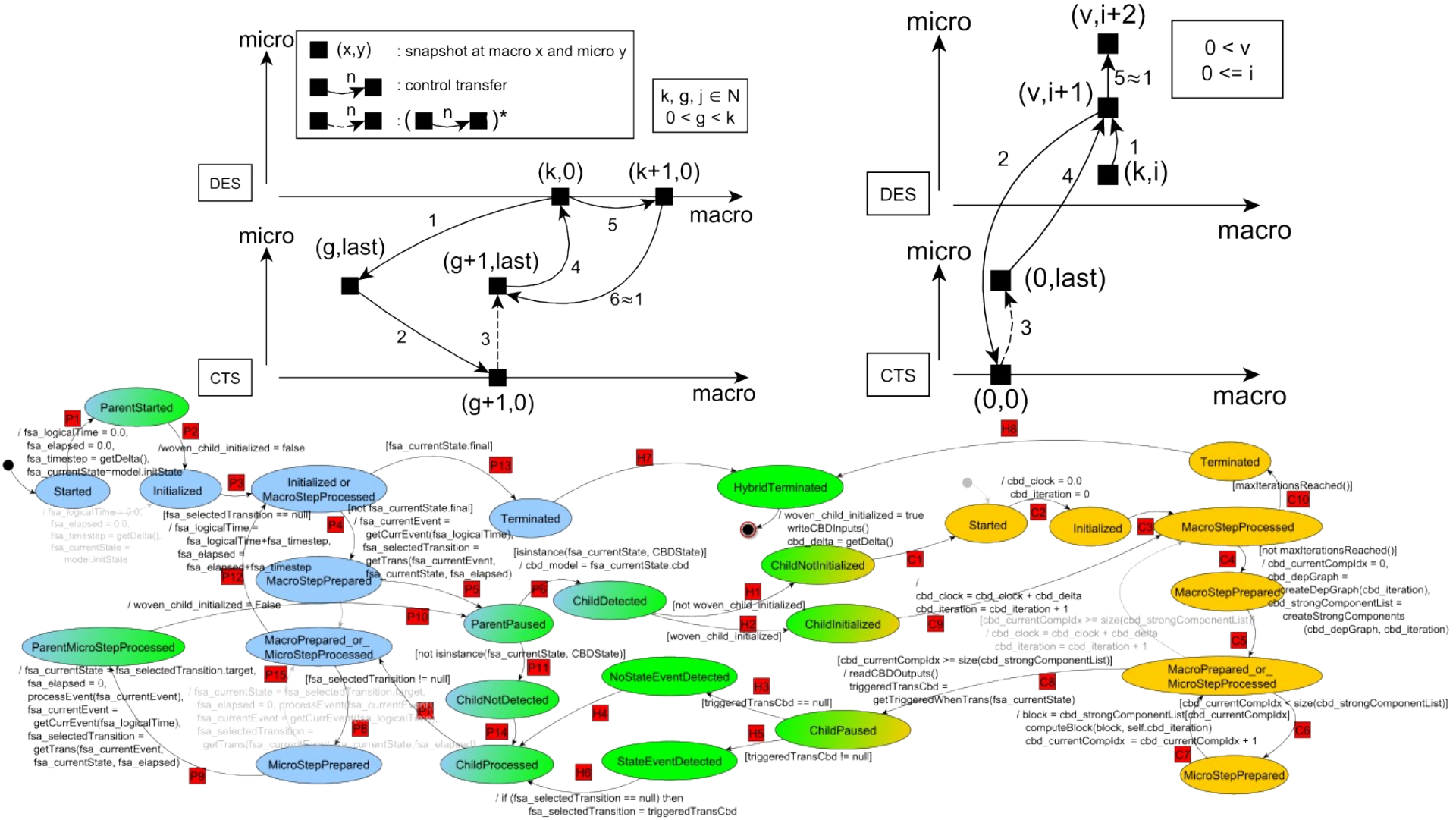


Simon Lacoste-Julien, Hans Vangheluwe, Juan de Lara, and Pieter J. Mosterman. Meta-modelling hybrid formalisms. In Pieter J. Mosterman and Jin-Shyan Lee, editors, IEEE International Symposium on Computer-Aided Control System Design , pages 65 - 70. IEEE Computer Society Press, September 2004. Taipei, Taiwan.

Hybrid Languages: Composition of Language Fragments



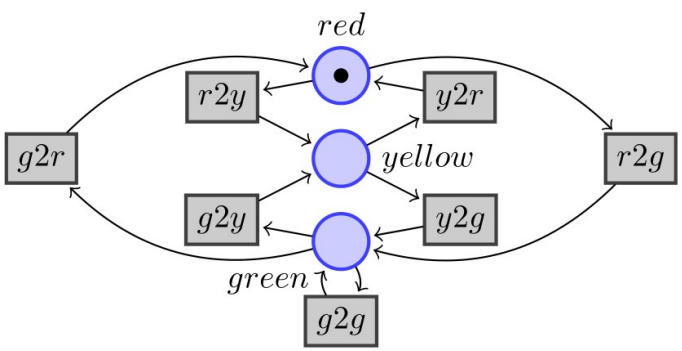
problem : **orchestration** of simulators
solution : model explicitly

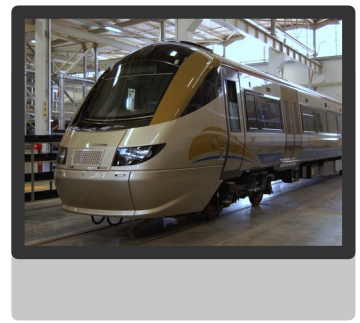


Sadaf Mustaz, Claudio Gomes, Bruno Barroca, and Hans Vangheluwe. Modular design of hybrid languages by explicit modeling of semantic adaptation. In Proceedings of TMS/DEVS '16/Spring Simulation Multi-Conference, pp 591-- 598. SCS, 2016.

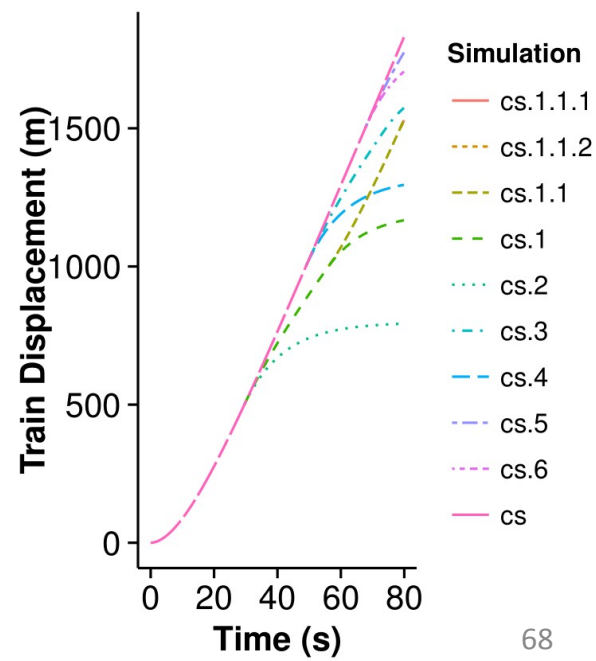
Matias Ezequiel Vara Larsen, Julien Deantoni, Benoit Combemale, Frédéric Mallet. A Behavioral Coordination Operator Language (BCOoL). 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS) pp.462 - 473, 2015

problem : **non-determinism**
solution : modify (FMI 2.0) co-simulation Master





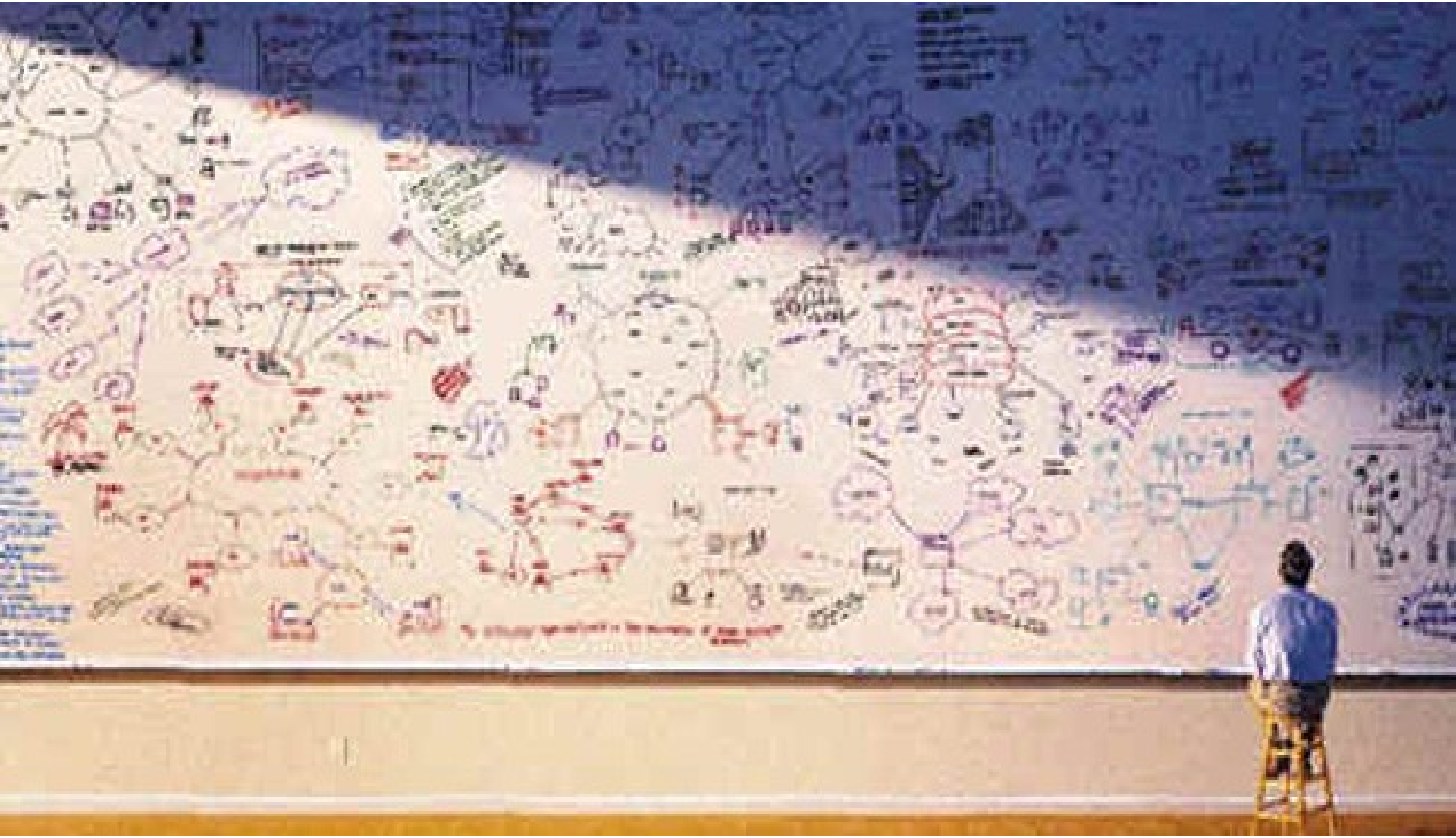
$$
\begin{cases}
e & = v_{ideal} - v \\
F_{traction} & = K_p \cdot e + K_i \cdot \int e \, d\tau + K_d \cdot \frac{de}{dt} \\
F_{spring} & = k(-x_{passger}) \\
F_{damper} & = c(-v_{passger})) \\
\frac{dv_{passger}}{dt} & = \frac{F_{spring} + F_{damper} - m_{passger} * \frac{F_{traction}}{(m_{train} + m_{passger})}}{m_{passger}} \\
\frac{dv_{train}}{dt} & = \frac{F_{traction}}{(m_{train} + m_{passger})} \\
\frac{dx_{passger}}{dt} & = v_{passger} \\
\frac{dx_{train}}{dt} & = v_{train}
\end{cases}
$$



Mancini, T., Mari, F., Massini, A., Melatti, I., Tronci, E. (2014). Anytime System Level Verification via Random Exhaustive Hardware in the Loop Simulation. In *Digital System Design (DSD), 2014 17th Euromicro Conference on* (pp. 236–245). http://doi.org/10.1109/DSD.2014.91

Lawrence, D. P. Y., Gomes, C., Denil, J., Vangheluwe, H., Buchs, D. (2016). **Coupling Petri nets with Deterministic Formalisms Using Co-simulation**. In *Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium* (p. 6:1--6:8). Pasadena, CA, USA.

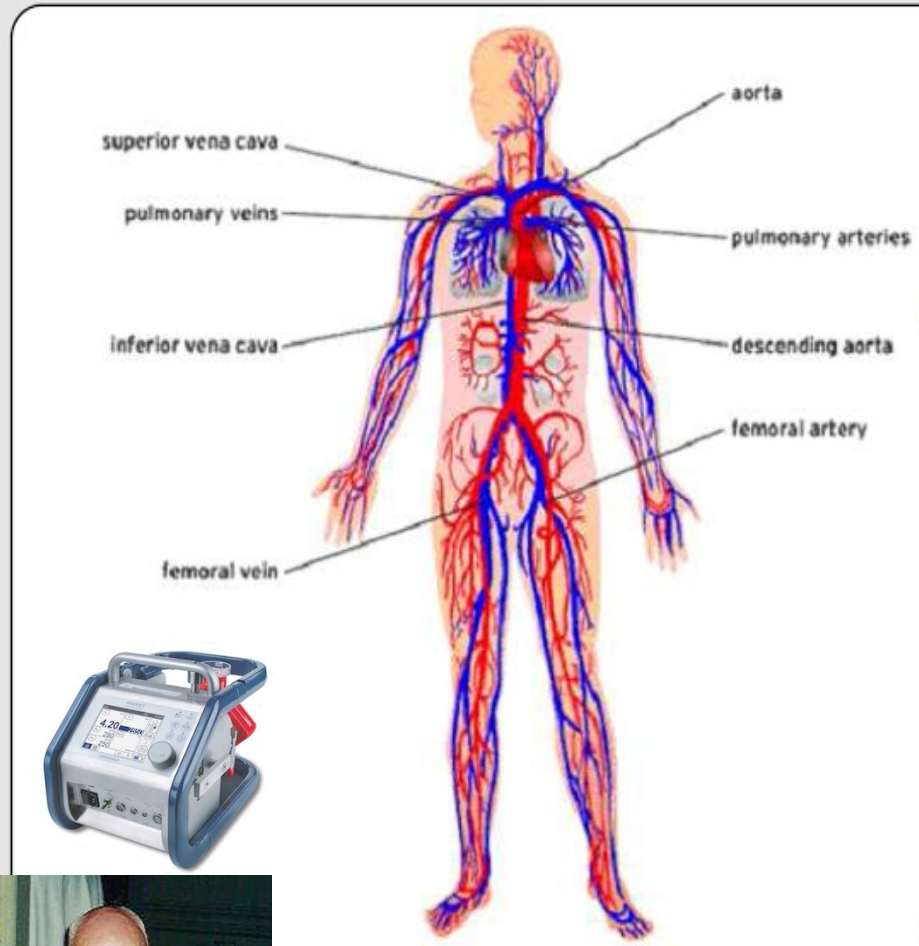
Clarke, E., Donzé, A., Legay, A. (2010). On simulation-based probabilistic model checking of mixed-analog circuits. *Formal Methods in System Design*, *36*(2), 97–113. http://doi.org/10.1007/s10703-009-0076-y

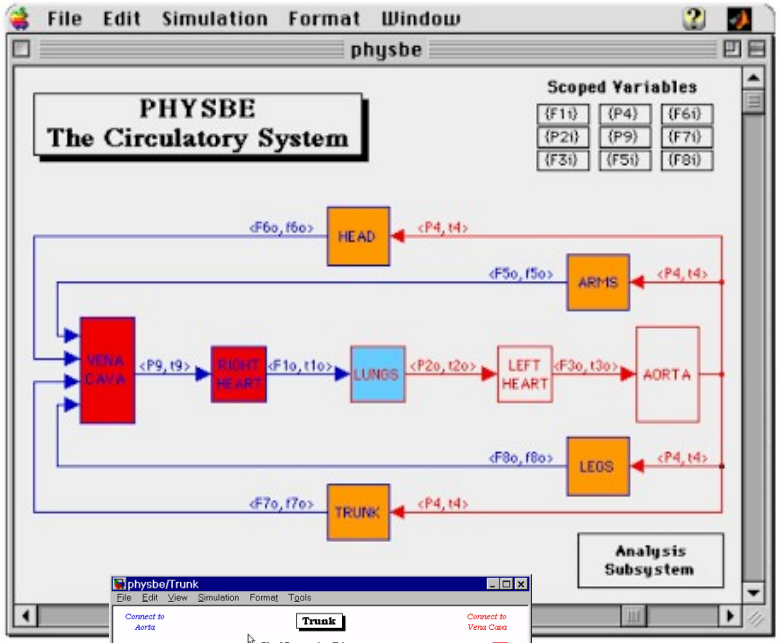


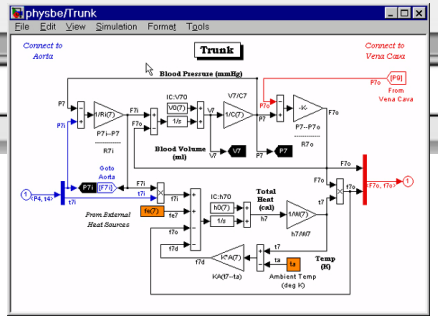
How to deal with **Complexity?**

(in engineered systems)

Different Abstraction Levels – properties preserved







THE SOCIETY FOR MODELING & SIMULATION INTERNATIONAL

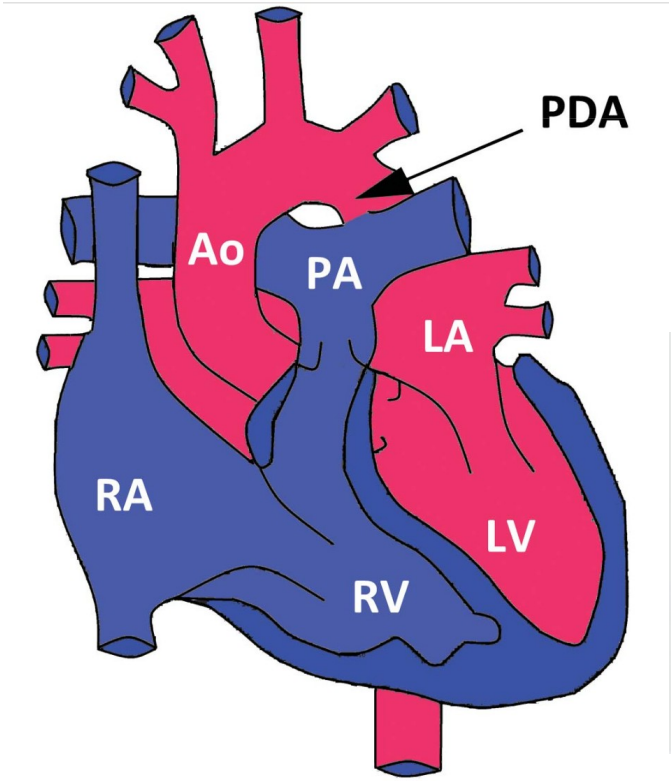
McLeod J. PHYSBE ... a physiological simulation benchmark experiment SIMULATION vol 7 no 6 December **1966** pp 324-329

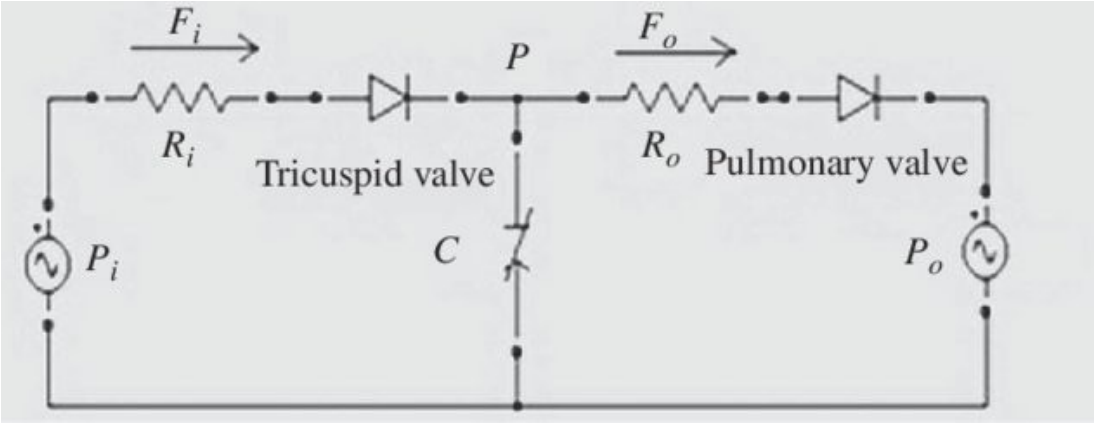
Mathematical and Computer Modelling of Dynamical Systems, 2013
Vol. 19, No. 3, 238–249, http://dx.doi.org/10.1080/13873954.2012.727187

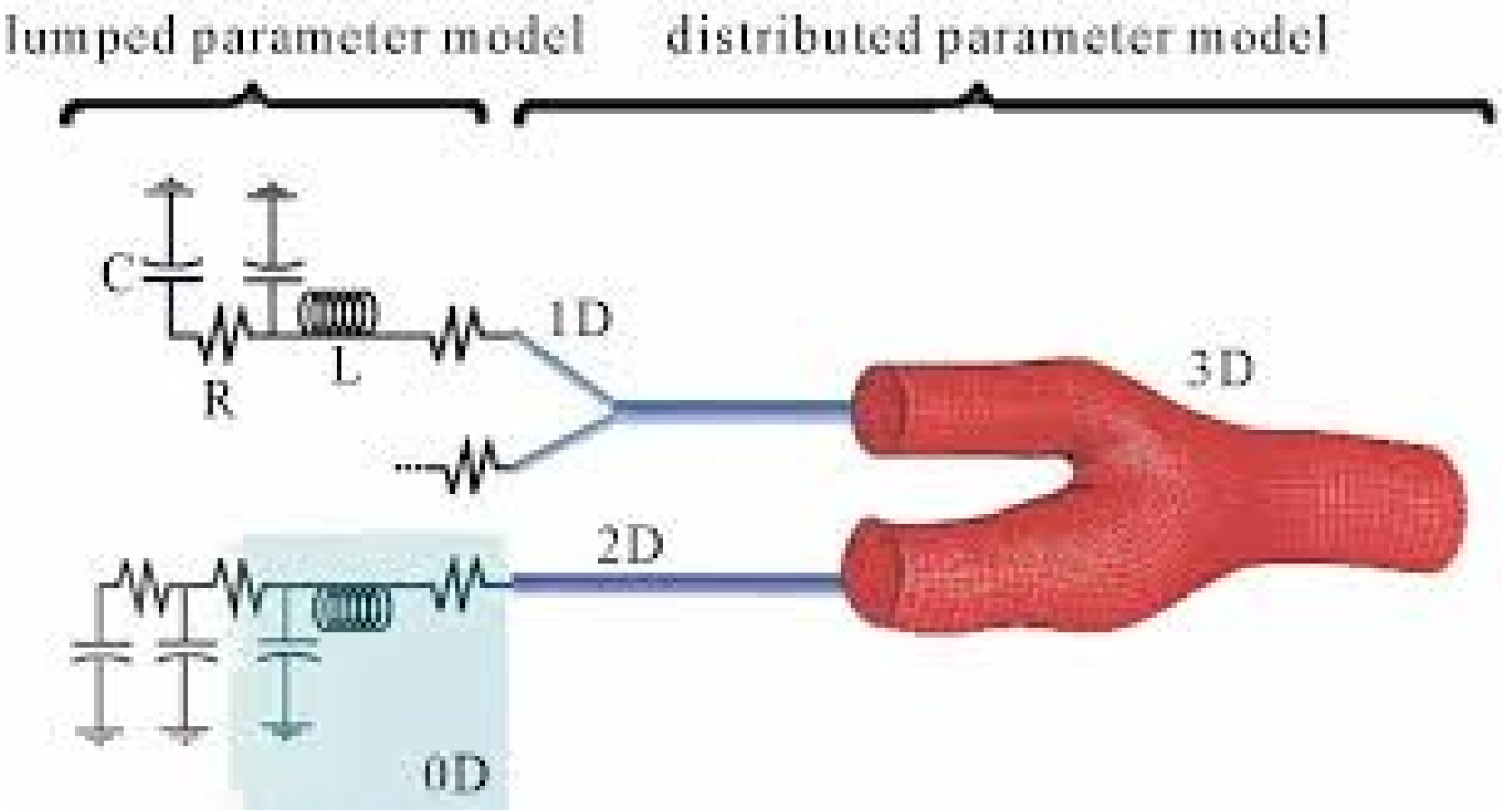
Mathematical modelling of the patent ductus arteriosus (PDA)

Mohamad Amin Bakhshali, Mahsa Mafi and Sabalan Daneshvar*

Biomedical Engineering Department, Electrical Engineering Faculty, Sahand University of Technology, Tabriz, Iran





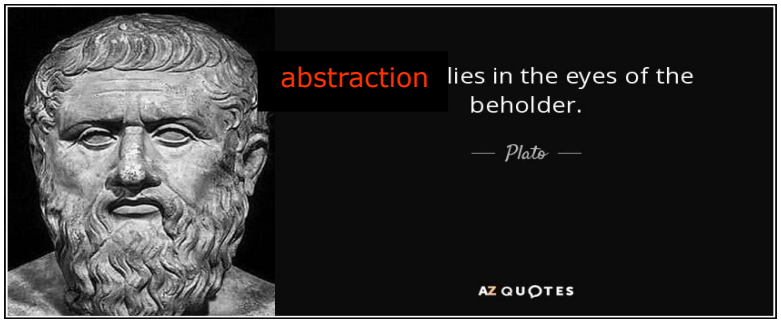

lumped parameter model
distributed parameter model
C
R
L
1D
3D
2D
0D

Abstraction Relationship

foundation: the *information* contained in a model M.
Different *questions* (properties) $P = I(M)$ which can be asked concerning the model.
These questions either result in true or false.

Abstraction and its opposite, *refinement* are *relative to a non-empty set of questions* (properties) P.

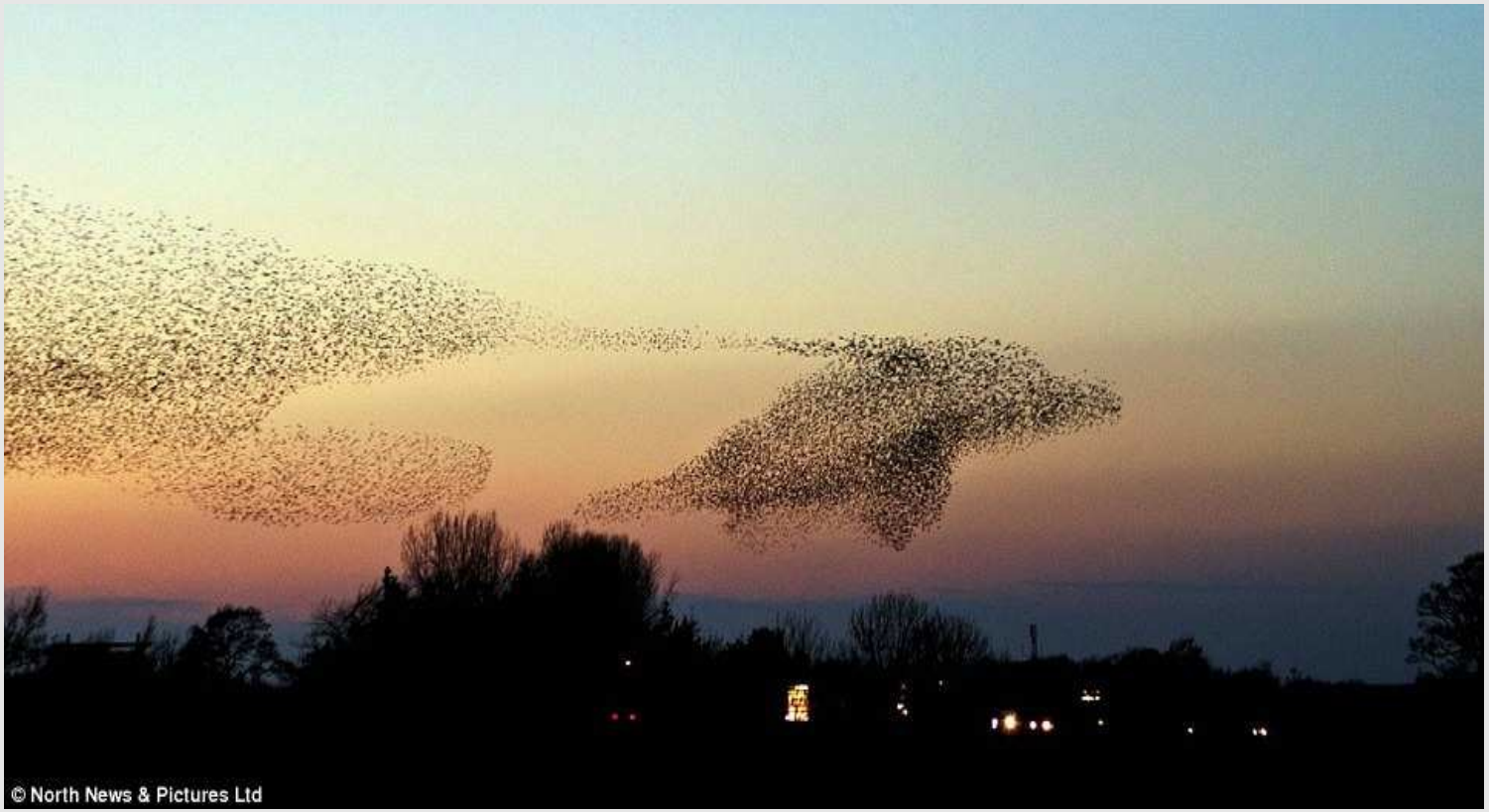
- If M_1 is an *abstraction* of M_2 with respect to P, for all $p \in P$: $M_1 \models p \Rightarrow M_2 \models p$. This is written $M_1 \sqsupseteq_P M_2$.
- M_1 is said to be a *refinement* of M_2 iff M_2 is an *abstraction* of M_1. This is written $M_1 \sqsubseteq_P M_2$.



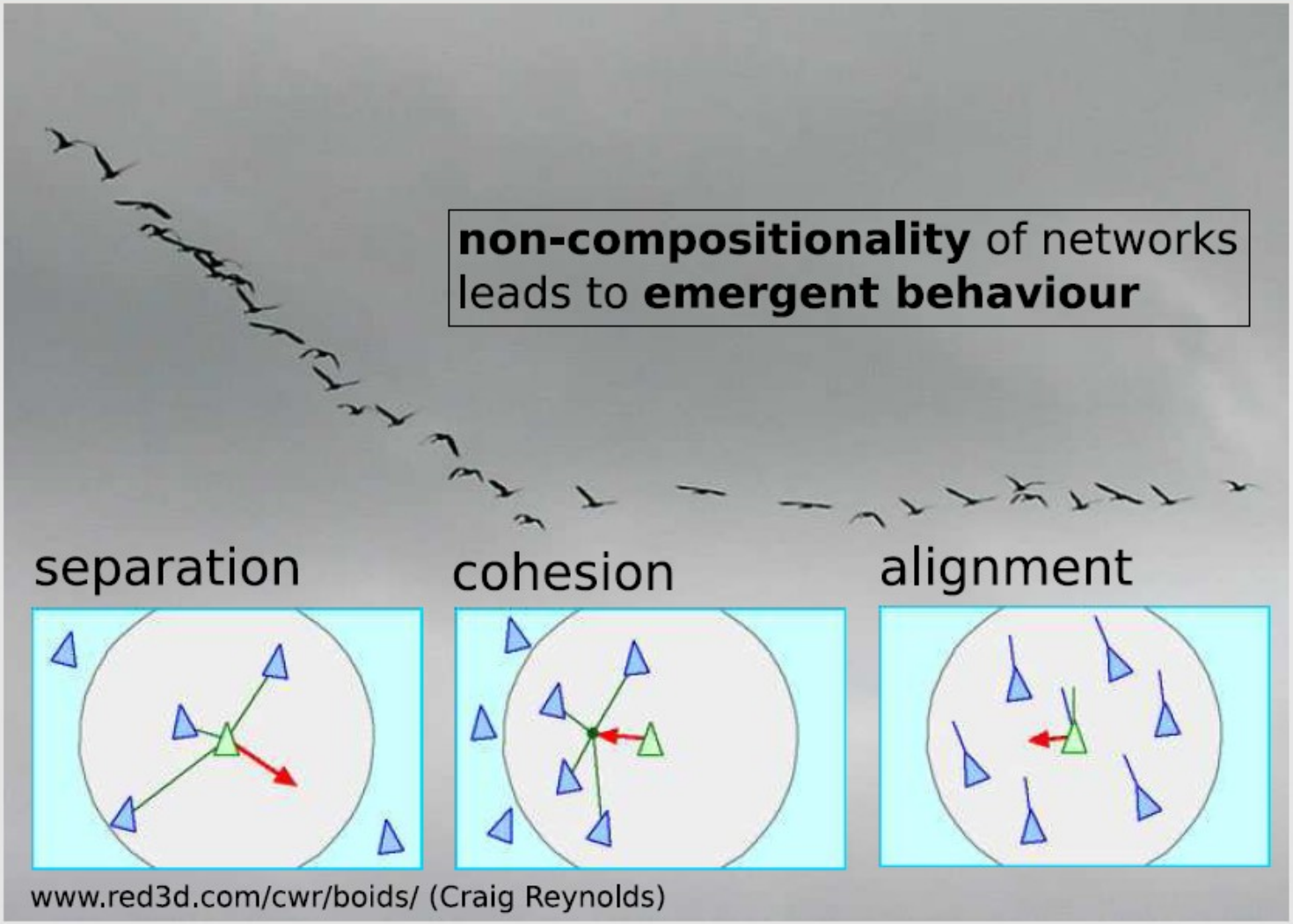
and depends

on the **properties** of interest!

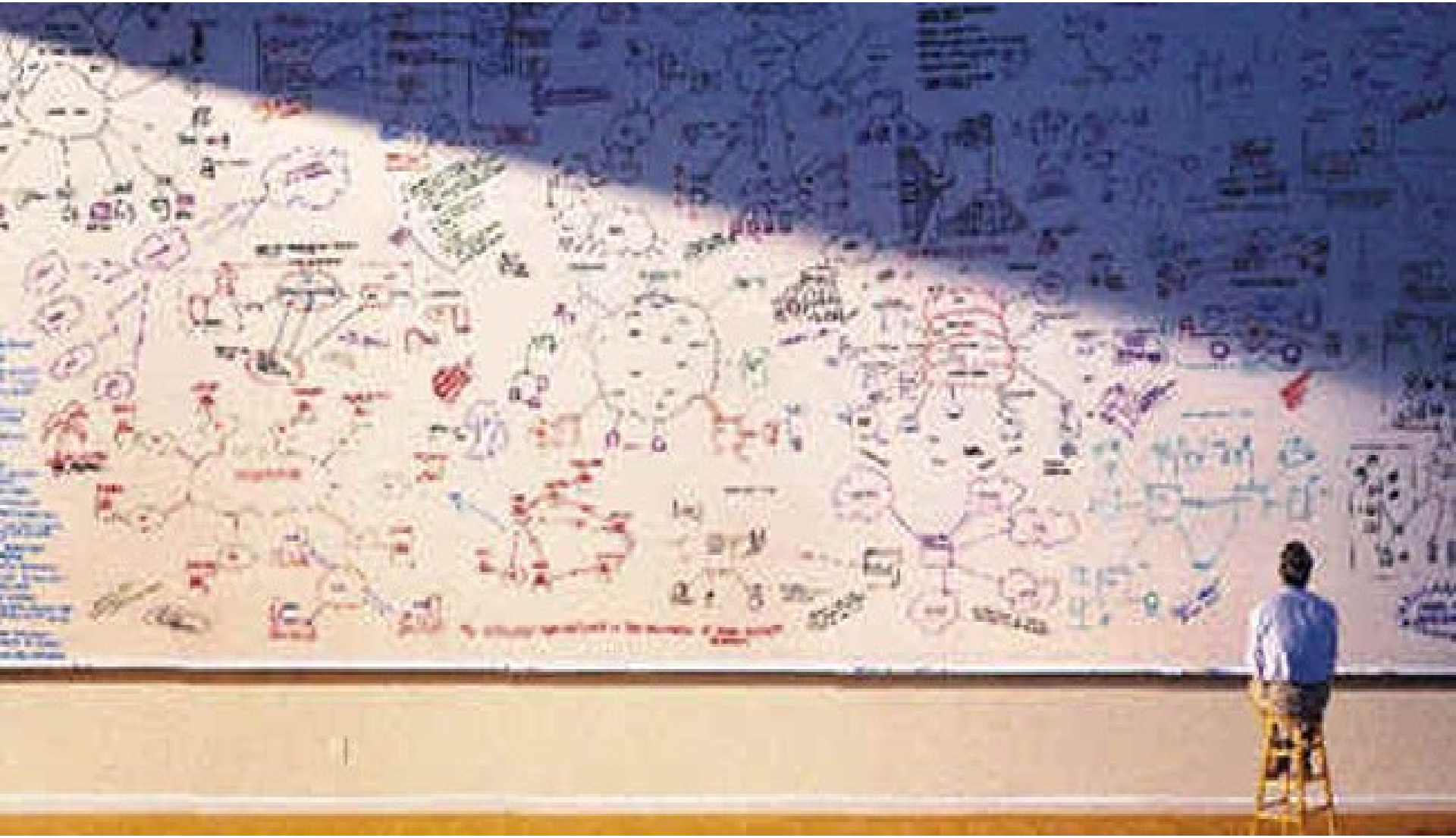
Emergent Behaviour



Non-compositional/Emergent Behaviour

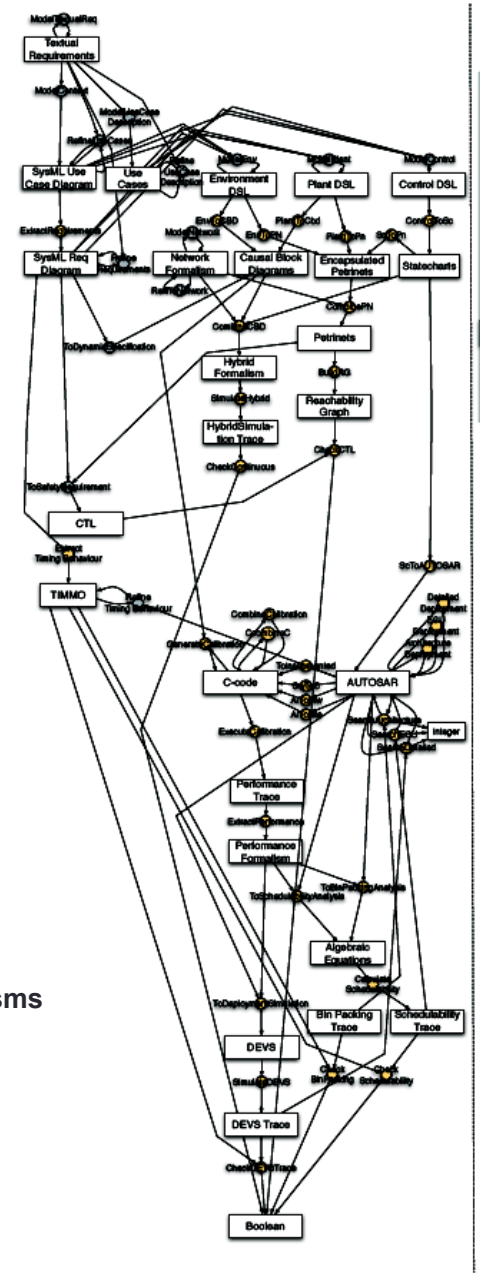


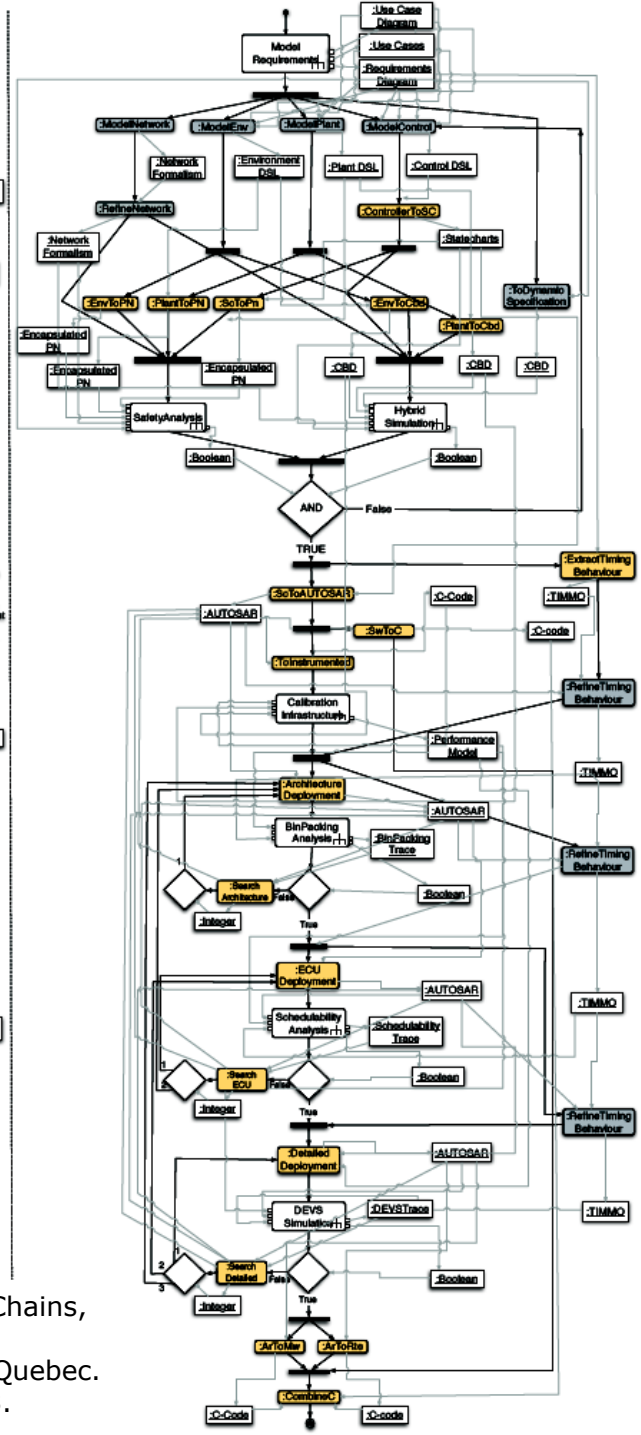
may use to reason (for a while) about abstraction “flock”



How to deal with **Complexity?**

(in engineered systems)



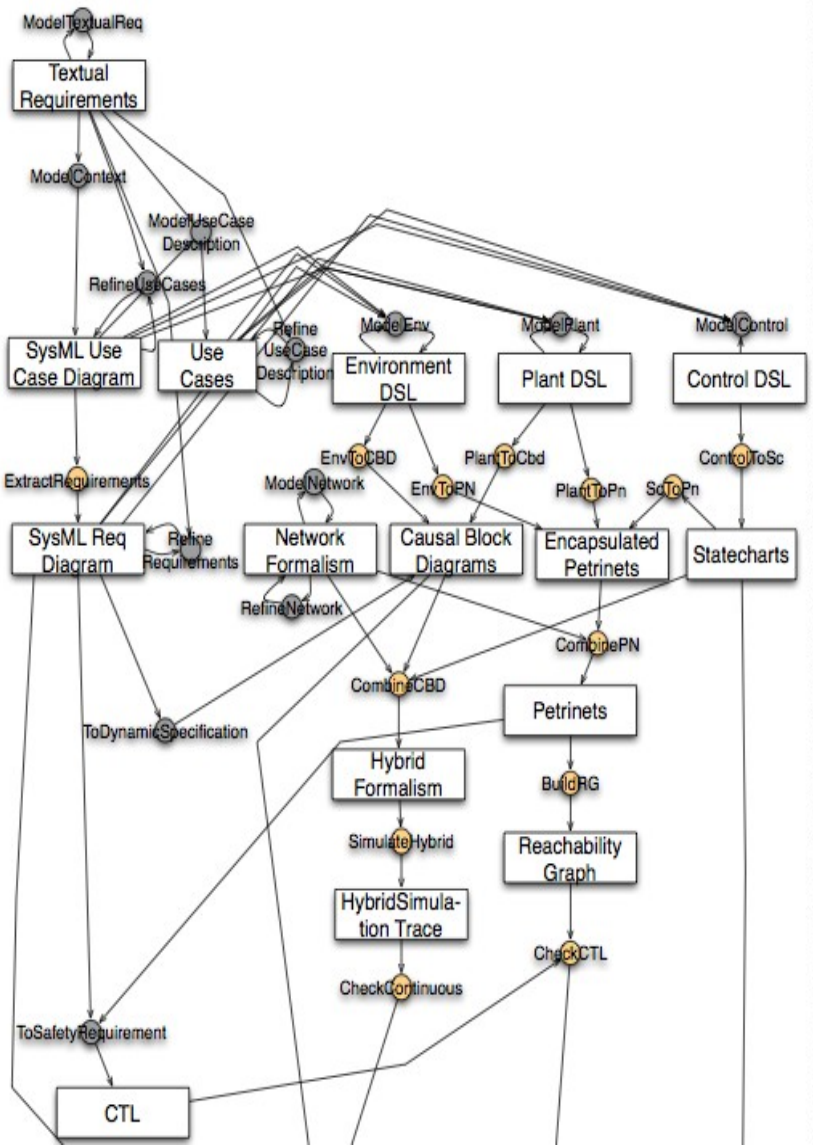


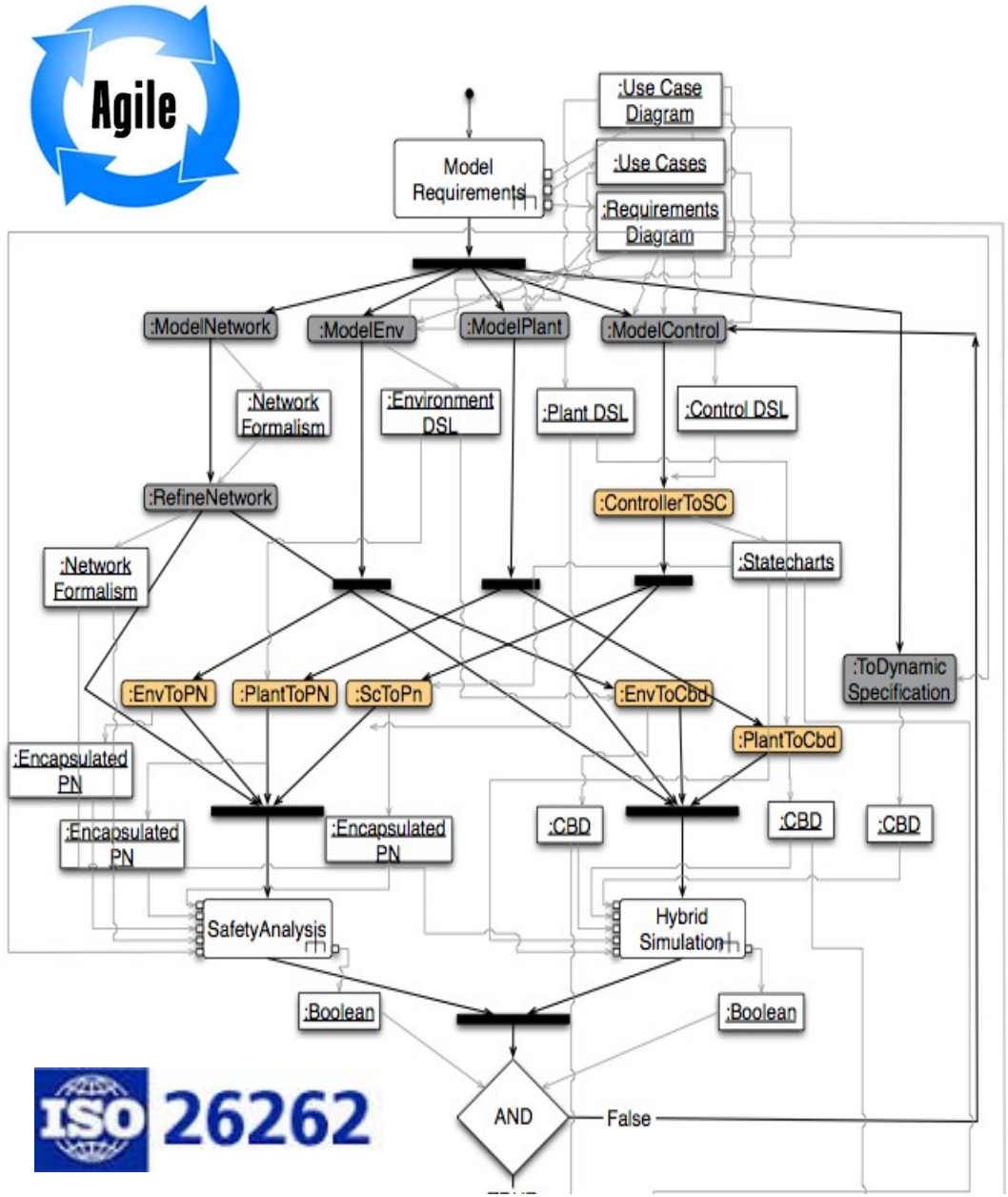
28 different **modelling formalisms**

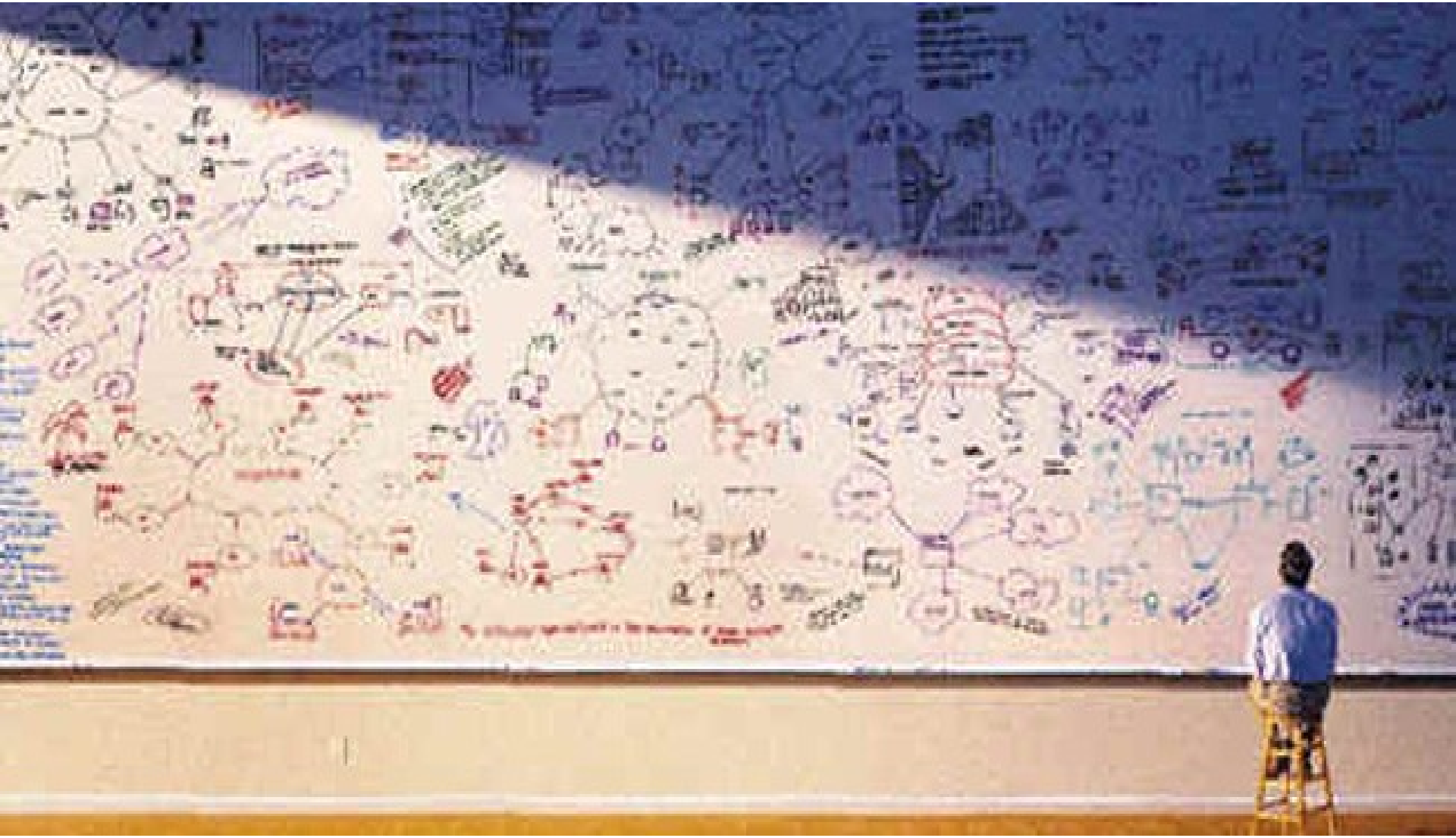
50 **transformations**

FTG+PM: An Integrated Framework for Investigating Model Transformation Chains,
Levi Lúcio, Sadaf Mustafiz, Joachim Denil, Hans Vangheluwe, Maris Jukss.
Proceedings of the System Design Languages Forum (SDL) 2013, Montreal, Quebec.
Lecture Notes in Computer Science (LNCS), Volume 7916, pp 182-202, 2013.

FTG+PM


ModelTextualReq
Textual Requirements
ModelContext
ModelUseCase Description
RefineUseCases
SysML Use Case Diagram
Use Cases
Refine UseCase Description
ModelEnv
Environment DSL
ModelPlant
Plant DSL
ModelControl
Control DSL
ExtractRequirements
EnvToCBD
PlantToCbd
ControlToSc
ModelNetwork
EnvToPN
PlantToPn
ScToPn
SysML Req Diagram
Refine Requirements
Network Formalism
Causal Block Diagrams
Encapsulated Petrinets
Statecharts
RefineNetwork
CombinePN
CombineCBD
Petrinets
ToDynamicSpecification
Hybrid Formalism
BuildRG
SimulateHybrid
Reachability Graph
HybridSimula-tion Trace
CheckCTL
CheckContinuous
ToSafetyRequirement
CTL

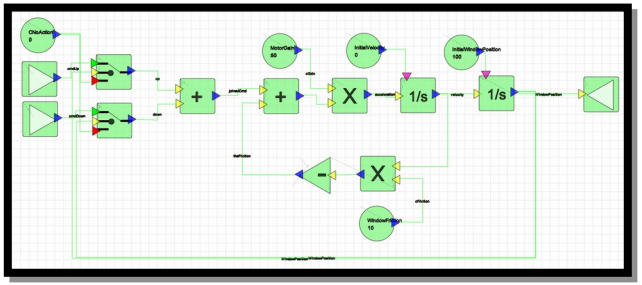

Agile
Model Requirements
:Use Case Diagram
:Use Cases
:Requirements Diagram
:ModelNetwork
:ModelEnv
:ModelPlant
:ModelControl
:Network Formalism
:Environment DSL
:Plant DSL
:Control DSL
:RefineNetwork
:ControllerToSC
:Statecharts
:Network Formalism
:EnvToPN
:PlantToPN
:ScToPn
:EnvToCbd
:ToDynamic Specification
:PlantToCbd
:Encapsulated PN
:Encapsulated PN
:Encapsulated PN
:CBD
:CBD
:CBD
SafetyAnalysis
Hybrid Simulation
:Boolean
:Boolean
AND
False
ISO 26262

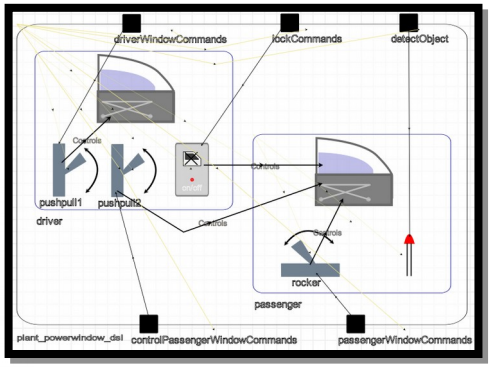


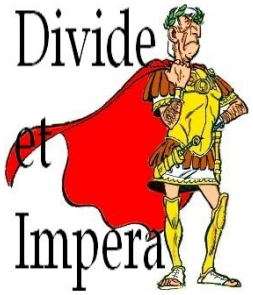
How to deal with **Complexity?**

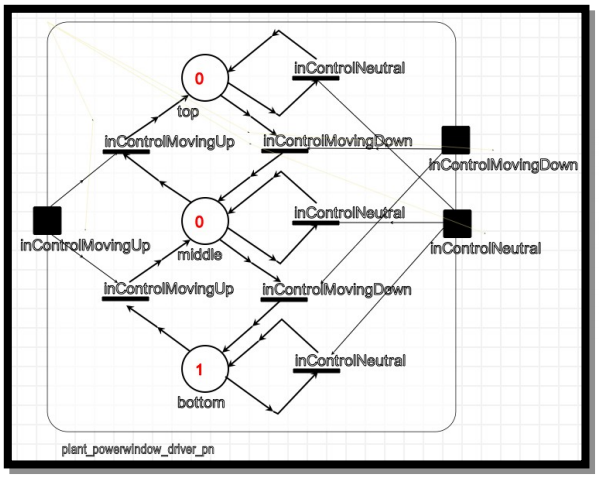
(in engineered systems)

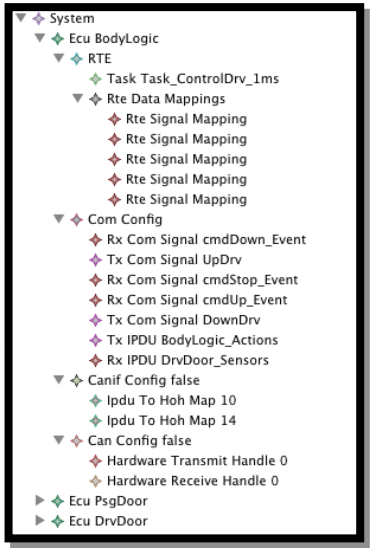
multi-view modelling

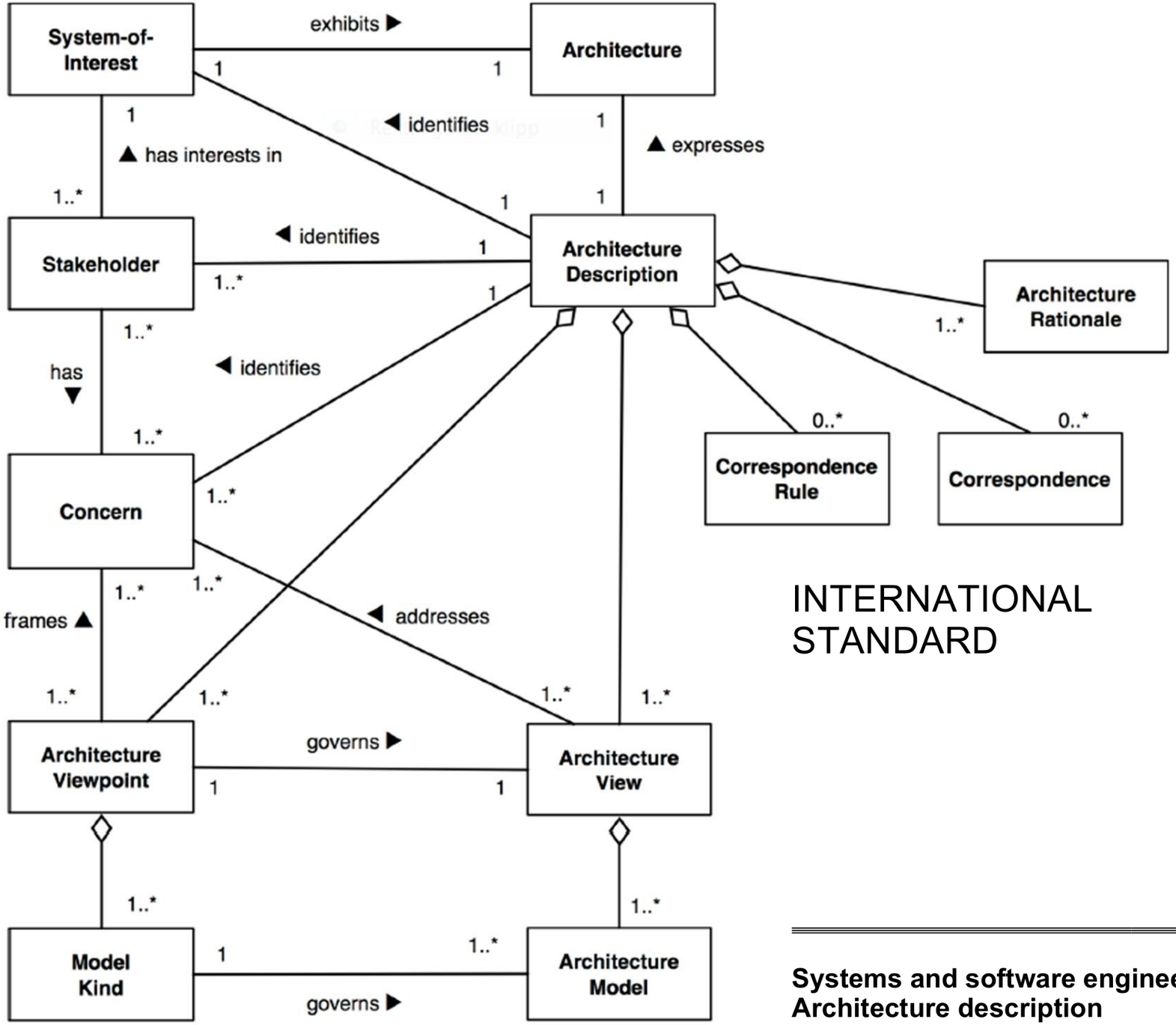












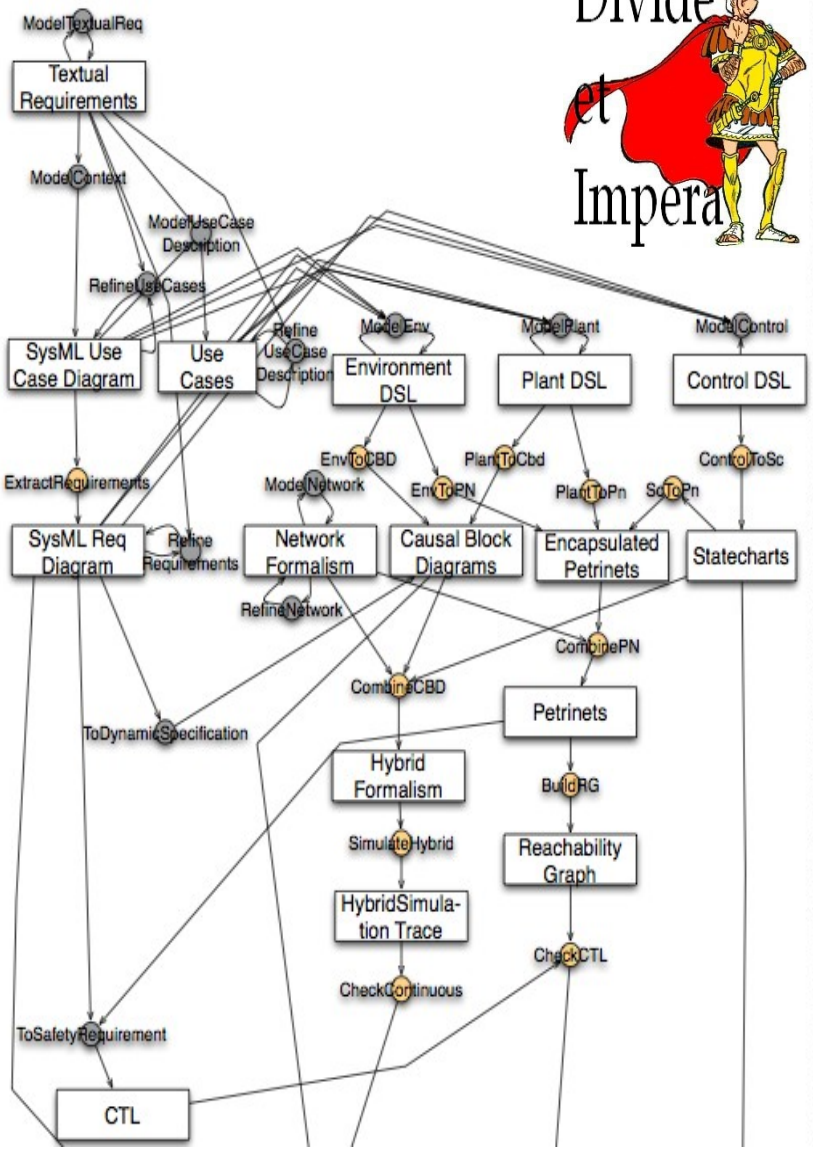
INTERNATIONAL STANDARD

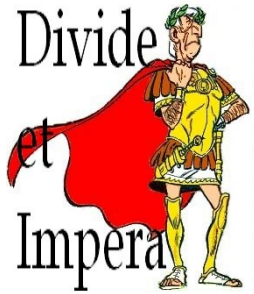
ISO/IEC/ IEEE 42010

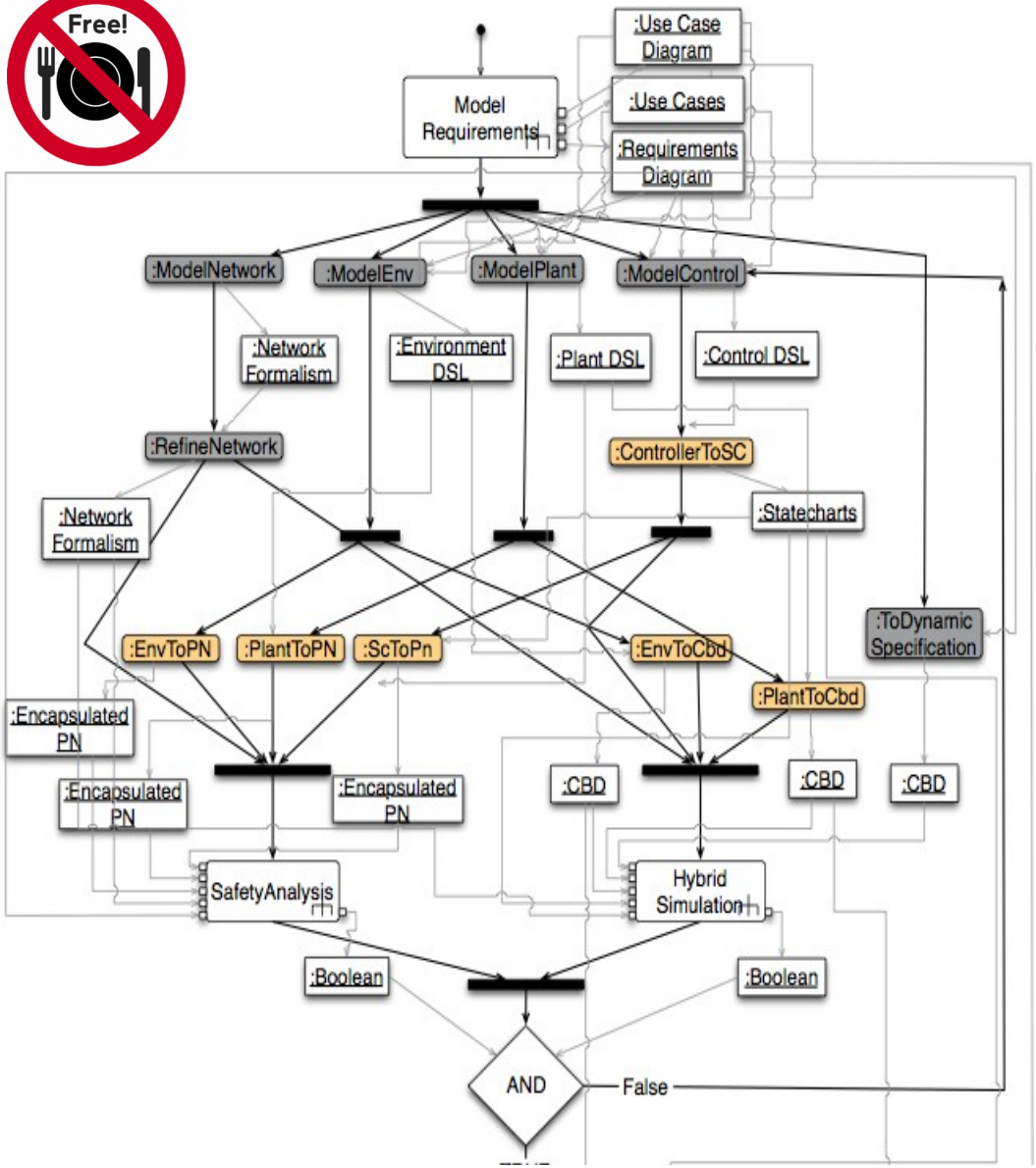
First edition
2011-12-01

Systems and software engineering — Architecture description

problem: multi-view/concurrent engineering leads to **incorrect** results



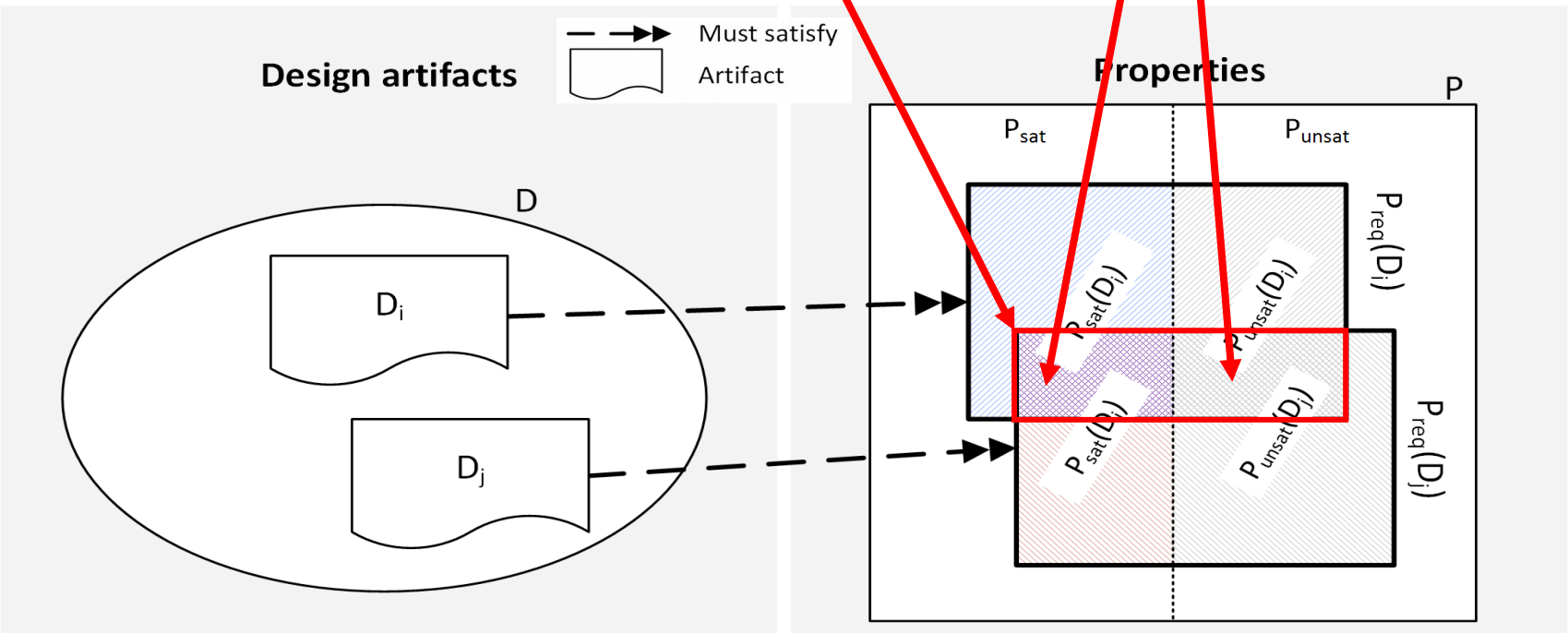




correctness vs. (in-)**consistency**

The design artifacts D_i and D_j are said to be consistent w.r.t. the set of properties $P' \subseteq P \equiv P_{req}(D_i) \bigcap P_{req}(D_j)$ iff $\forall p \in P' : D_i \vDash p \Leftrightarrow D_j \vDash p$.

	$D_i \vDash P'$	$D_j \vDash P'$	Consistent	Correct
(1)	✓	✓	✓	?
(2)	✓	×	×	×
(3)	×	✓	×	×
(4)	×	×	✓	×

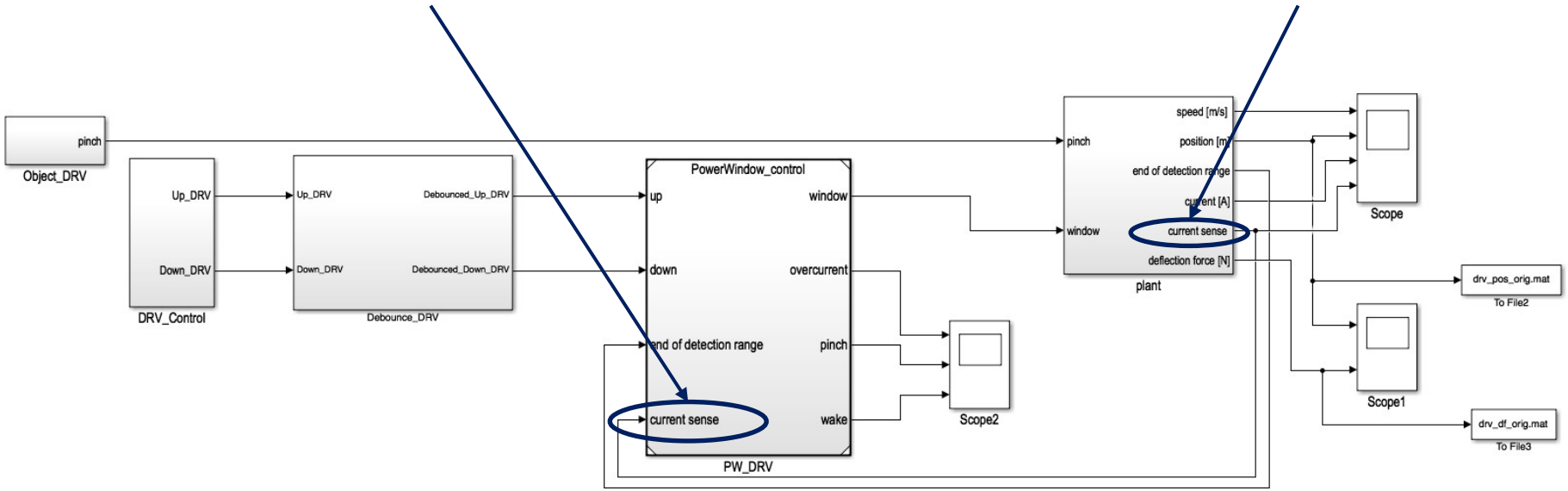


I. Dávid. Inconsistency Management in Model-Based Systems Engineering. PhD. thesis. University of Antwerp. May 2019.

Horizontal (In)Consistency – Example

Horizontal inconsistency

Between 0 and 20 A ⟷ Between 0 and 10 A

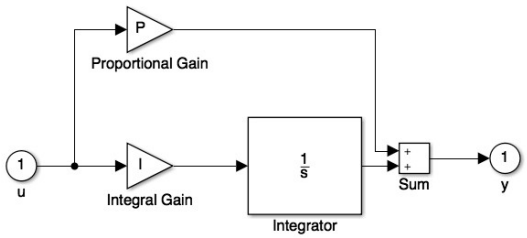


Control Engineer

Mechanical Engineer

Vertical (In)Consistency – Example

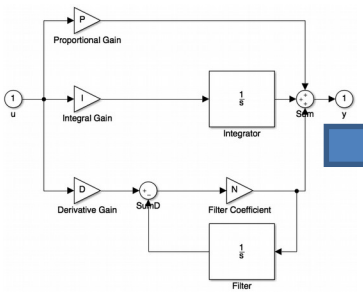
Settling time low

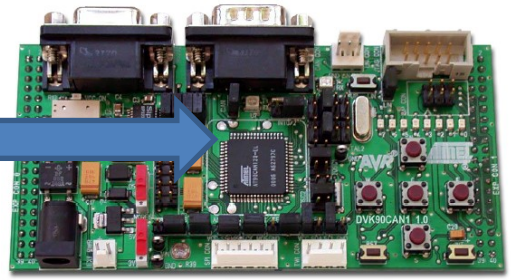


Control Engineer

Vertical inconsistency

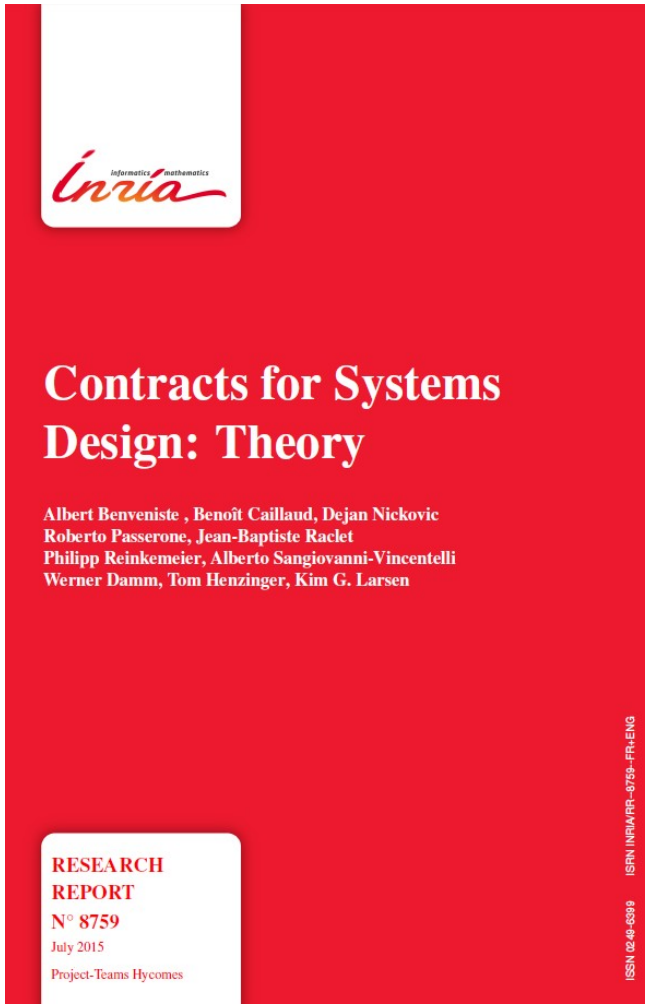
Sample Frequency low



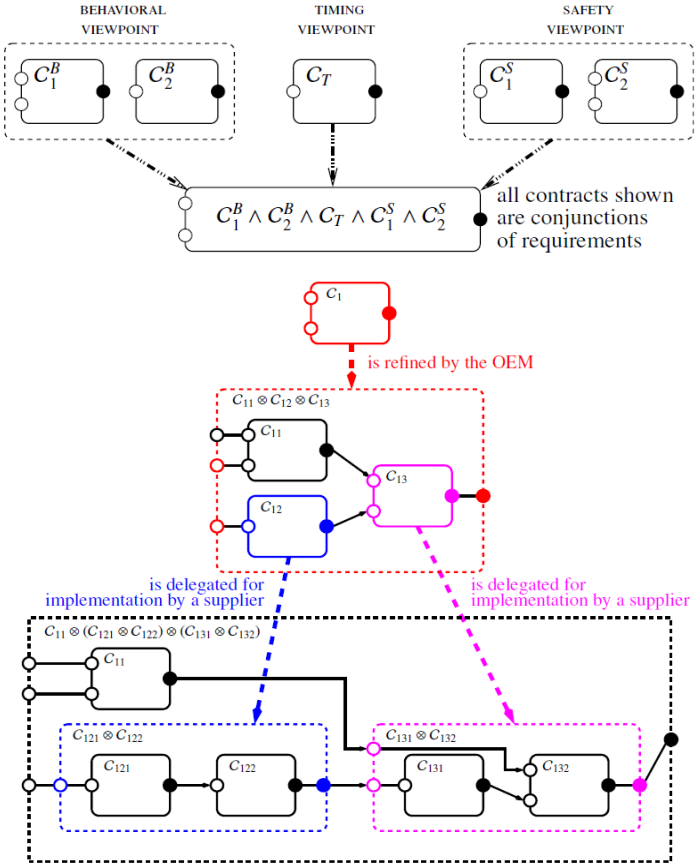


Embedded Engineer

solution 1: (in-)consistency management prevention (**contract-based design**)



guarantees offered by the component
assumptions on its possible context



K. Vanherpen. A contract-based approach for multi-viewpoint consistency in the concurrent design of cyber-physical systems. PhD Thesis. University of Antwerp. 2018.

Inconsistency management: **avoid** through “contracts”

Software World: Eiffel Programming Language (late 80’s):

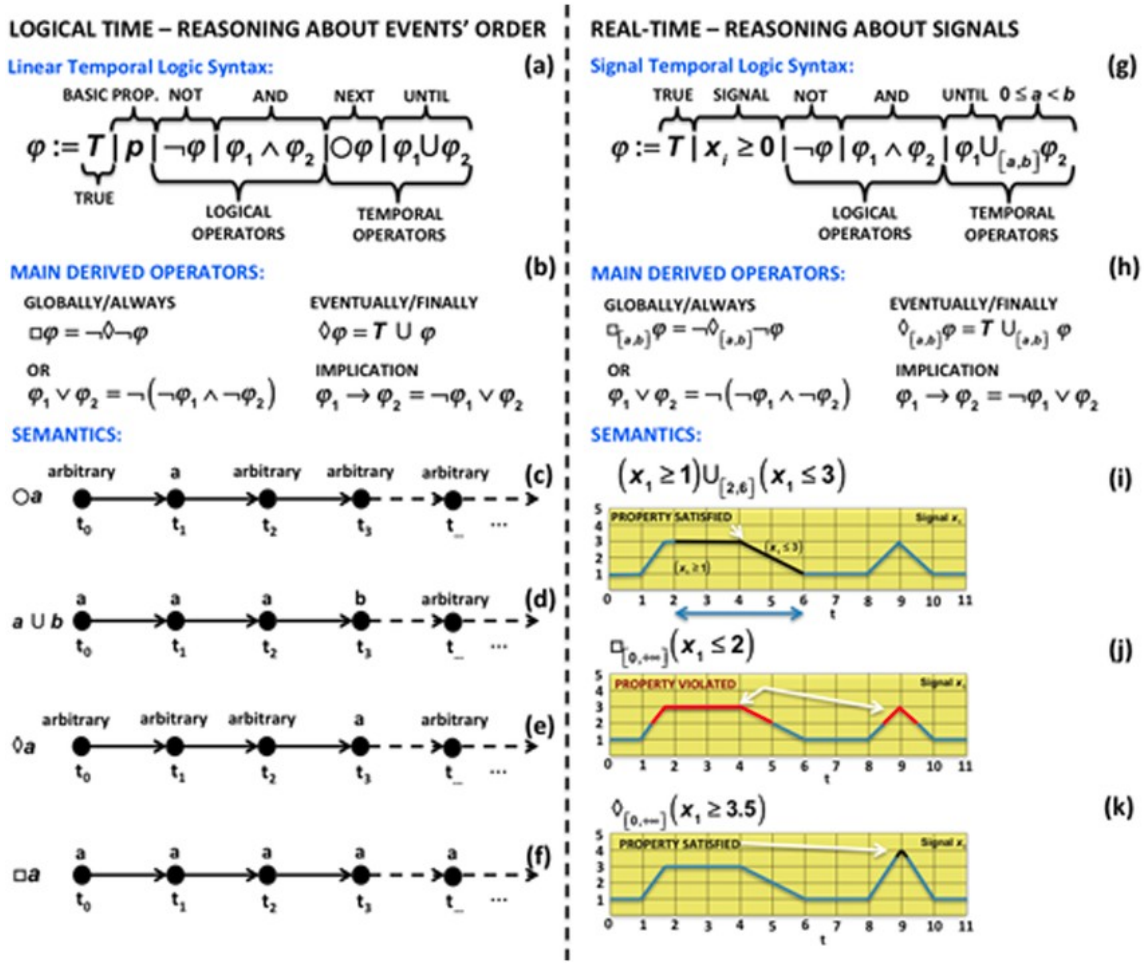
```
set_integer_range (a_min, a_max: INTEGER)
        -- Set range with `a_min' and `a_max' values.
    require
        min_less_than_or_equal_to_max: a_min <= a_max
        min_over_floor: a_min >= min_floor
        max_under_ceiling: a_max <= max_ceiling
    do
        integer_range := [a_min, a_max]
    ensure
        min_set: integer_range.min = a_min
        max_set: integer_range.max = a_max
    end
```

B. Meyer, “Eiffel: A language and Environment for Software Engineering,” Journal of Systems and Software, vol. 8, no. 3, pp. 199 – 246, 1988.

B. Meyer, “Applying ’**Design by Contract**’,” Computer, vol. 25, pp. 40–51, Oct. 1992.

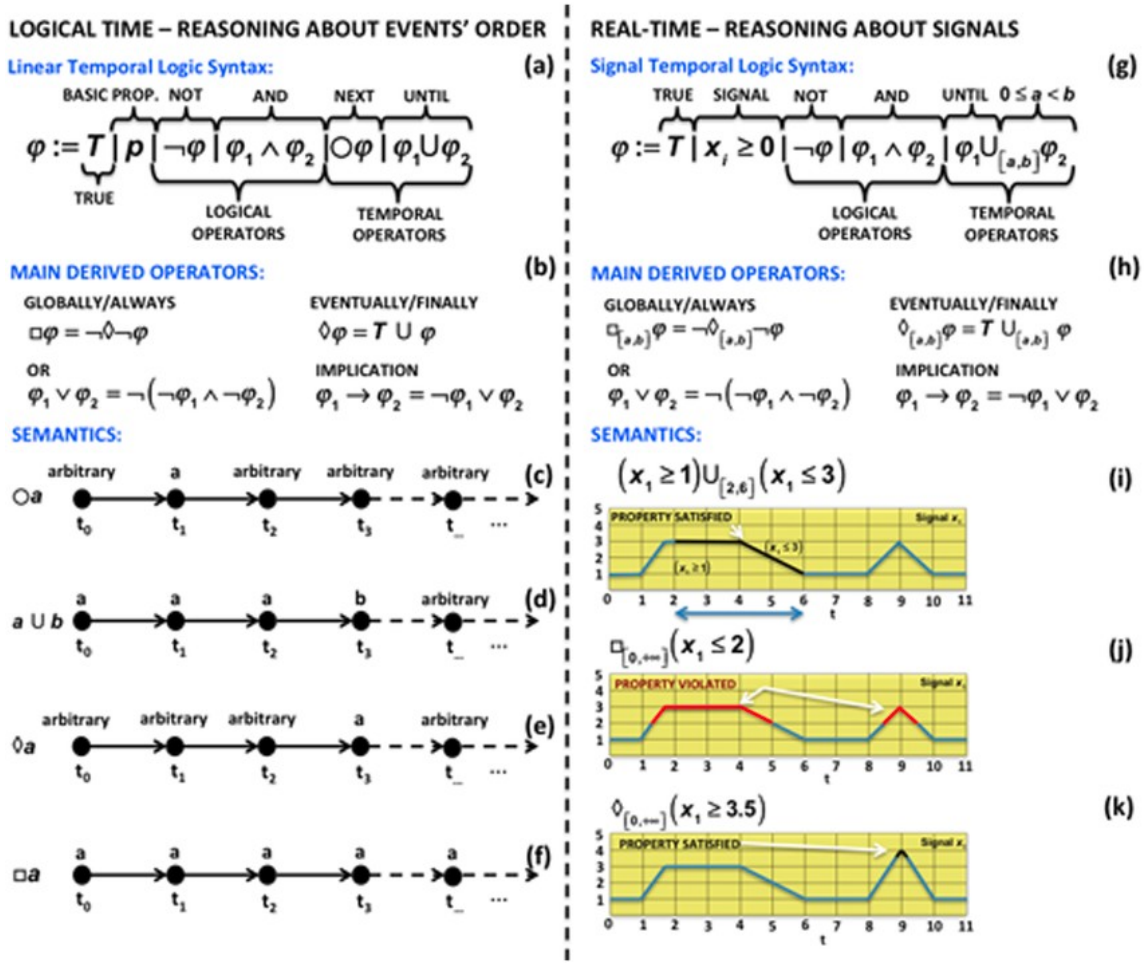
Inconsistency management: **avoid** through "contracts"

Caveat: need appropriate (domain-specific) language to describe temporal patterns



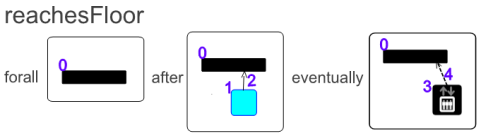
Inconsistency management: **avoid** through “contracts”

Caveat: need appropriate (domain-specific) language to describe temporal patterns

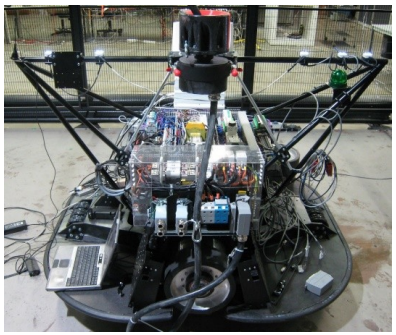


$$\Box(((go0 \wedge up0) \vee \Diamond(floor0 \vee idle)) \rightarrow ((\neg(floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge (((floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge ((\neg(floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge (((floor0) \vee \neg(floor0 \vee idle))\mathcal{U}((floor0 \vee idle) \wedge (\neg(floor0)\mathcal{U}(floor0 \vee idle)))))))))) \vee \Box(((go1 \wedge up1 \wedge down1) \vee \Diamond(floor1 \vee idle)) \rightarrow ((\neg(floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge (((floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge ((\neg(floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge (((floor1) \vee \neg(floor1 \vee idle))\mathcal{U}((floor1 \vee idle) \wedge (\neg(floor1)\mathcal{U}(floor1 \vee idle)))))))))) \vee \Box(((go2 \wedge down2) \vee \Diamond(floor2 \vee idle)) \rightarrow ((\neg(floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge (((floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge ((\neg(floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge (((floor2) \vee \neg(floor2 \vee idle))\mathcal{U}((floor2 \vee idle) \wedge (\neg(floor2)\mathcal{U}(floor2 \vee idle)))))))))))$$

vs.



solution 2: (in-)consistency management
detect-and-resolve

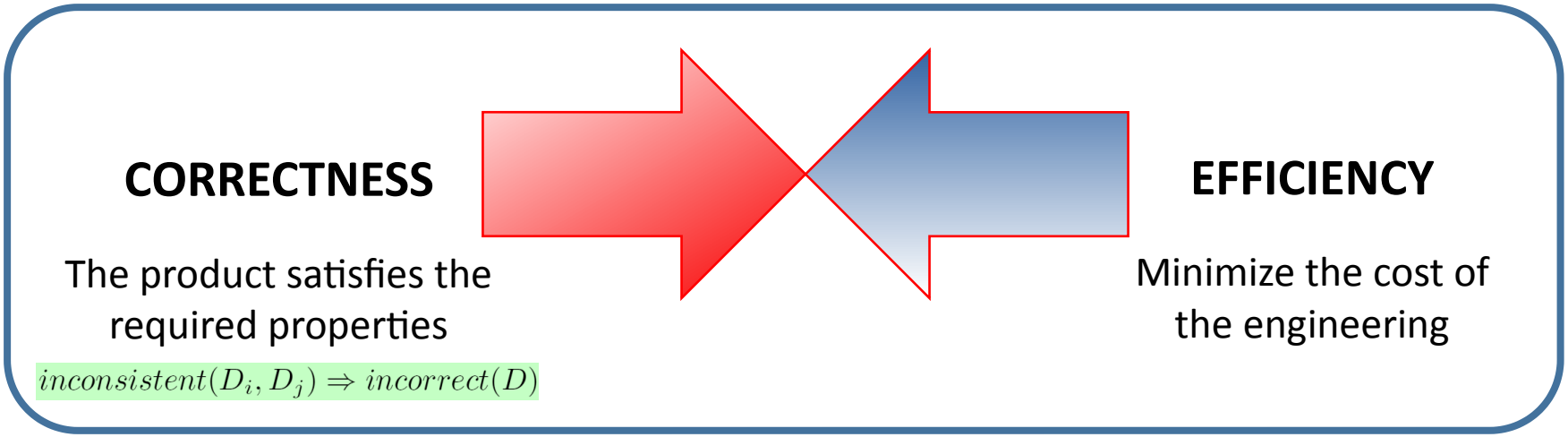


I. Dávid. Inconsistency Management in Model-Based Systems Engineering. PhD. thesis. University of Antwerp. May 2019.

problem : sub-**optimal time**-to-market
solution : **process optimization**
caveat : correctness of product

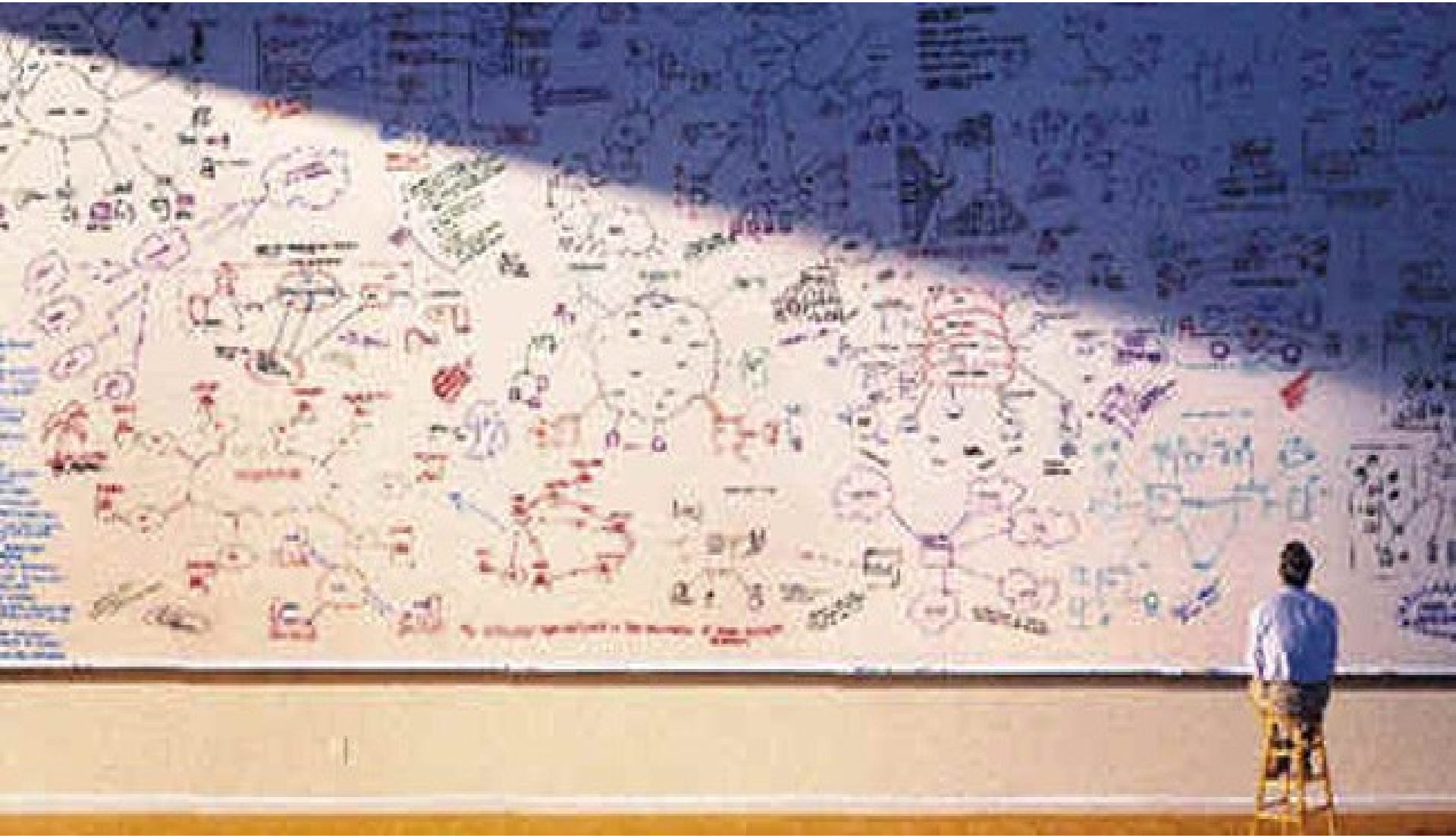
optimize so that (i) a correct product is delivered; and (ii) the process is as efficient as possible.

PROCESS



Istvan David, Joachim Denil, and Hans Vangheluwe. Process-oriented inconsistency management in collaborative systems modeling. In Proceedings of the 2018 Industrial Simulation Conference (ISC), pages 54 -- 61. Eurosis – ETI. June 2018.

I. Dávid, Y. Van Tendeloo, H. Vangheluwe. Translating Engineering Workflow Models to DEVS for Performance Evaluation. In Proceedings of the 2018 Winter Simulation Conference, pp. 616-627. IEEE Press, 2018.

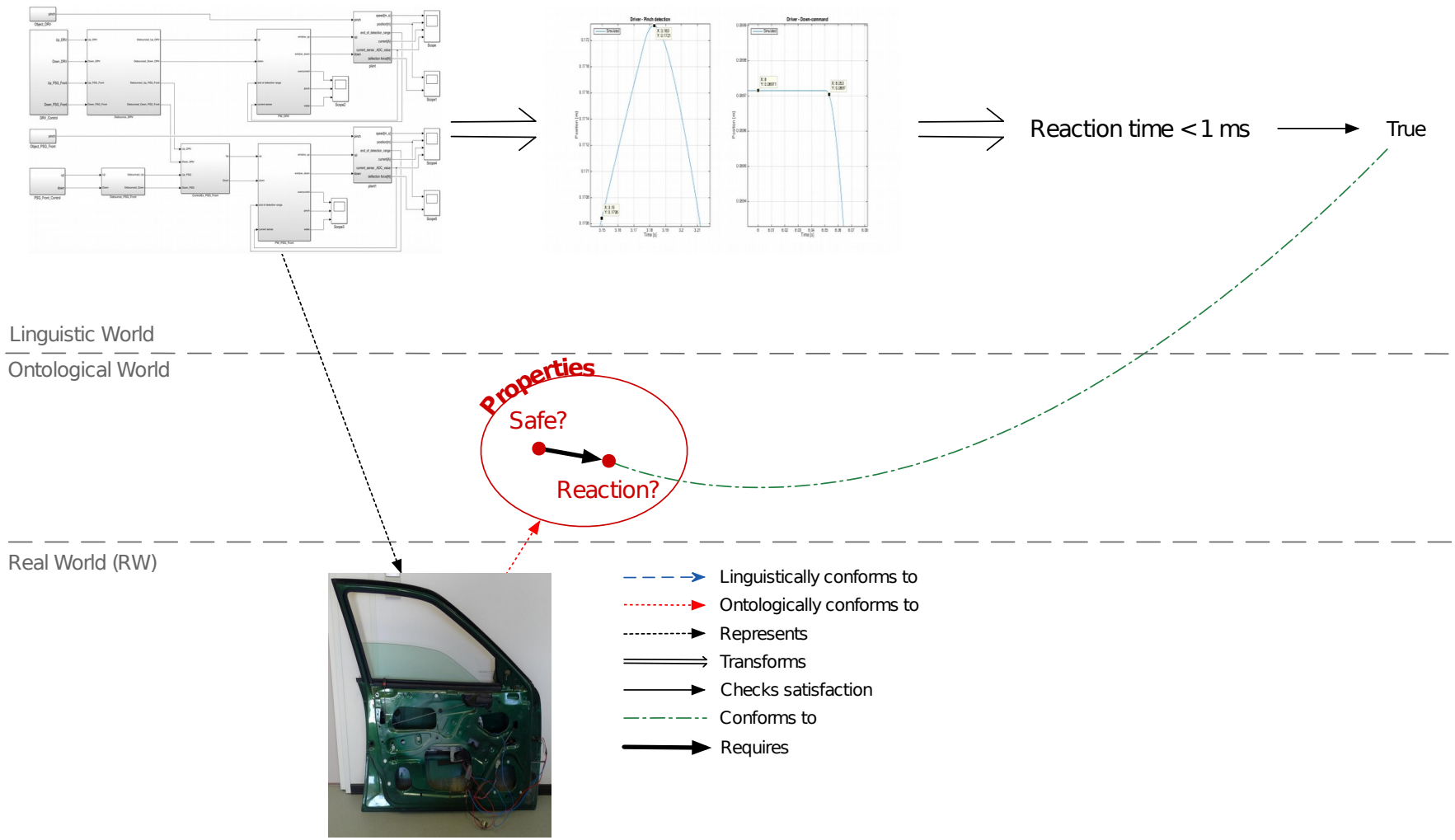


How to deal with **Complexity?**

(in engineered systems)

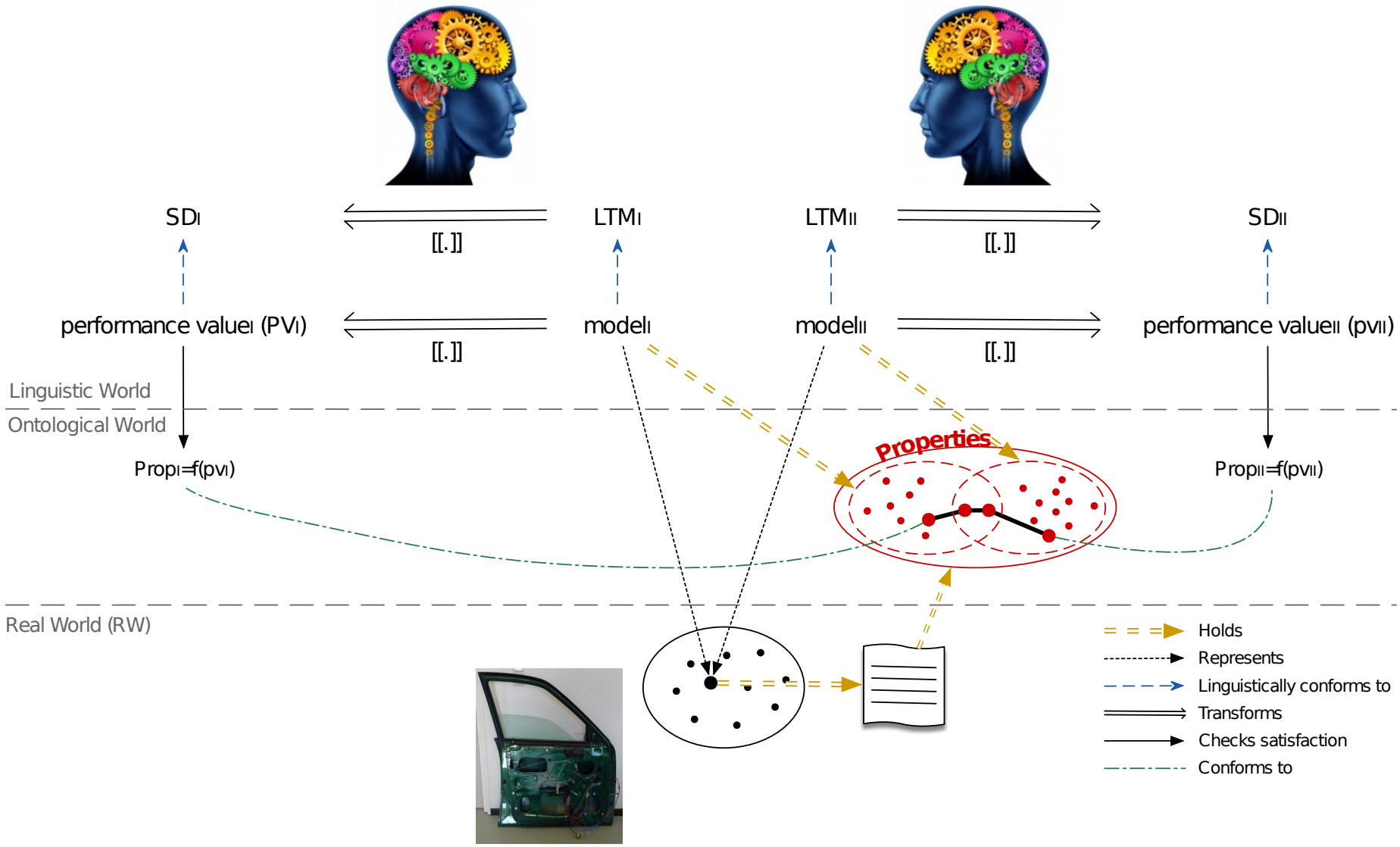
problem: stakeholders do not understand each other

solution: **Linguistic** vs. **Ontological** Reasoning in MBSE



B. Barroca, T. Kühne, and H. Vangheluwe. Integrating language and ontology engineering. In MPM '14, volume 1237 of CEUR, pages 77–86, September 2014.

solution: **Linguistic** vs. **Ontological** Reasoning in MBSE

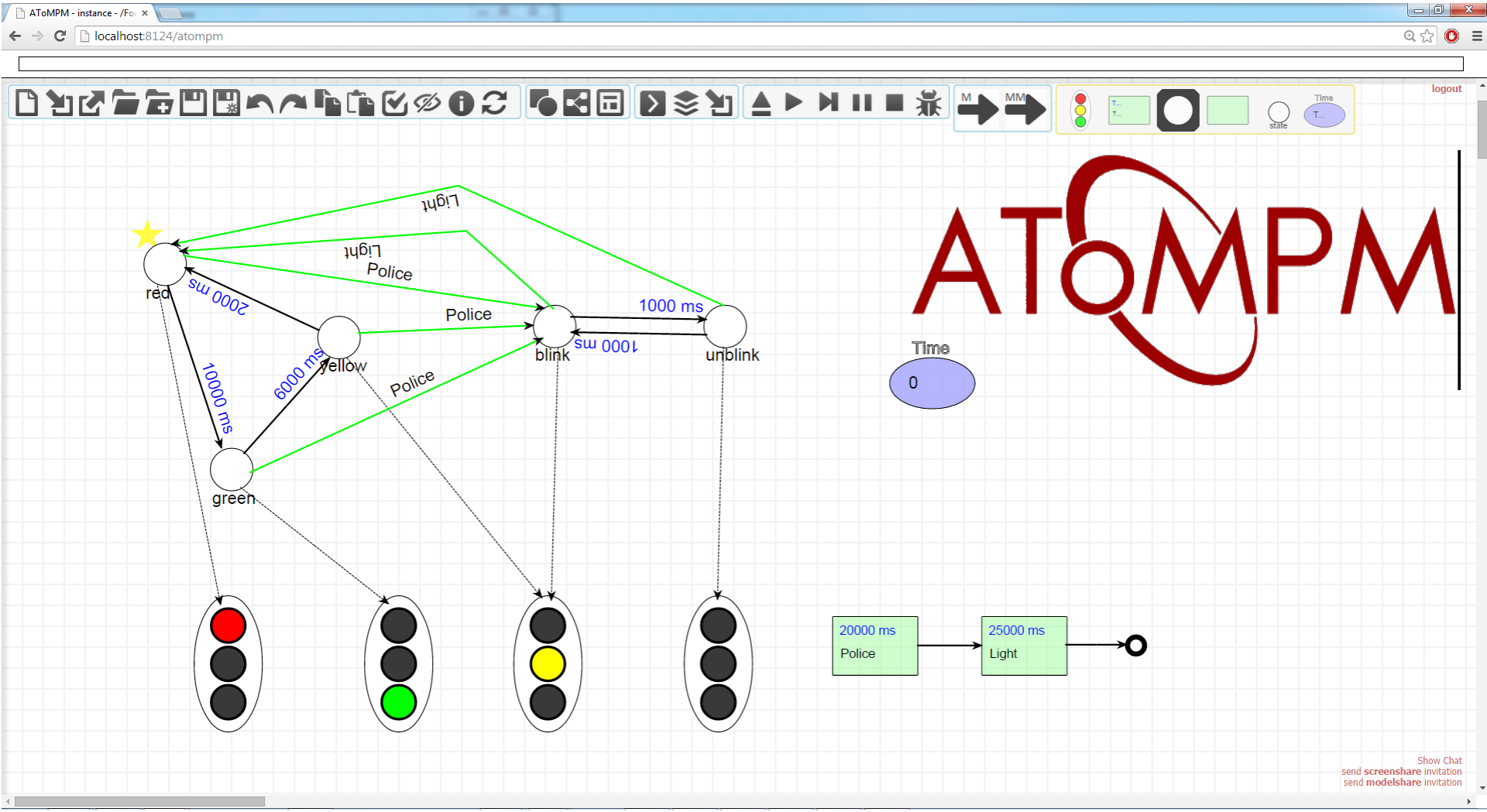


Ontological Reasoning for Consistency in the Design of Cyber-Physical Systems.
K. Vanherpen, J. Denil, I. Dávid, P. De Meulenaere, P. J. Mosterman, M. Törngren, A. Qamar, H. Vangheluwe.
CPPS 2016 - 1st International Workshop on Cyber-Physical Production Systems, 2016.

Enabler/Foundation

Modelling Language Engineering

(domain-specific) “linguistic” Modelling Language Engineering

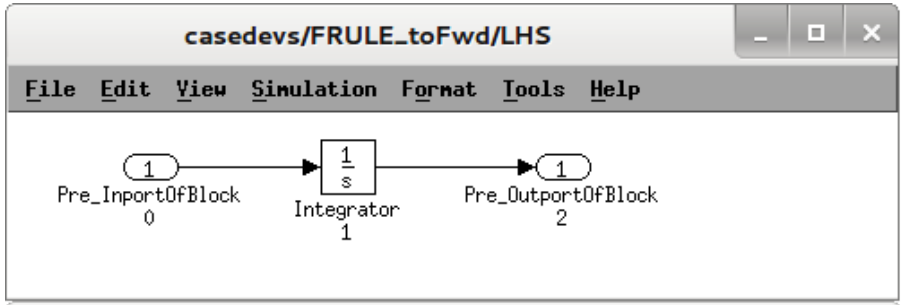


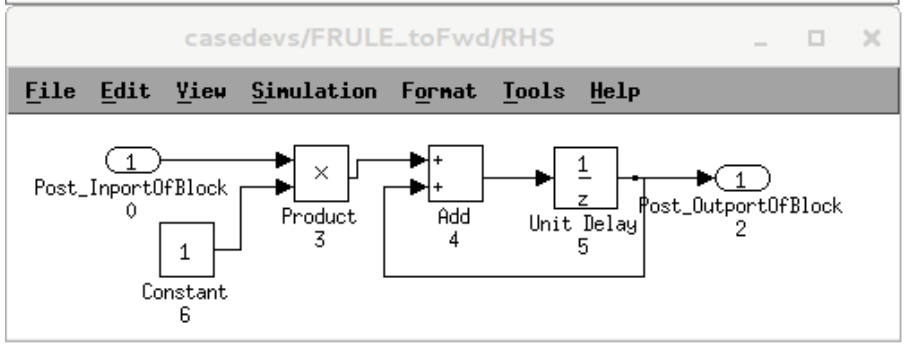
Raphael Mannadiar. A Multi-Paradigm Modelling Approach to the Foundations of Domain-Specific Modelling. PhD thesis, McGill Univ., 2012.

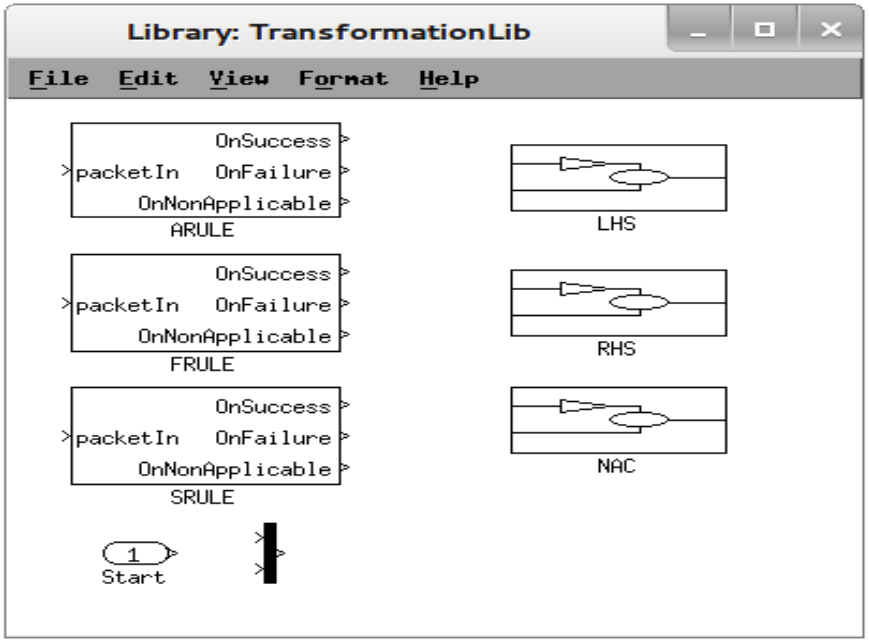
Eugene Syriani, Hans Vangheluwe, Raphael Mannadiar, Conner Hansen, Simon Van Mierlo, and Huseyin Ergin. AToMPM: A web-based modeling environment. In Proceedings of MODELS'13 Demonstration Session co-located with the 16th International Conference on Model Driven Engineering Languages and Systems (MODELS 2013), Miami, USA, pages 21- 25, 2013.

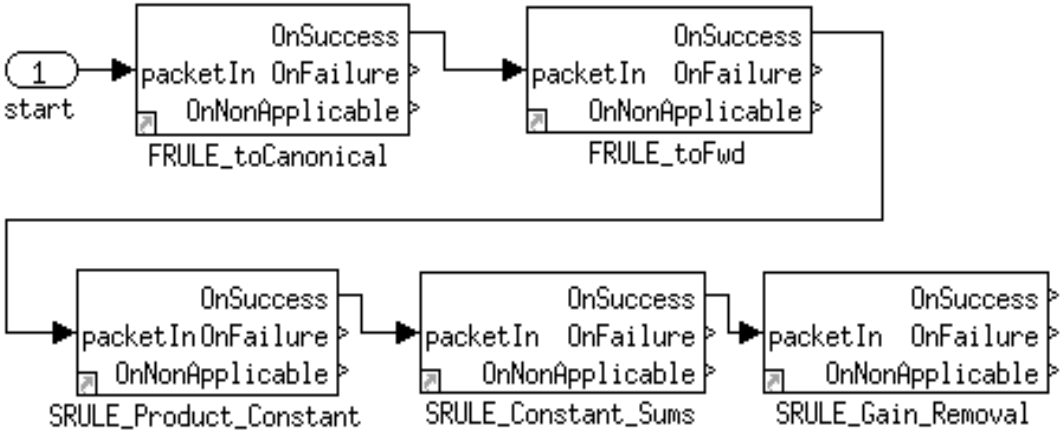
https://www.youtube.com/watch?feature=player_detailpage&v=RYtea2BiQ98

Transformations for/in Simulink





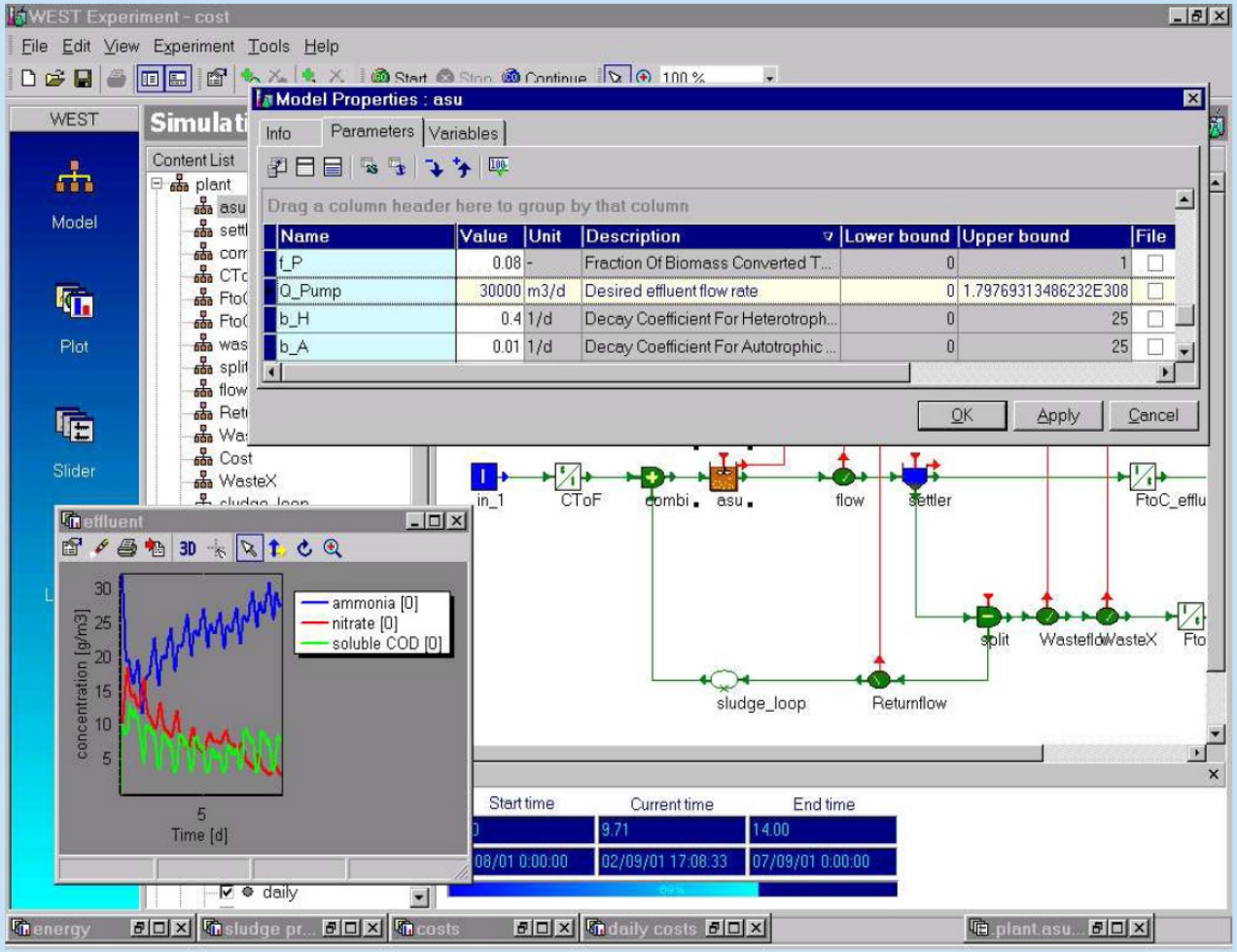




Joachim Denil, Pieter J. Mosterman, and Hans Vangheluwe. Rule-Based Model Transformation For, and In Simulink. In Proceedings of the Spring Simulation Multi-Conference, pages 314 - 321. SCS, April 2014. Overall best paper award of the SpringSim multi-conference .

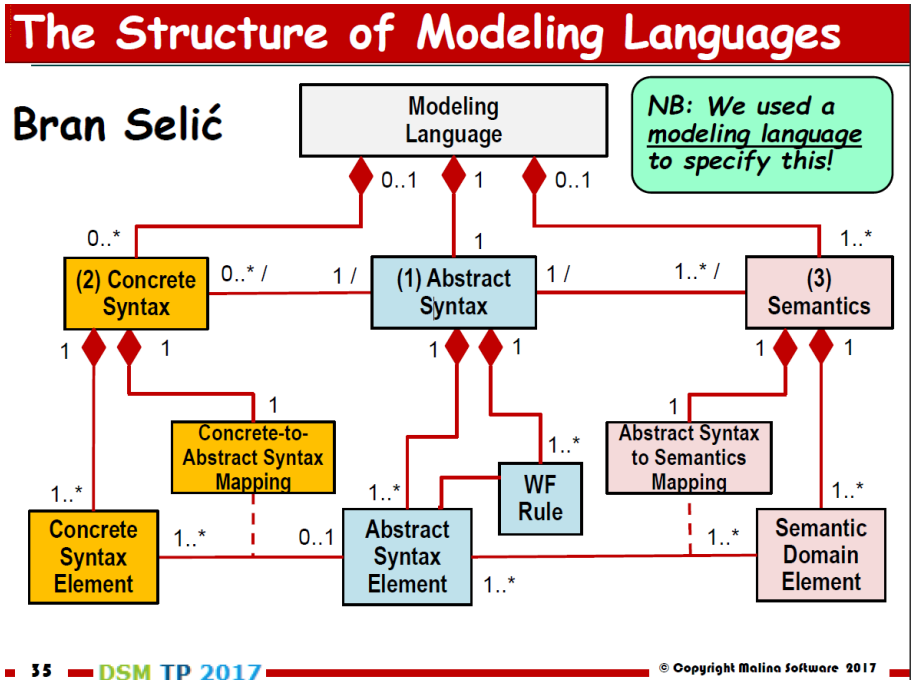
Bentley Oakes: optimizing Simulink models, reverse-engineering legacy simulation code to Simulink (with LLVM). Tech. Report.

DS(V)M Environment

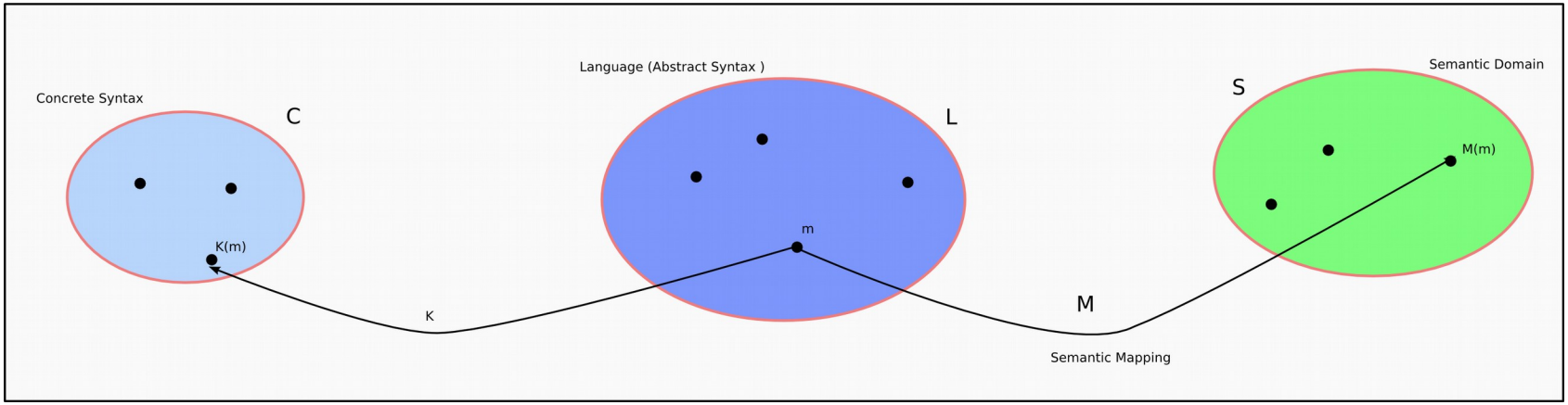


WEST: modelling biological wastewater treatment.
Henk Vanhooren, Jurgen Meirlaen, Youri Amerlinck, Filip Claeys, Hans Vangheluwe and Peter A.Vanrolleghem.
Journal of Hydroinformatics 5 (2003) 27-50

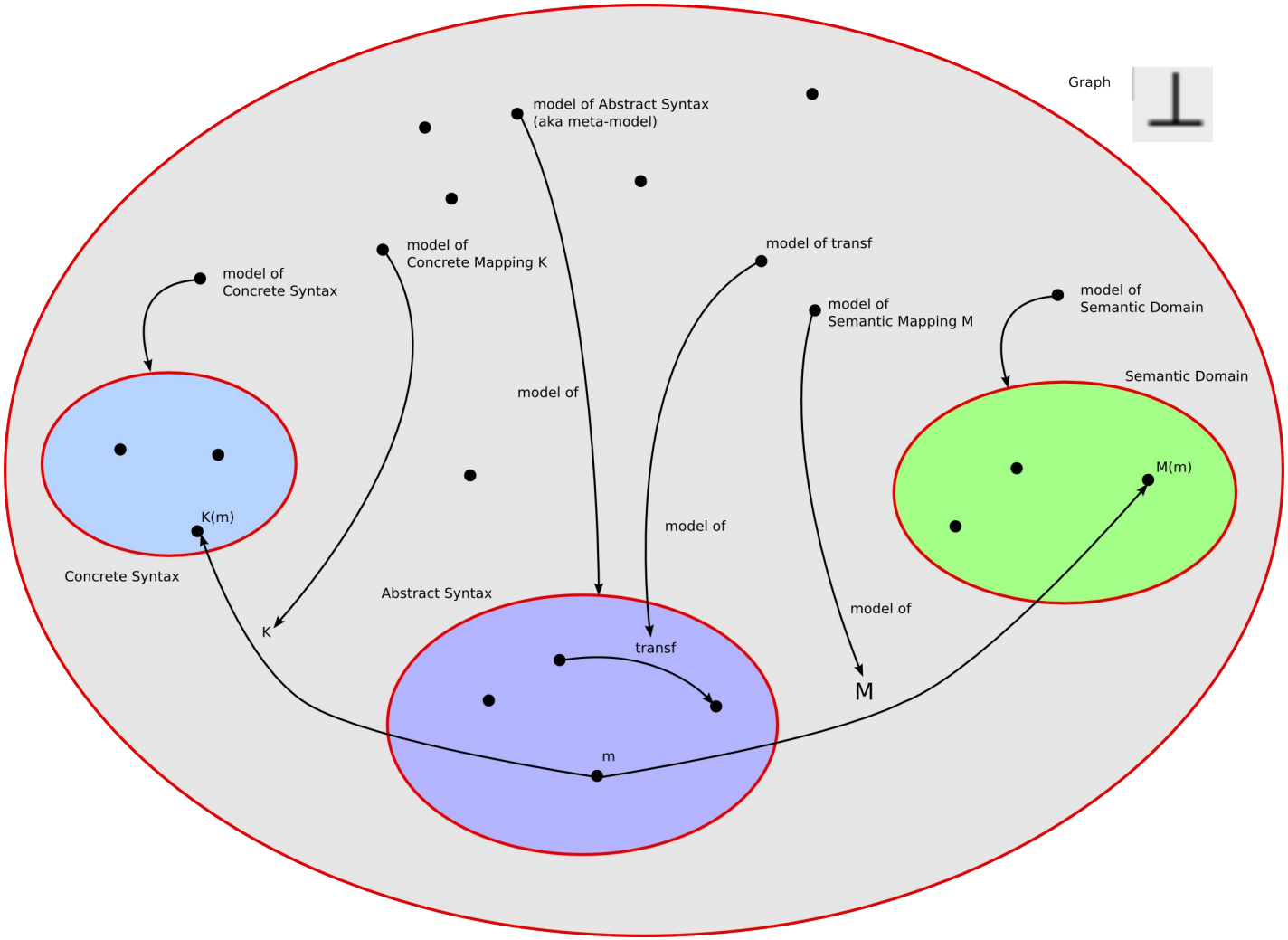
Modelling Languages/Formalisms
Syntax and Semantics

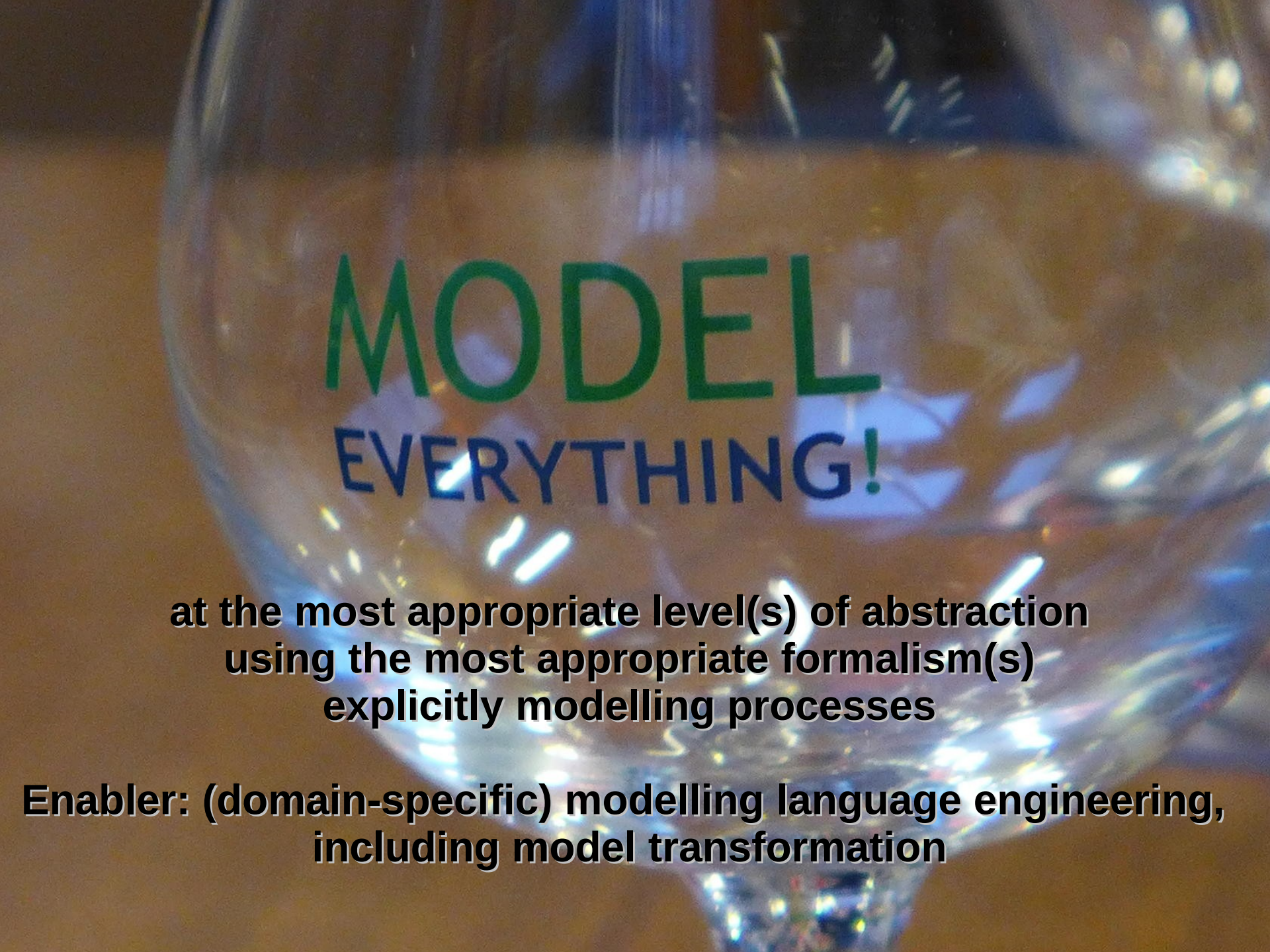


Concrete Formalism F



Explicit "linguistic" Modelling of Modelling Languages/Formalisms





at the most appropriate level(s) of abstraction
using the most appropriate formalism(s)
explicitly modelling processes

Enabler: (domain-specific) modelling language engineering, including model transformation