Nordic Systems Engineering Tour 2019

18/09/2019 Oslo

S2ML+X in Pedagogical Action

Prof. Antoine B. Rauzy

Department of Mechanical and Industrial Engineering
Norwegian University of Science and Technology
Trondheim, Norway

&

Chair Blériot-Fabre
CentraleSupélec
Paris, France

Teaching Model-Based Systems Engineering

Targeted audience:

- Students in engineering studies, master degree (mechanical engineering, reliability engineering…)

Targeted systems:

- Engineered technical systems (cars, trains, power plants, engines…)

  "A combination of interacting elements that are organized so to achieve one or several established objectives" (INCOSE)

Course:

- 1 semester, ~30 hours of course + ~20 hours of tutorial
- 20-50 students

Toward
a general theory of models
in systems engineering

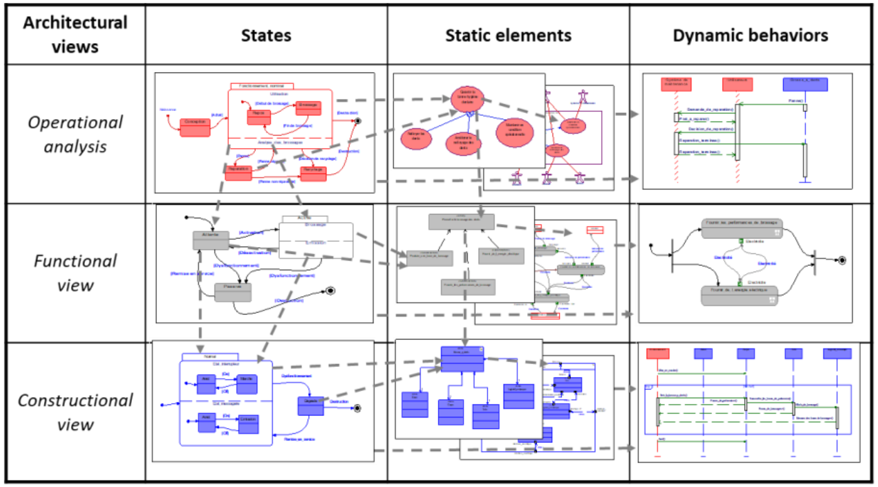
Objectives (examples)

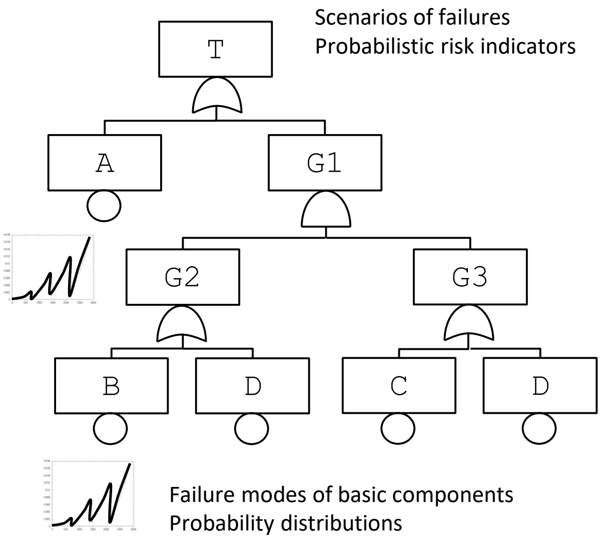
System Architecture

Reliability Engineering

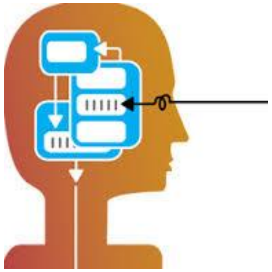
What the system should do?
What the system should be?

What can go wrong?
What is the severity of consequences?
What is the likelihood?

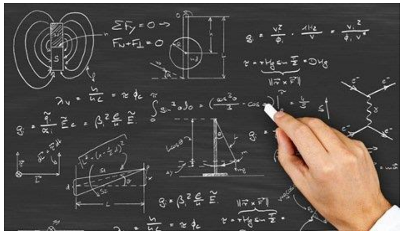




Which Models?



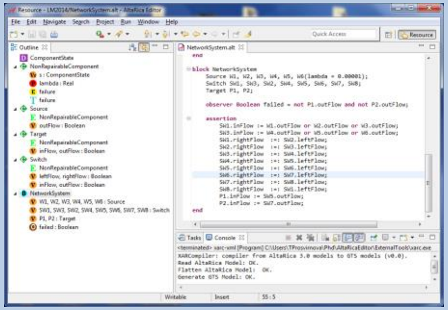
Cognitive Model



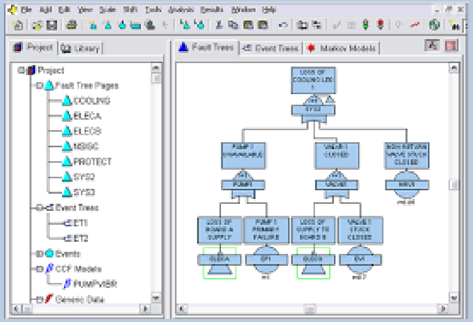
Mathematical Model

mind & paper models

computerized models



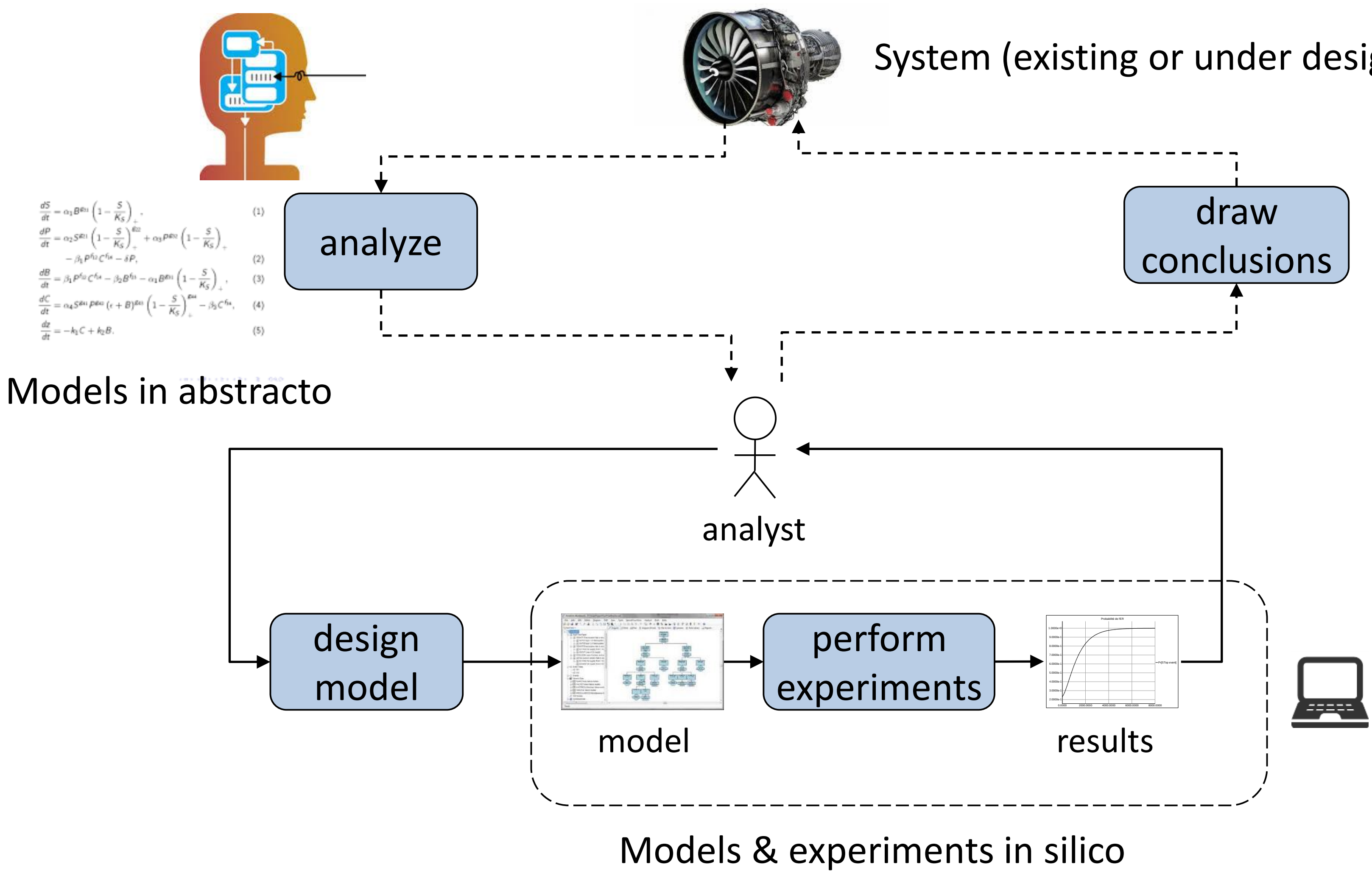
Text



Diagrams

Models are **working tools**, not (platonic) ideals the system should comply to.

Virtual experiments



What is a Model?

Models (textual or graphical) are eventually **sequences of symbols**.

To deserve the **status of model**, such a sequence of symbols must:

1 Have a well-defined **syntax**

There should be a **grammar**, i.e. a set of rules, that makes it possible to **verify mechanically** which sequences of symbols are models and which are not.

2 Have a well-defined **semantics**

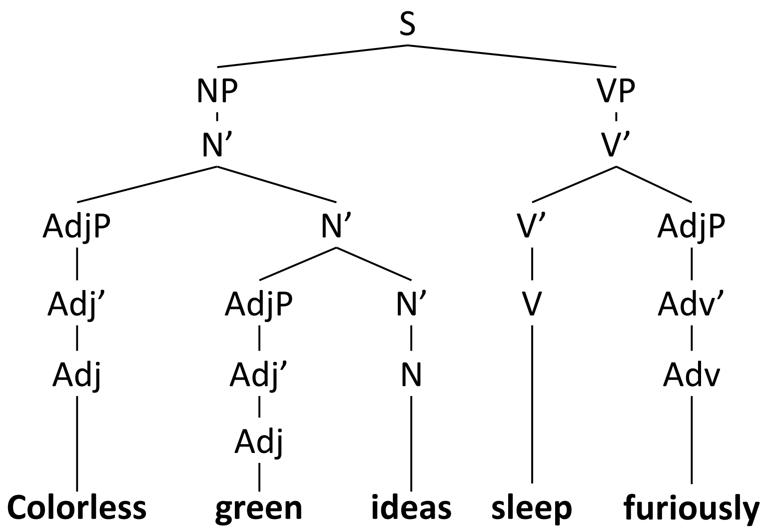
There should be an **interpretation**, i.e. a set of rules, that **associates without ambiguity** to each model a **mathematic object** belonging to a certain **algebra** (collection objects + set of operators operating on these objects)

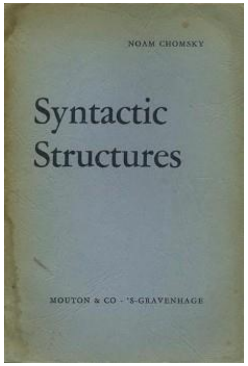
More of less standardized notations are not models… which does mean that they are not useful to support the communication among stakeholders.

Pragmatics

In addition to their syntax and their semantics, models have a **pragmatics**, i.e. relation to the system they represent.

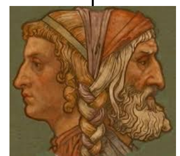
Pragmatics is a subfield of linguistics and semiotics that studies the ways in which **context** contributes to **meaning**.





Noam Chomsky (1957)

Pragmatic versus Formal Models

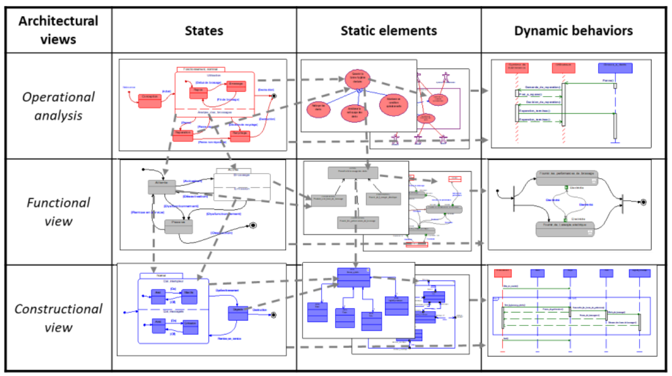


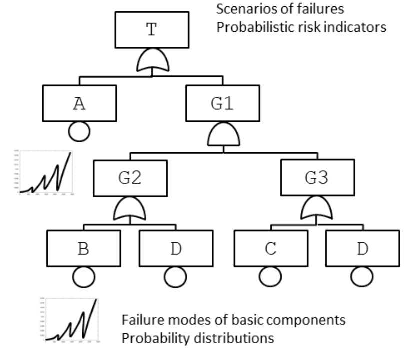
System Architecture

Reliability Engineering

Models to communicate
amongst stakeholders

Models to calculate
performance indicators





Descriptive/pragmatic models

Predictive/formal models

Epistemic gap

The S2ML+X Paradigm

Constituents of Models

Behaviors + Structures = Models*

Meaning and practical consequences:

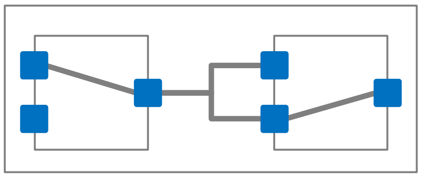
- Any modeling language is the combination of a **mathematical framework** to describe the behavior of the system under study and a **structuring paradigm** to organize the model.
- The choice of the **appropriate mathematical framework** for a model depends on which aspect of the system we want to study
- **Structuring paradigms** are to a very large extent **independent** of the chosen mathematical framework. They can be studied on their own.

(*) In reference to Wirth's seminal book "Algorithms + Data Structures = Programs"

Systems Structure Modeling Language (S2ML)

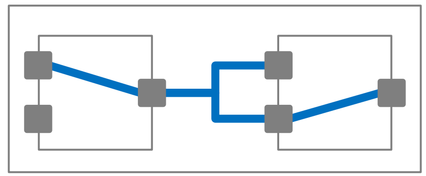
A **structuring paradigm** that unifies **object-** and **prototype-orientation** and an **ontology** of **behavioral modeling languages**

Port



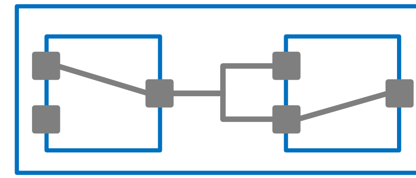
Variable, event…

Connection



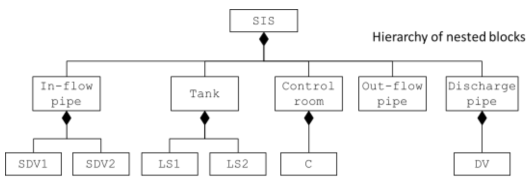
Equation, transition…

Container



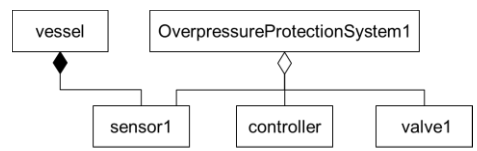
Model, component…

Composition



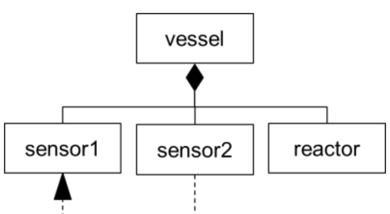
Is-part-of

Aggregation

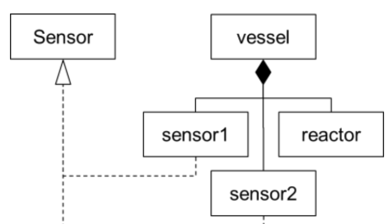


Uses

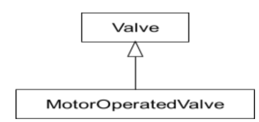
Prototype/Cloning



Class/Instantiation

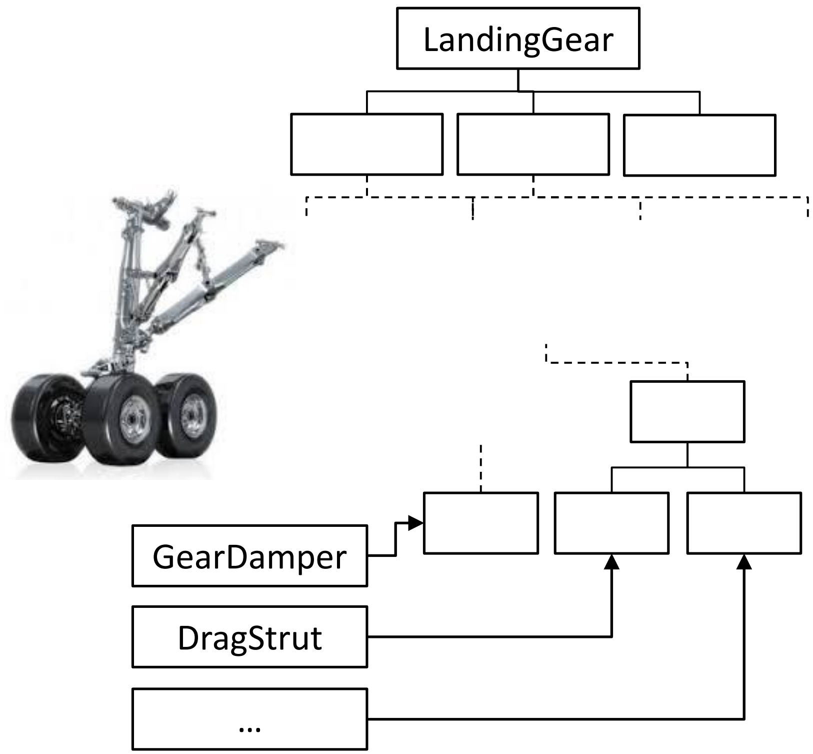


Inheritance



Is-a

Modeling Approaches



- Top-down model design
- System level
- Reuse of modeling patterns
- Prototype-Orientation

system architecture

safety

- Bottom-up model design
- Component level
- Reuse of modeling components
- Object-Orientation

Multiphysics simulation

Models as Scripts

The **model "as designed"** is a script to build the **model "as assessed"**.

```
domain WF {WORKING, FAILED} WORKING<FAILED;

operator Series arg1 arg2 =
   (if (and (eq state1 WORKING) (eq state2 WORKING))
      WORKING
      FAILED);

class Component
   WF state(init = WORKING);
   WF in, out(reset = WORKING)
   probability state FAILED = (exponentialDistribution lambda (missionTime));
   parameter Real lambda = 1.0e-3;
   assertion
      out := (Series in state);
end
```

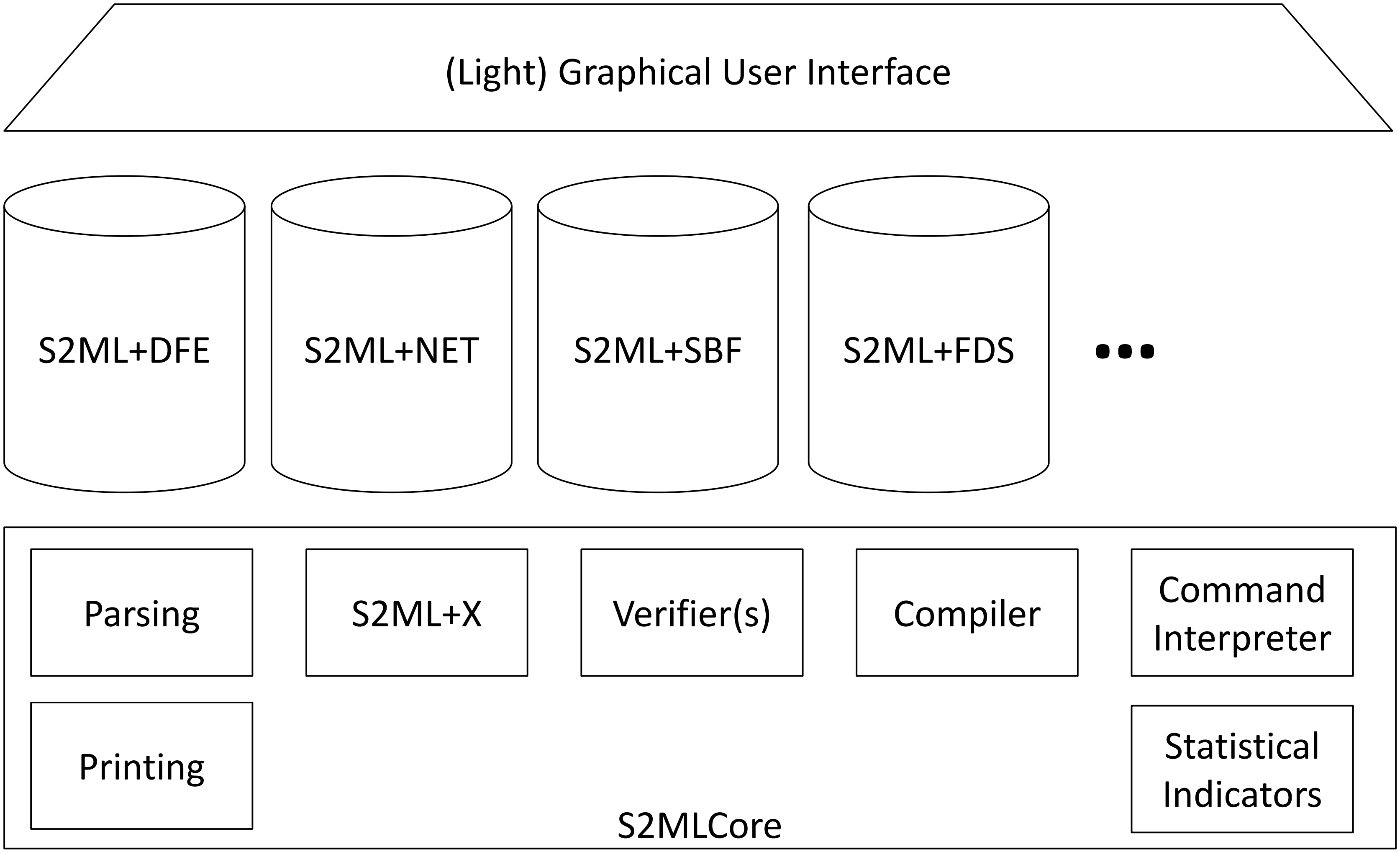
Complex models can be built using **libraries** of **reusable modeling components** and **modeling patterns**.

The S2ML+X Paradigm

Versatile set of **domain specific modeling languages**

Domain	Language
System architecture (structural diagrams)	S2ML
Combinatorial Optimization	S2ML + constraints
Reliability Engineering	S2ML + stochastic Boolean formulas
Logistics	S2ML + hierarchical graphs
Stochastic processes	S2ML + Markov chains
Model-checking	S2ML + finite state automata
Discrete event systems	S2ML + guarded transition systems (AltaRica)
Business processes	S2ML + process algebra (Scola)
…	…

S2ML Toolbox (Proof of Concept)

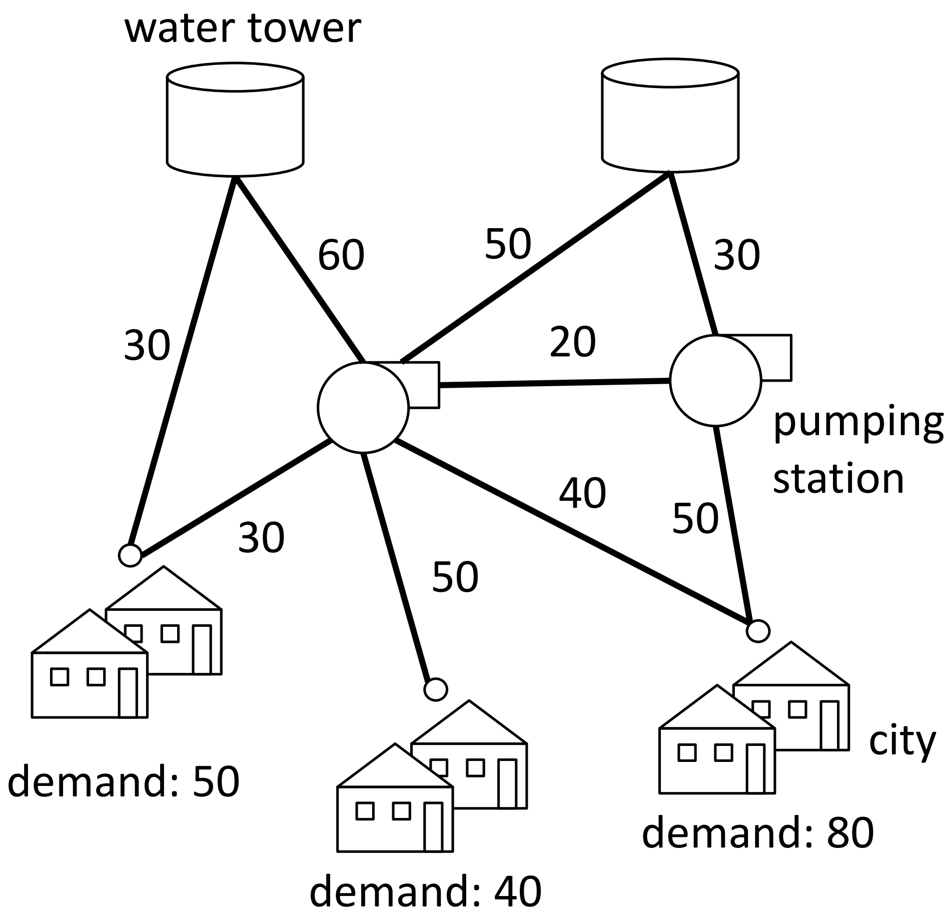


Organization of the course

lecture	Observe	• Description of a use case • Informal analysis of the use case
	Formalize	• Presentation of a modeling framework (S2ML+…) • Study of the use case in this modeling framework
tutorial	Model	• Modeling of similar use cases in the framework
	Experiment	• Virtual experiments • Report on the result of the virtual experiments

Example of Exercise

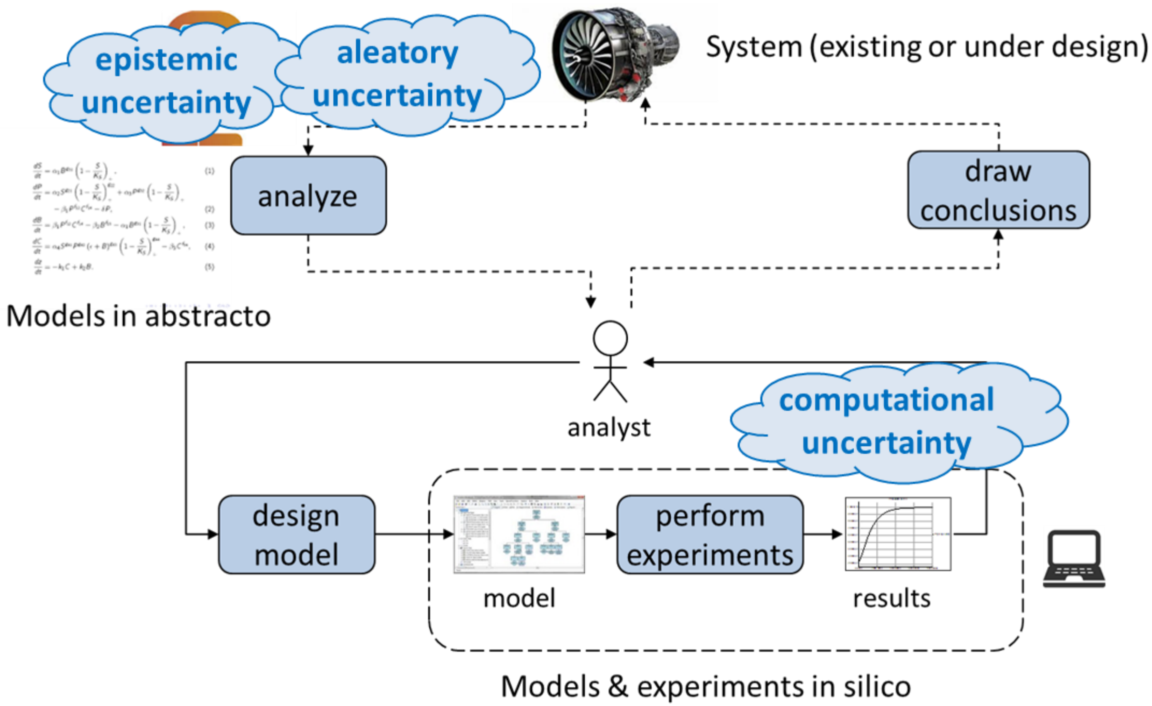
Two water towers supply three cities through a network of pipes and pumping stations. The capacity of pipes is limited.

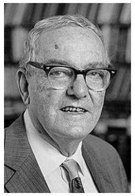


1. Design a S2ML+NET model for this system
2. Use the Ford-Fulkerson algorithm to verify that the demands cities can be satisfied

Computational Uncertainties

The **computational complexity** of virtual experiments **overdetermines** the modeling process: models result always of a **tradeoff** between **accuracy of the description** and **ability to perform calculations** within the available computational resources





Herbert Simon
(1916-2001)
Bounded Rationality

Classes of Modeling Languages

The example of reliability engineering:

Combinatorial Formalisms
- Fault Trees
- Event Trees
- Reliability Block Diagrams
- Finite Degradation Structures

States Automata
- Markov chains
- Dynamic Fault Trees
- Stochastic Petri Nets
- …

Universal Languages
- Agent-based models
- Process algebras
- Python/Java/C++
- …

Expressive power →

States | States + transitions | Deformable systems

Complexity of assessments →

#P-hard but reasonable polynomial approximation | PSPACE-hard | Undecidable

Difficulty to design, to validate and to maintain models →

Experience Feedback

Challenges and Assets

Challenges:

- Software development and maintenance
- Low level in computer science/discrete mathematics/software engineering/programming of students
- Lack of suitable exercises
- Unconventional of teaching/evaluating the course

Assets:

- General theory of behavioral models (systems engineering)
- S2ML Toolbox
- Dozens of exercises ready
- Book in preparation
- Positive feedbacks from students

Concluding Remark

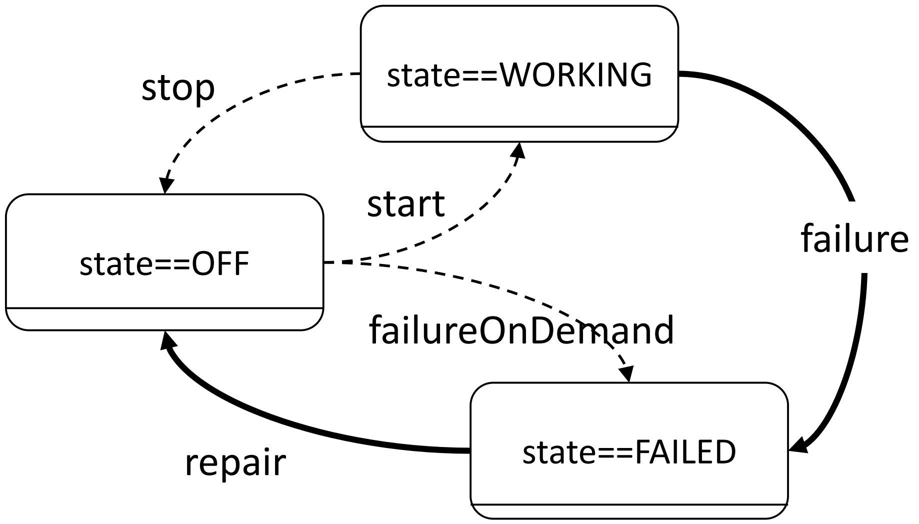
The theory and the technology are (almost) ready
for a full-scale deployment of model-based systems engineering,
but… are we?

Additional Material

AltaRica 3.0 (S2ML + Guarded Transition Systems)

Guarded Transitions Systems:

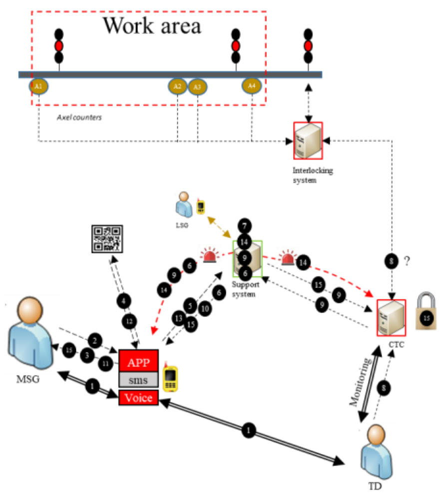
- Are a probabilistic Discrete Events System formalism.
- Are a compositional formalism.
- Generalize existing mathematical framework.
- Take the best advantage of existing assessment algorithms.

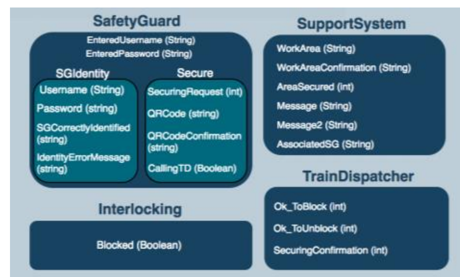


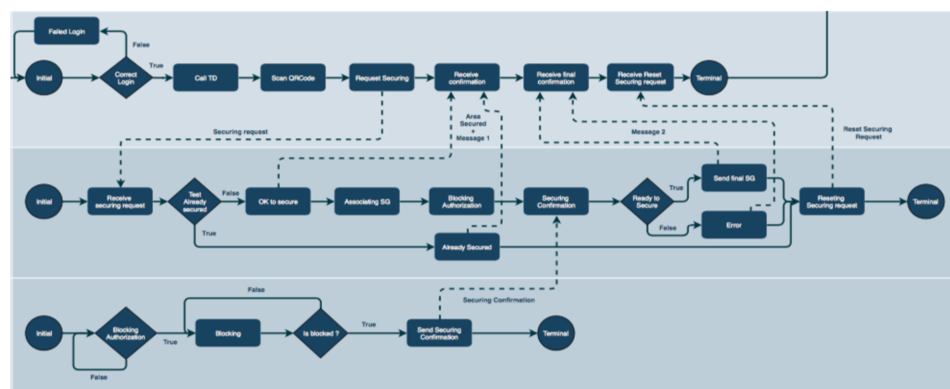
Scola (S2ML + Process Algebra)

Scenario-oriented modeling methodology

- Architecture description
- Dynamic modification of components
- Moving components
- Dynamic creation/deletion of components



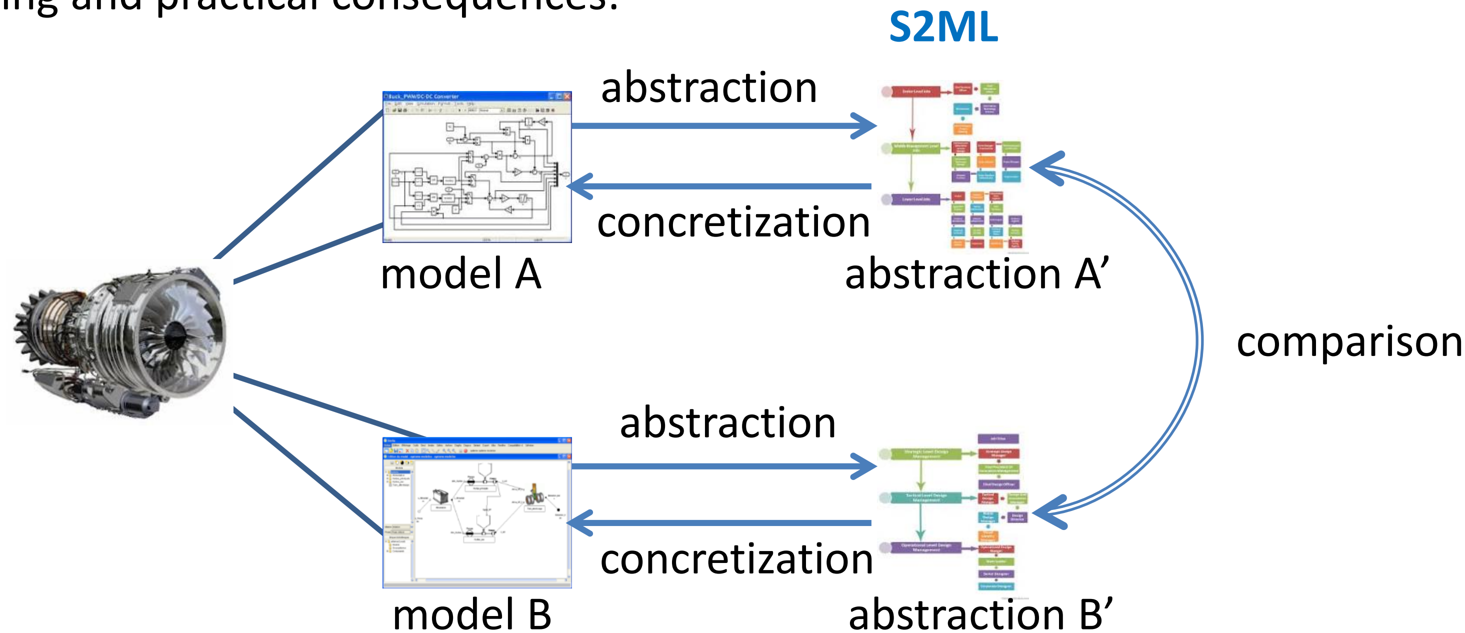




Model Synchronization

Abstraction + Comparison = Synchronization

Meaning and practical consequences:



How to agree on disagreements?

SmartSync Platform