ALSSE 2017 Oslo

S2ML for X

Prof. Antoine B. Rauzy

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

Chair Blériot-Fabre
CentraleSupélec
Paris, France
Agenda

Model-Based Systems Engineering
System Structure Modeling Language (S2ML)
Model-Based Risk/Safety Assessment
Syntactic Structures
Model Synchronization
Agenda

Model-Based Systems Engineering
System Structure Modeling Language (S2ML)
Model-Based Risk/Safety Assessment
Syntactic Structures
Model Synchronization
Models Are Everywhere

• The systems designed by industry are more and more complex and interconnected. Not only these products are more and more complex but also the processes by which they are designed/produced/operated/decommissioned and organizations that implement these processes are.

• To face this complexity, the different engineering disciplines (mechanics, thermic, electric and electronic, software, architecture...) virtualize their contents to a large extent, i.e. they are designing models. We entered the era of:

  Model-Based Systems Engineering

• Each system comes with dozens of models. More and more of these models are embedded into systems and used for their operation.
The Science and Engineering of Models

Models must be taken seriously and considered as first class citizens. This raises a number of challenges:

• Better understand the nature of models and their roles in industrial processes.
• Develop the “Art of Modeling”(*) in each and every engineering discipline.
• Manage models throughout the life-cycle of systems.
• Design tools and methods to support the integration of engineering disciplines/processes through the integration of models they produce.
• Teach and give taste of modeling to (future) engineers.
• ...

The emerging science of complex systems is the science of models

(*) In reference to Knuth’s famous series of books about “The Art of Programming”
Models Engineering

Fact 1: To design a model, we need a **modeling language** (would it be purely graphical), just as to design a program, we need a programming language.

Fact 2: Models of a complex system cannot be simple, otherwise they cannot capture the complexity of the system* (information loss). Therefore, they need to be **structured**, documented, managed… in a word, we need an **engineering of models**.

Questions:
- What is a good modeling language?
- What is a good palette of modeling languages?
- How to manage versions and configurations of models through the life-cycle of systems?
- …

(*) Models of complex systems are simplex, in the sense of A. Berthoz.
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”
Agenda

Model-Based Systems Engineering
System Structure Modeling Language (S2ML)
Model-Based Risk/Safety Assessment
Syntactic Structures
Model Synchronization
S2ML: System Structure Modeling Language

- A structuring paradigm that unifies the two dominant structuring paradigms for modeling languages, i.e. object-orientation and prototype-orientation.
- A modeling language on its own, dedicated to architecture description.

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

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

NTNU Norwegian University of Science and Technology
S2ML Promise: 1) Models of Structures

S2ML aims at providing a necessary and sufficient language to describe the functional and/or physical structures of systems.

Describing the structure of a system is a modeling process that aims at architecting the system, i.e. eventually at improving the comprehension / specification of that system.
S2ML Promise: 2) Structure of Models

S2ML aims at providing a structuring paradigm of system engineering modeling languages.

Structuring helps to design, to debug, to share, to maintain and to synchronize models.
Why not SysML?

SysML is a graphical notation, derived from UML, to address system modeling. It provides two types of diagrams to represent structures: Definition Block Diagrams and Internal Block Diagrams (1). It could thus be a candidate formalism for our purpose. However,

- A model, which is a **mathematical object**, should not be confused with its **graphical representations**. Even though graphical representations are excellent supports for the **communication** amongst stakeholders, they are able to represent only **partially** the models, except for formalisms with very low (or very ambiguous) expressiveness. Moreover, there may be **several graphical representations** of the same concept, each more or less convenient in a given context.

- SysML **lacks** of some **essential structuring constructs**.

(1) Parametric Diagrams and Package Diagrams cannot be used directly to represent structures, although they are considered also as structural.
Why not SysML?

In a word:

• Graphical representations are a very good communication mean. Therefore, we shall use SysML graphics and vocabulary as much as possible.

• However:  

  **Concepts should come first**

S2ML aims at proposing a minimal yet sufficient set of concepts to represent structures of systems and to structure models.
Cooling System

tank

vessel

controller

valve 1

valve 2

level sensor 1

level sensor 2

reactor
## Basic Components

S2ML is made of the following basic components.

<table>
<thead>
<tr>
<th>Component</th>
<th>Representation</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ports</td>
<td><img src="image" alt="Ports Diagram" /></td>
<td>Ports are basic objects of models, e.g. variables, events, equations, transitions…</td>
</tr>
<tr>
<td>Connections</td>
<td><img src="image" alt="Connections Diagram" /></td>
<td>Connections are used to describe relations existing between ports.</td>
</tr>
<tr>
<td>Blocks</td>
<td><img src="image" alt="Blocks Diagram" /></td>
<td>Blocks are containers. They can contain ports, connections and other blocks.</td>
</tr>
</tbody>
</table>
Blocks as Prototypes & Composition

A block is a **container** for ports, connections and other blocks. Each block is a **prototype**: it has a unique occurrence in the model.

The block “system” **composes** the blocks “tank”, “valve 1”… The block “reactor” **is part of** the block “vessel”.

![Hierarchy of nested blocks diagram](image)
Cloning

A system may contain similar components, e.g. the sensors or the valves of our example. The corresponding copy then contains several copies of the same block.

A first way to avoid duplicating the description of a block consists in cloning an already existing block.

```
block vessel
  block sensor1
    port input, output;
  end
  block sensor2 clones sensor1;
  end
  block reactor
    port output1, output2;
  end
  connection [sensor1.input, reactor.output1];
  connection [sensor2.input, reactor.output2];
  ...
end
```

cloning (of a block)
A second way to avoid duplicating the description of a block consists in declaring a model of the duplicated block in a separate modeling entity, so-called a **class**, and then in **instantiating** this class.

```plaintext
class Sensor
    port input, output;
end

block vessel
    Sensor sensor1, sensor2;
    block reactor
        port output1, output2;
    end
    connection [sensor1.input, reactor.output1];
    connection [sensor2.input, reactor.output2];
    ...
end
```

---

**Diagram:**
- **Sensor**
- **vessel**
- **sensor1**
- **reactor**
- **instantiation** (of a class)
Prototypes versus Classes

**Concept space (C)**
- «Sandbox»
- Model creation
- refinement
- refinement
- Final model
- prototypes

**Knowledge space (K)**
- Stabilized knowledge
- Libraries of «on-the-shelf» components

- Reuse
- Add new components
- Reuse
- Add new components

NTNU Norwegian University of Science and Technology
Inheritance

Aside the composition, that defines a “is-part-of” relation, S2ML provides also an inheritance mechanism, i.e. a “is-a” relation. A class or a block can inherit the content of another class (or another block in the same modeling entity).

```plaintext
class Valve
    port input, output;
end

class MotorOperatedValve extends Valve;
    port inputTorque;
end

block system
    ...
    block MyValve extends Valve;
    ...
end
end
```

![Inheritance Diagram]

- ![Inheritance Diagram]
Functional Chains
Aggregation

S2ML provides a mechanism for blocks to use blocks defined elsewhere in the same modeling entity. The using block aggregates the used block. This mechanism is especially useful to describe the so-called functional chains.

```plaintext
block system
    ...
    block OverpressureProtectionSystem1
        embeds owner.vessel.sensor1 as sensor;
        embeds owner.controller as controller;
        embeds owner.valve1 as actuator;
    ...
end
...
end
```

To access to the parent block
Agenda

Model-Based Systems Engineering
System Structure Modeling Language (S2ML)
Model-Based Risk/Safety Assessment
Syntactic Structures
Model Synchronization
Issues with “Classical” Safety Models

Classical modeling formalisms used for safety analyses lack of expressive power and/or are very close to mathematical equations (lack of structure).

→ **Distance** between systems specifications and models;

→ Models are **hard to design** and even **harder to share with stakeholders** and to **maintain** throughout the **life-cycle** of systems.

→ **Very conservative** approximations
The Promise of Model-Based Safety Assessment

Modeling systems at **higher level** so to reduce the distance between systems specifications and models (without increasing the complexity of calculations).

- Ability to **animate/simulate** models: to ease **model validation** and **discussions with stakeholders**;
- One model, several safety goals: to ease **versioning**, **configuration** and **change** management;
- One model, several assessment tools: **versatility** of assessments, **quality-assurance** of results;
- Fine grain analyses: to **avoid over-pessimism**.

```
class HydraulicPump
  Boolean working (init = false);
  event failure (delay = exponential(lambda));
  transition
    failure: working -> working := false;
end
```

AltaRica 3.0
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.

AltaRica 3.0 is an optimal modeling formalism
# Open-PSA format 3.0

## Purpose of Format

The Open PSA model exchange format aims at:

- giving a clear semantics for each and every construct of Probabilistic Safety Assessment (PSA) and Probabilistic Risk Assessment (PRA) models,
- providing a way to exchange models amongst tools,
- making it possible to connect models written in different formalisms.

The version 3.0 of the format encompasses:

- Fault Trees
- Block Diagrams
- Event Trees
- Markov Chains

A module is defined for each of these formalisms.

<table>
<thead>
<tr>
<th>Formalism</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2ML + Boolean equations</td>
<td>Fault Trees</td>
</tr>
<tr>
<td>S2ML + Markov chains</td>
<td>Block Diagrams, Event Trees, Markov Chains</td>
</tr>
</tbody>
</table>
Agenda

Model-Based Systems Engineering
System Structure Modeling Language (S2ML)
Model-Based Risk/Safety Assessment
Syntactic Structures
Model Synchronization
Taxonomy of Engineering Models

Models are designed at different level of abstraction, for different purposes and in different modeling formalisms.

Models to communicate amongst stakeholders

Models to generate artefacts (via code generation) or physical components (via additive manufacturing)

Informal models, even thought they are written in standardized notations, sometimes called semi-formal

Formal models, that essentially encode and organize (a given type of) mathematical equations
Thesis

There is an epistemic gap between informal and formal models

Meaning and practical consequences:
• Informal models and formal models have radically different natures and purposes.

<table>
<thead>
<tr>
<th>Models to communicate</th>
<th>Models to calculate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standardized notations</td>
<td>Languages</td>
</tr>
<tr>
<td>Pragmatics</td>
<td>Formal semantics</td>
</tr>
<tr>
<td>(external meaning)</td>
<td>(mathematical equations)</td>
</tr>
<tr>
<td>Implicit knowledge</td>
<td>Explicit knowledge</td>
</tr>
</tbody>
</table>

• **Both types** of models are **useful**.
• **Passing from informal** models to **formal** ones requires an **engineering process**. This process **cannot be automated**.
• As **informal models** are **computerized**, we can design tools to **process** them.
The Syntactic Point of View

*Colorless green ideas sleep furiously*

Noam Chomsky (1957)
Syntactic Structures
Reverse Engineering of Textual Specifications

Mélissa Issad PhD Thesis

Siemens CBTC

Technical Specification

- 6 documents
- ~1000 pages each
- Incomplete
- Mixing levels of abstraction

Objective: Safety Assessment
Scenario-Based Approach

Mélissa’s proposal:
• Designing scenarios of use is the most efficient way to communicate with experts (system designers & safety analysts)
• Scenarios: formal syntax + pragmatics
• Co-construction of scenarios and model of system architecture

Scola = S2ML + Process Algebra
How to check requirements for:
- Clarity?
- Consistency?
- Completeness?
- ...
Requirements Engineering

Benoît’s proposal:

- Requirements: syntactic structure + hypertext + pragmatics
- Co-construction of requirements and models of system architecture (S2ML+X)

- Scripts to check syntactic properties of requirements and models
Agenda

Model-Based Systems Engineering
System Structure Modeling Language (S2ML)
Model-Based Risk/Safety Assessment
Syntactic Structures
Model Synchronization
Thesis

Abstraction + Comparison = Synchronization

Meaning and practical consequences:

How to agree on disagreements?
Model Synchronization

Anthony Legendre PhD Thesis

- “Schizophrenic” development of MBSE and MBSA processes
- Definition of synchronization points and synchronization needs