Dresden 08/10/2019

The S2ML+X Paradigm
for Model-Based Systems Engineering and Model-Based Safety Assessment

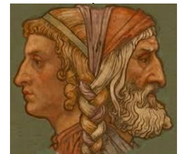
Prof. Antoine B. Rauzy

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

&

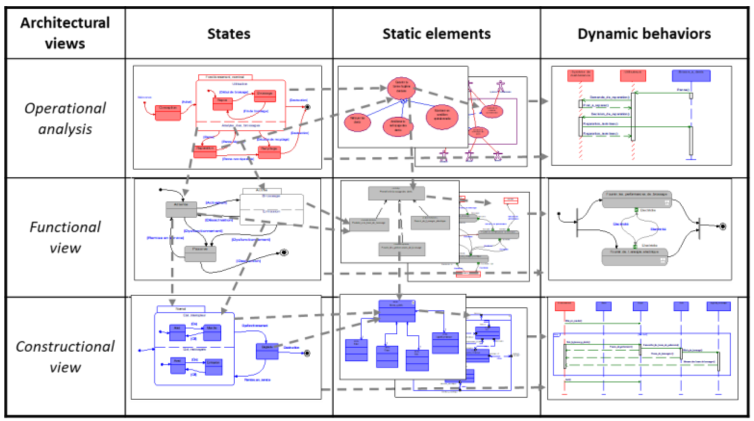
Chair Blériot-Fabre
CentraleSupélec
Paris, France

Disciplines



System Architecture

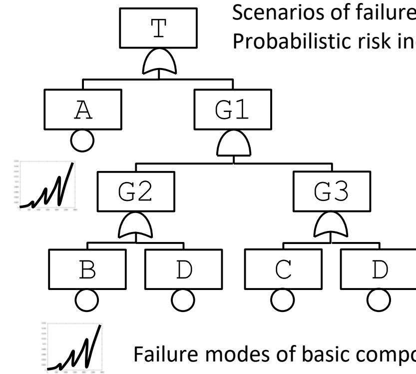
What the system should do?
What the system should be?



Proof that there exists a system that meets the given specification.

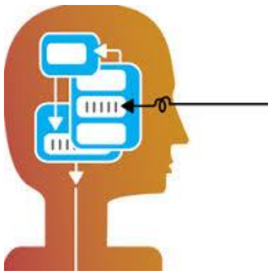
Reliability Engineering

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

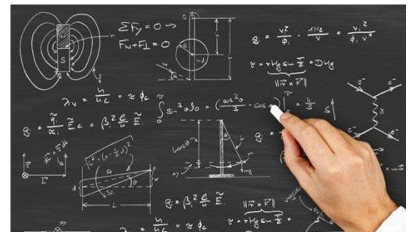


Proof that the specified system is reliable enough to be operated.

Behavioral Models of Technical Systems



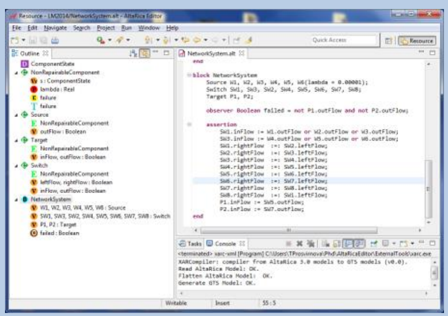
Cognitive Model



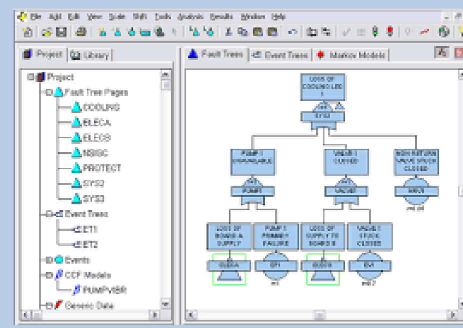
Mathematical Model

Models *in abstracto*

Models *in silico*



Text



Diagrams

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

Ontology of Behavioral Models

Behaviors + Structures = Models*

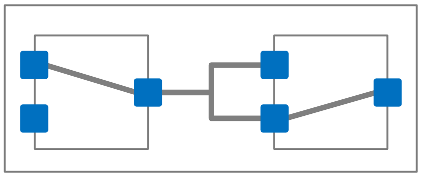
Meaning and practical consequences:

- 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"

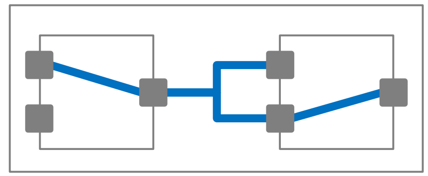
Systems Structure Modeling Language (S2ML)

Port



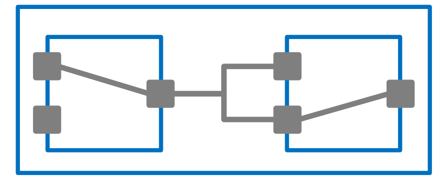
Variable, event…

Connection



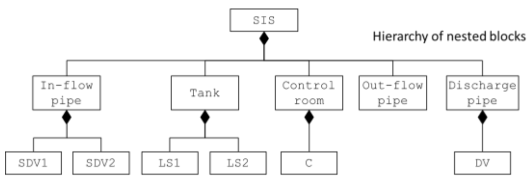
Equation, transition…

Container



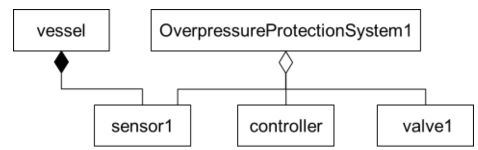
Model, component…

Composition



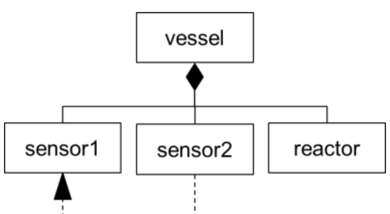
Is-part-of

Aggregation

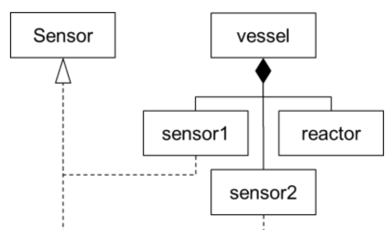


Uses

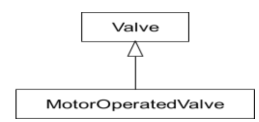
Prototype/Cloning



Class/Instantiation



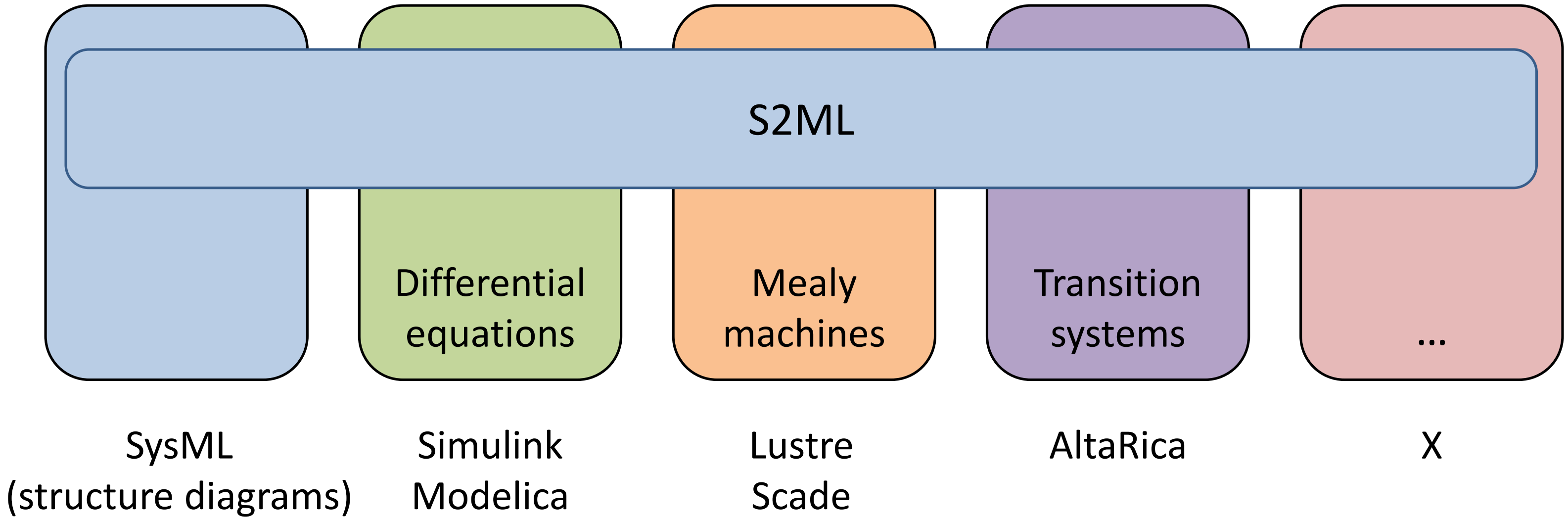
Inheritance



Is-a

The S2ML+X Promise

S2ML: a coherent and versatile set of **structuring constructs** for any behavioral modeling language.



- The **structure of models** reflects the **structure of the system**, even though to a **limited extent.**
- **Structuring** helps to design, to debug, to share, to maintain and to align heterogeneous models.

Models as Scripts

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

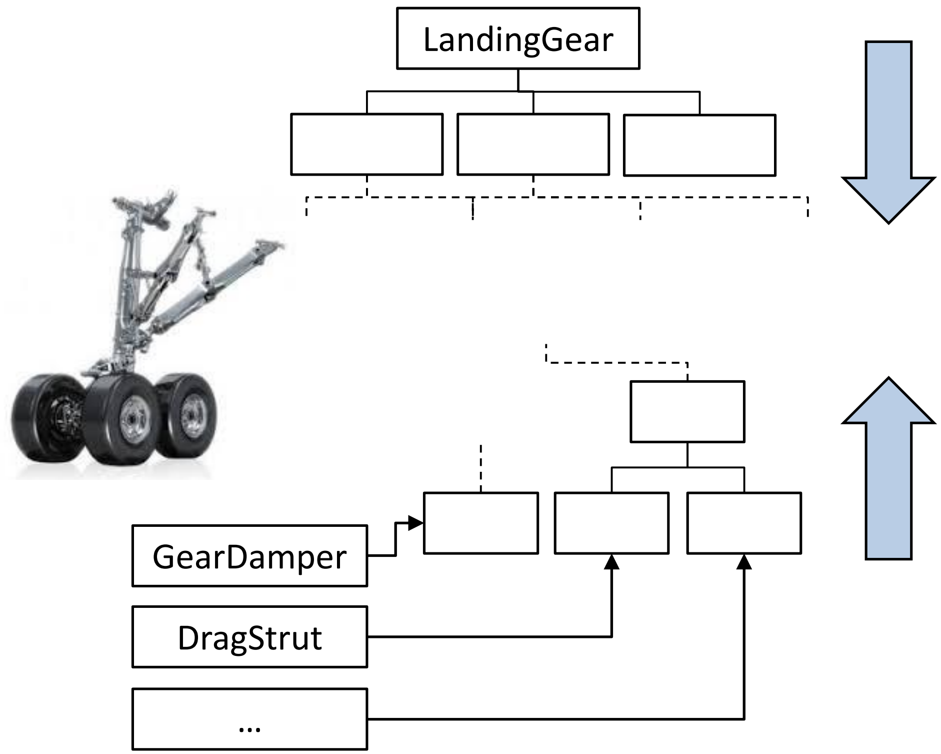
```
domain WF {WORKING, FAILED} WORKING<FAILED;

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**.

Modeling Approaches



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

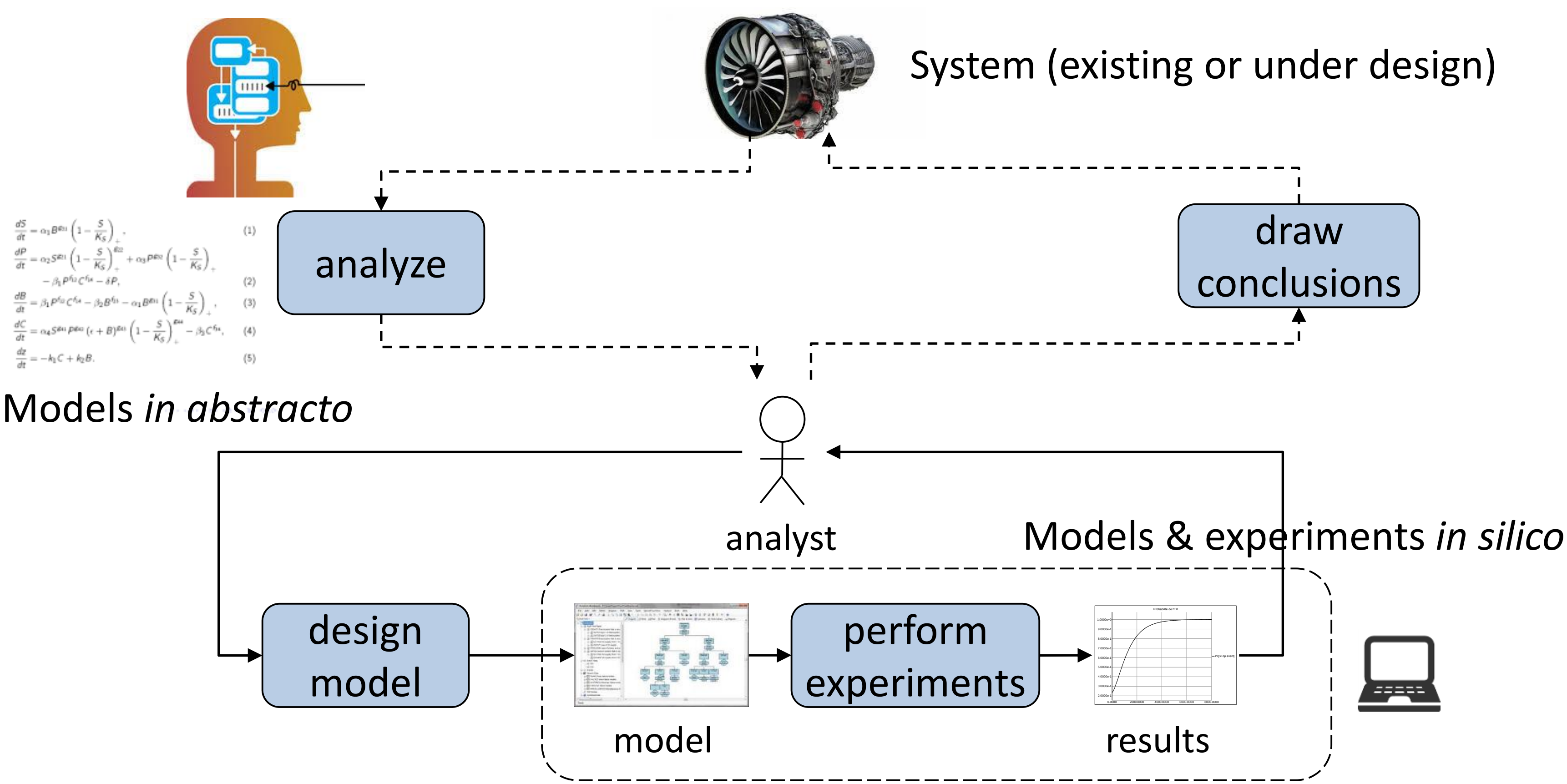
system architecture

safety

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

Multiphysics simulation

Virtual experiments



A model results always of a **tradeoff** between the **accuracy of the description** and the **computational cost** of virtual experiments.

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

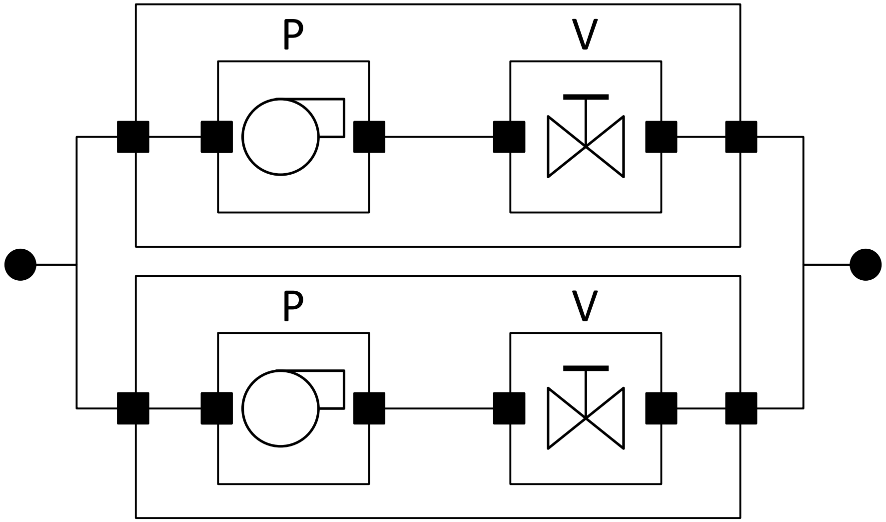
Complexity of assessments →

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

Difficulty to design, to validate and to maintain models →

Open-PSA V4 (S2ML + Boolean Equations)

Enhancing classical **reliability models** (fault trees, reliability block diagrams) with the **expressive power of object-orientation** at **no algorithmic cost**



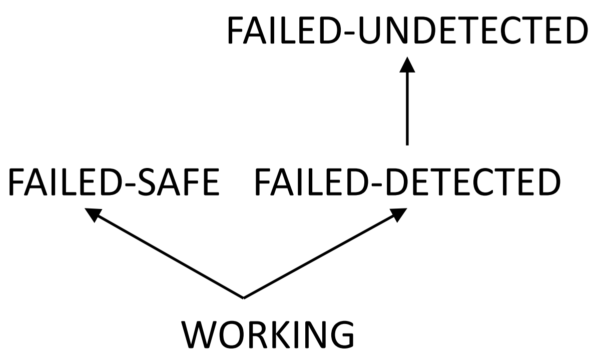

```
class Pump
    extends Component
    …
end

block System
    block Line1
        Pump P;
        …
    end
    clones Line1 as Line2;
    …
end
```

```
Line1.in := in;
Line1.P.in := Line1.in;
Line1.P.out := Line1.P.in and not Line1.P.failed;
…
```

S2ML + Finite Degradation Structures

Lifting-up all classical concepts of reliability engineering to **multi-valued logics** and giving these logics the **expressive power of object-orientation**.



||	W	Fs	Fd	Fu
W	W	W	W	W
Fs	W	Fs	Fs	Fs
Fd	W	Fs	Fd	Fd
Fu	W	Fs	Fd	Fu

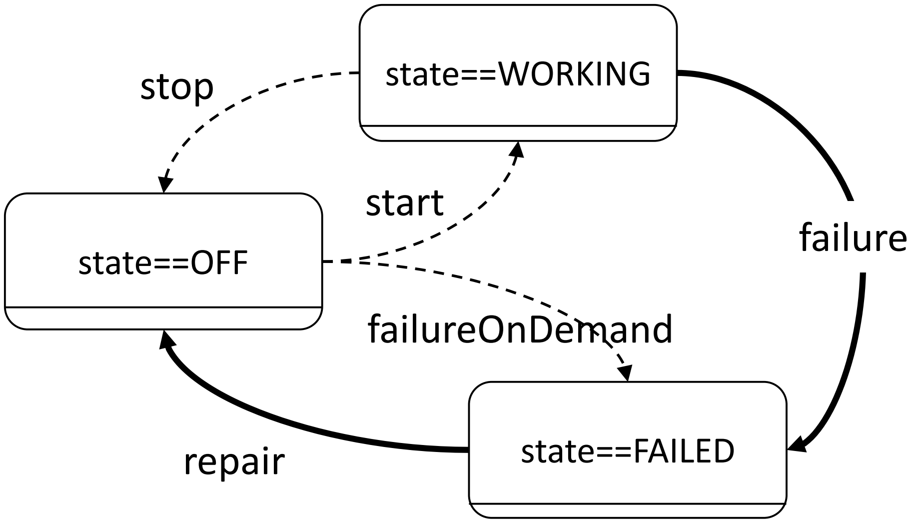
```
domain IEC61508
    {WORKING, FAILED_SAFE,
     FAILED_DETECTED,
     FAILED_UNDETECTED}
    WORKING<FAILED_SAFE,
    WORKING<FAILED_DETECTED,
    …

operator Parallel
    …
end
```

AltaRica 3.0 (S2ML + Guarded Transitions 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.



OpenAltaRica SYSTEM SAFETY ANALYSIS TECHNOLOGY | SystemX INSTITUT DE RECHERCHE TECHNOLOGIQUE

SAFRAN

THALES

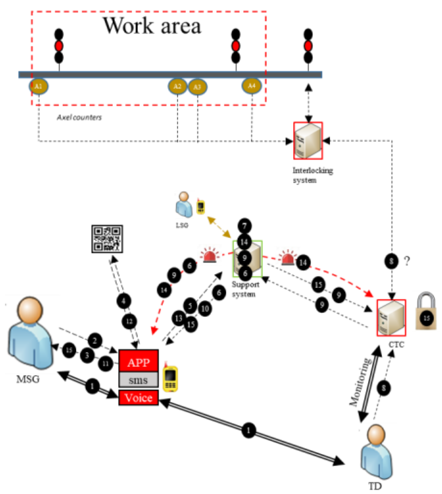
AIRBUS

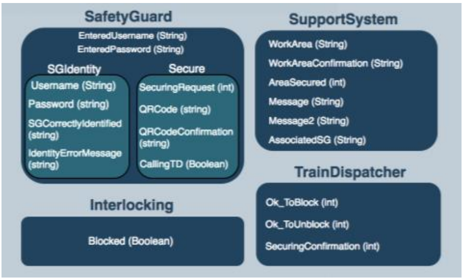
Scola (S2ML + Process Algebra)

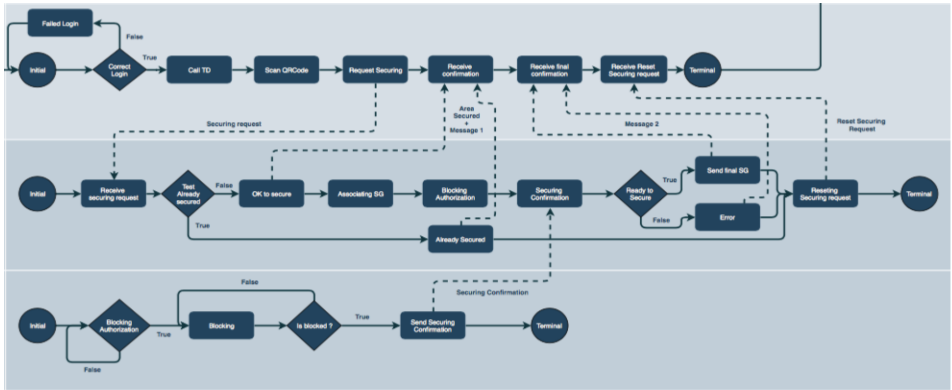
BANE NOR

Scenario-oriented modeling methodology

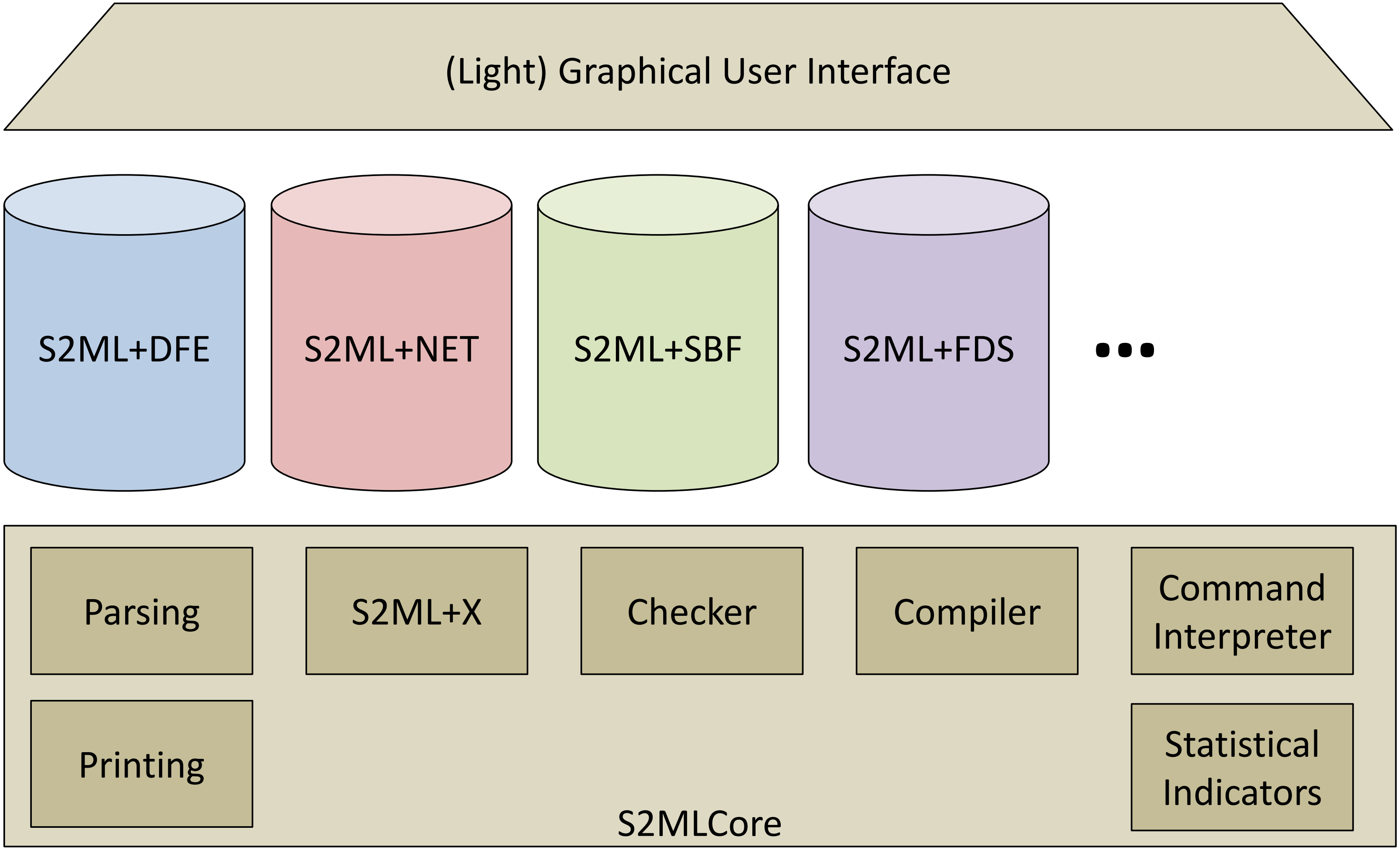
- Architecture description
- Dynamic modification of components
- Moving components
- Dynamic creation/deletion of components







S2ML Toolbox (Proof of Concept)



The S2ML+X Paradigm in Pedagogical Action

Versatile set of **domain specific modeling languages**

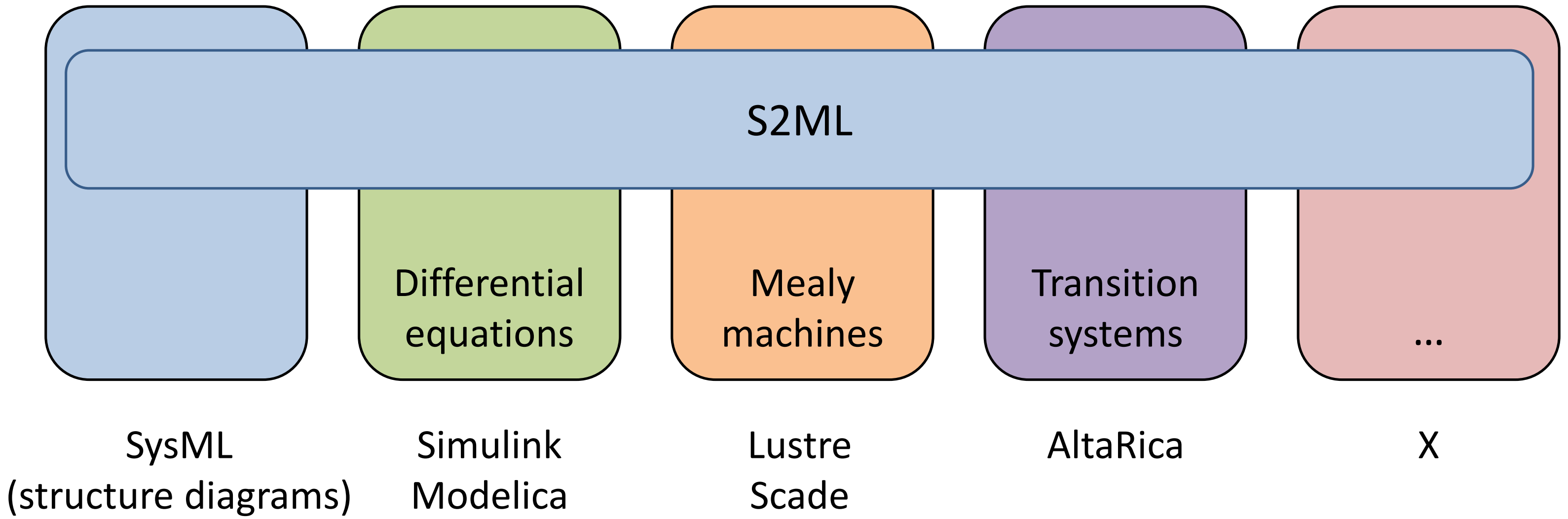
Domain	Language
System architecture (structural diagrams)	S2ML
Stochastic simulation	S2ML + data-flow equations
Combinatorial Optimization	S2ML + constraints
Reliability Engineering	S2ML + stochastic Boolean formulas
Logistics	S2ML + hierarchical graphs
Stochastic processes	S2ML + Markov chains
Model-checking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

Alignment of Heterogeneous Models

Models are designed by **different teams** in **different languages** at **different levels of abstraction**, for **different purposes.** They have also **different maturities**.

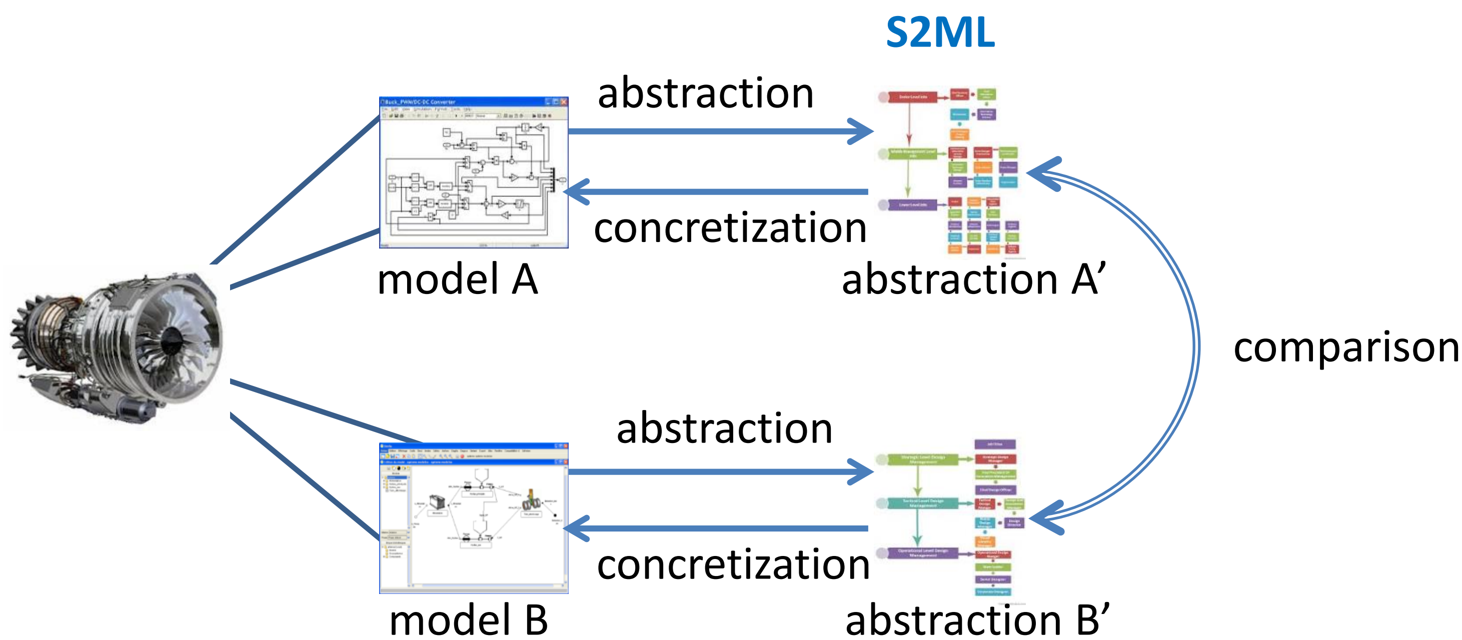
The question is how to ensure that they are "speaking" about the **same system**, i.e. to **align** them.

As the **behavioral part** of models is **purpose-dependent**, the main way to compare models is to compare their **structure**.



Model Synchronization

Abstraction + Comparison = Synchronization



How to agree on disagreements?