Towards an Ontological Framework for Validity Frames

Rakshit Mittal[1], Raheleh Eslampanah[1], Lucas Lima[1,2], Dominique Blouin[3], Hans Vangheluwe[1]

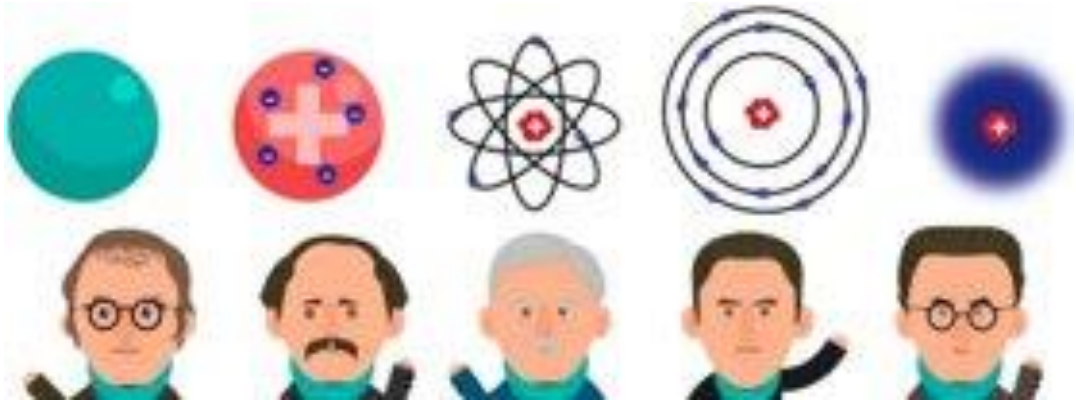
[1]University of Antwerp – Flanders Make, Belgium
[2]Departamento de Computação, Universidade Federal Rural de Pernambuco, Brazil
[3]Telecom Paris, Institut Polytechnique de Paris, France

Science

- Building models to explain the world

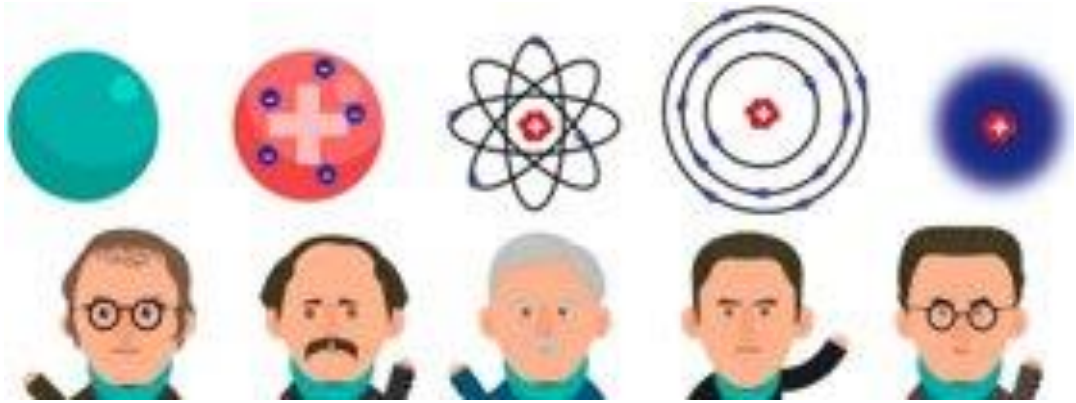
Science

- Building models to explain the world
- Models 'evolve'

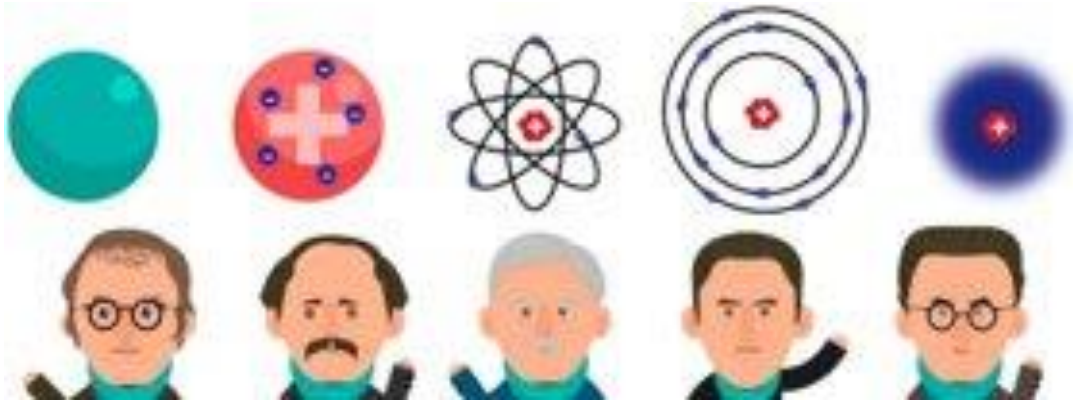


Science

- Building models to explain the world
- Models 'evolve'
- Why?

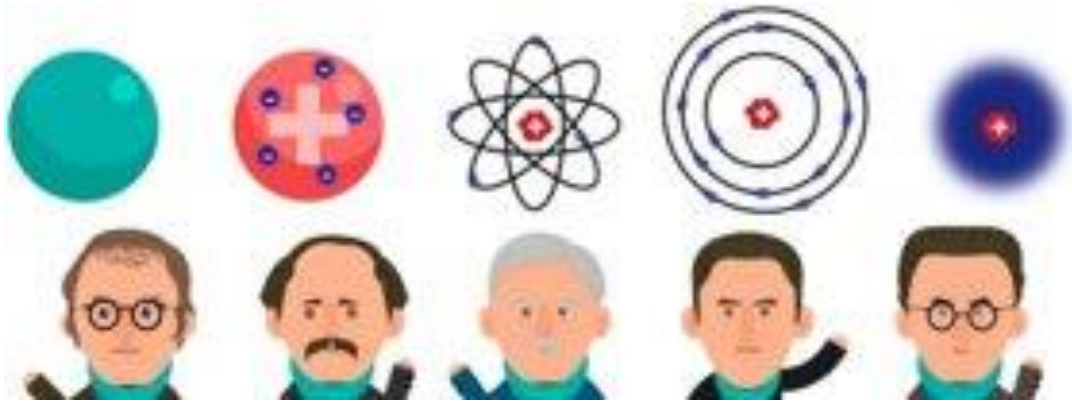


Science



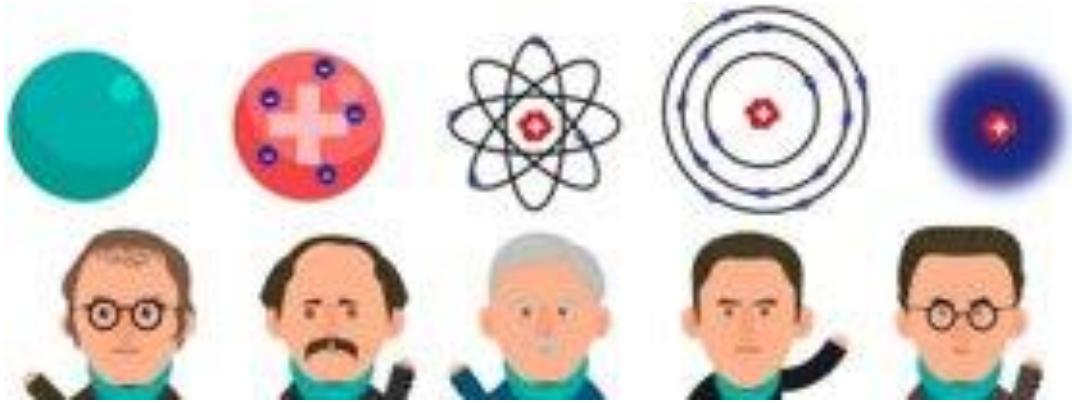
- Building models to explain the world
- Models 'evolve'
- Why?
- Because our new observations do not match the predictions from the previous model anymore!

Science



- Building models to explain the world
- Models 'evolve'
- Why?
- Because our new observations do not match the predictions from the previous model anymore!
- Or maybe, our resolution has decreased, meaning the previous model is not as accurate as needed now.

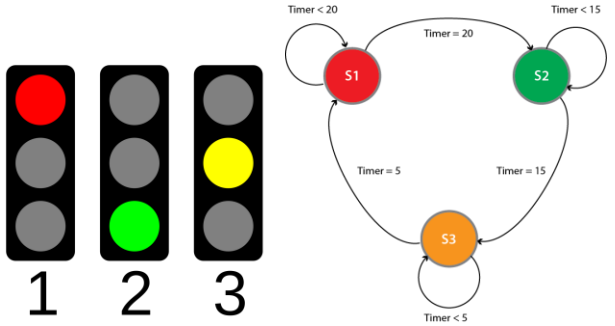
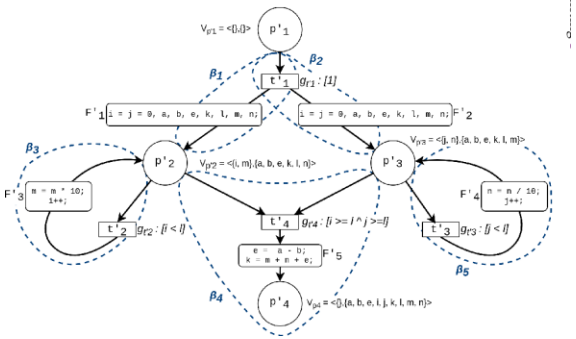
Science



- Building models to explain the world
- Models 'evolve'
- Why?
- Because our new observations do not match the predictions from the previous model anymore!
- Or maybe, our resolution has increased, meaning the previous model is not as accurate as needed now.
- In other words, 'the evolved experimental frame and requirements are not within the model's validity frame'

disclaimer

- The model need not always be 'conceptual', and the modelled system need not always be 'real'

	Real Model	Conceptual Model
Real System		
Conceptual System		

Proposal (What?)

- Model the experimental frame
- Model the (representation) validity frame of a model

Why?

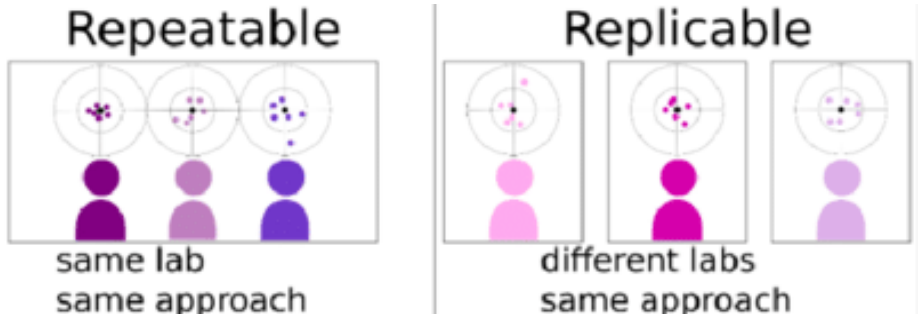
Why?

- Model everything!

Why?

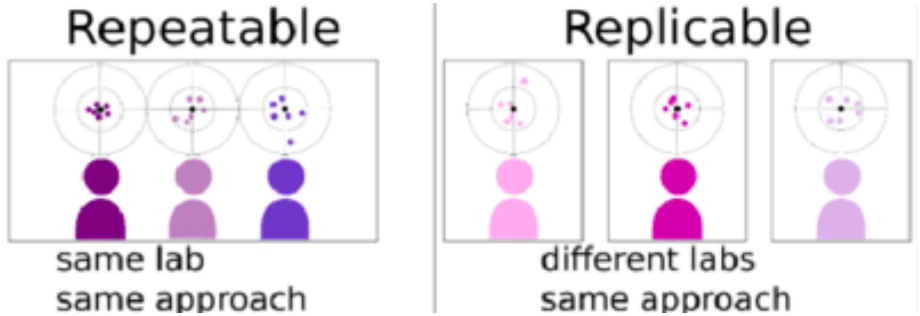
*experiments to build models

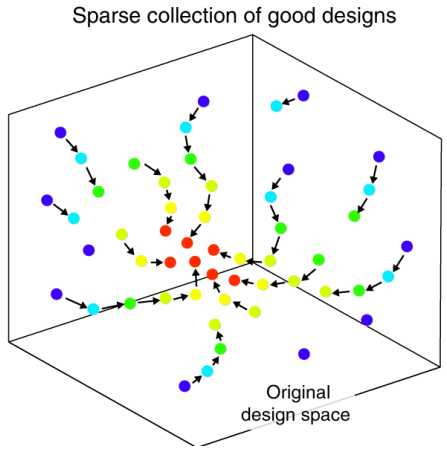
- Model everything!
- Modeling Experiments*:
  - Traceability of Experiment Data
  - Experiment Replicability



Why?

- Model everything!
- Modeling Experiments:
  - Traceability of Experiment Data
  - Experiment Replicability
- Modeling Validity:
  - Model substitutability
  - Consistent twinning
  - Pruning design-space



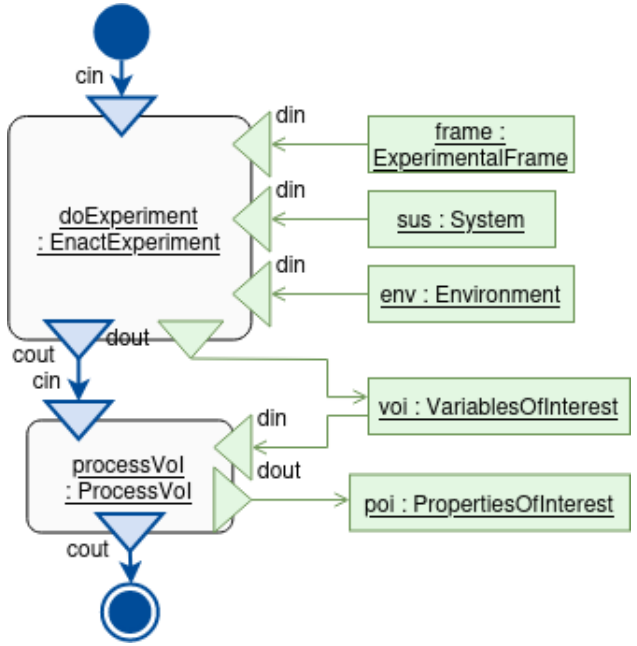


How?

- Experimental Frame
  - The set of circumstances in which an experiment takes place
- Frame Specifications (diverging/orthogonal from Zeigler's 'experimental frame')
  - 'Descriptors' of the Experimental Frame

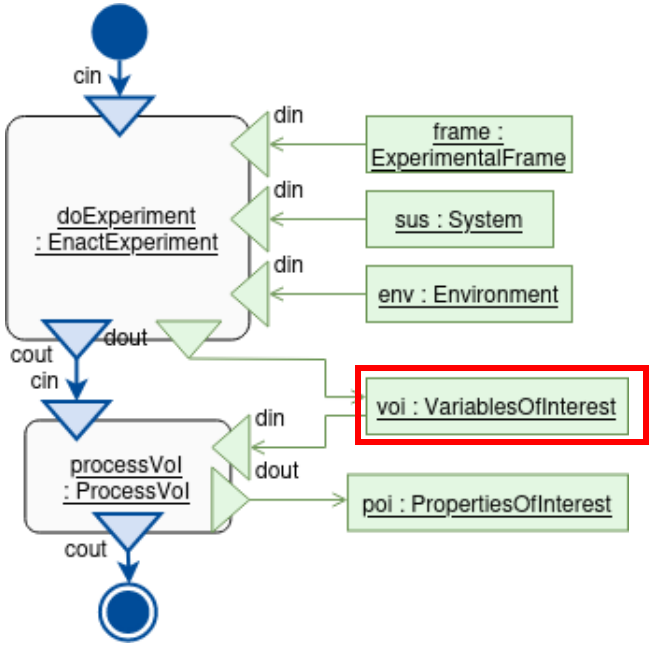
What is an experiment, then?

- A set of activities
  - Performed according to a defined workflow
  - On a specific system
  - In a specific environment
  - Under specific conditions (the frame)
  - To obtain certain variables of interest
  - Used to compute properties of interest



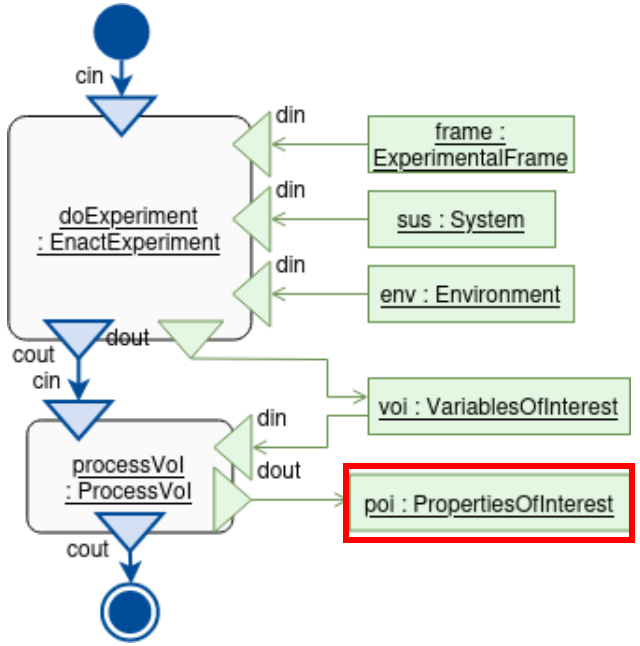
What is an experiment, then?

- A set of activities
  - Performed according to a defined workflow
  - On a specific system
  - In a specific environment
  - Under specific conditions (the frame)
  - To obtain certain variables of interest
  - Used to compute properties of interest
- Variable of Interest
  - Experiment traces
  - Observable inputs/states/outputs



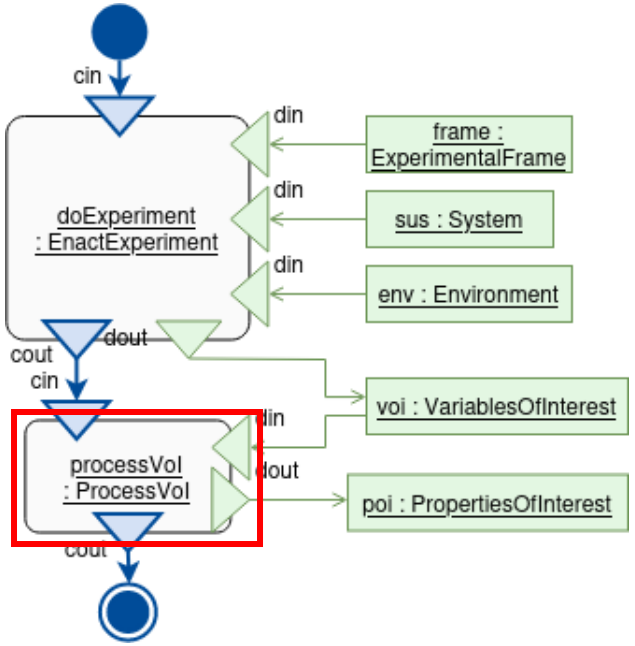
What is an experiment, then?

- Property of Interest
  - The final property/outcome of a system
  - May or may not be observable
  - Usually relates to goal and requirement specification



What is an experiment, then?

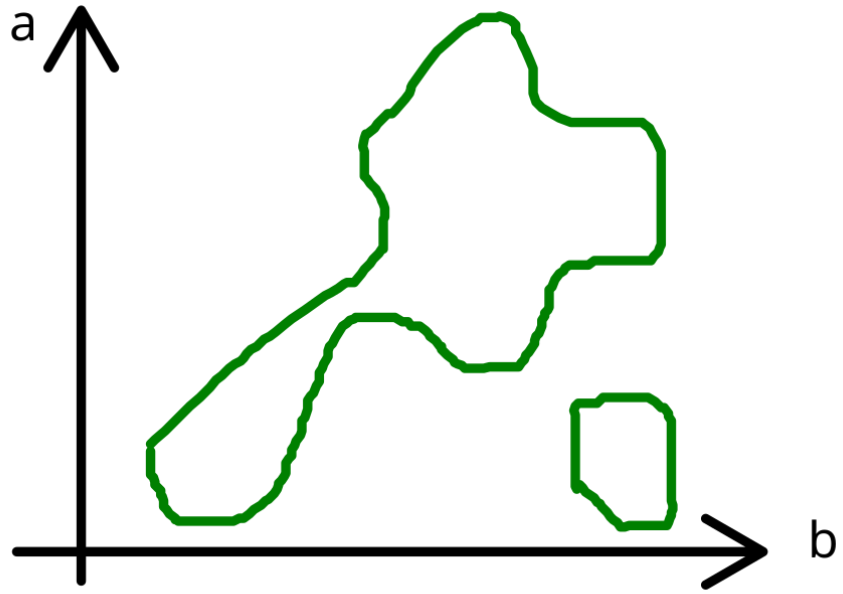
- Property of Interest
  - The final property/outcome of a system
  - May or may not be observable
  - Usually relates to goal and requirement specification
- 'Process VoI' activity
  - algorithm which takes VoI
  - and computes the final PoI
  - e.g. calculating Gain from input and output amplitude
  - Note: could be an identity function



But why waste time describing experimental frames when the topic is validity frames?

But why waste time describing experimental frames when the topic is validity frames?

- Because the validity frame of a model is a subspace of its experimental frame space!



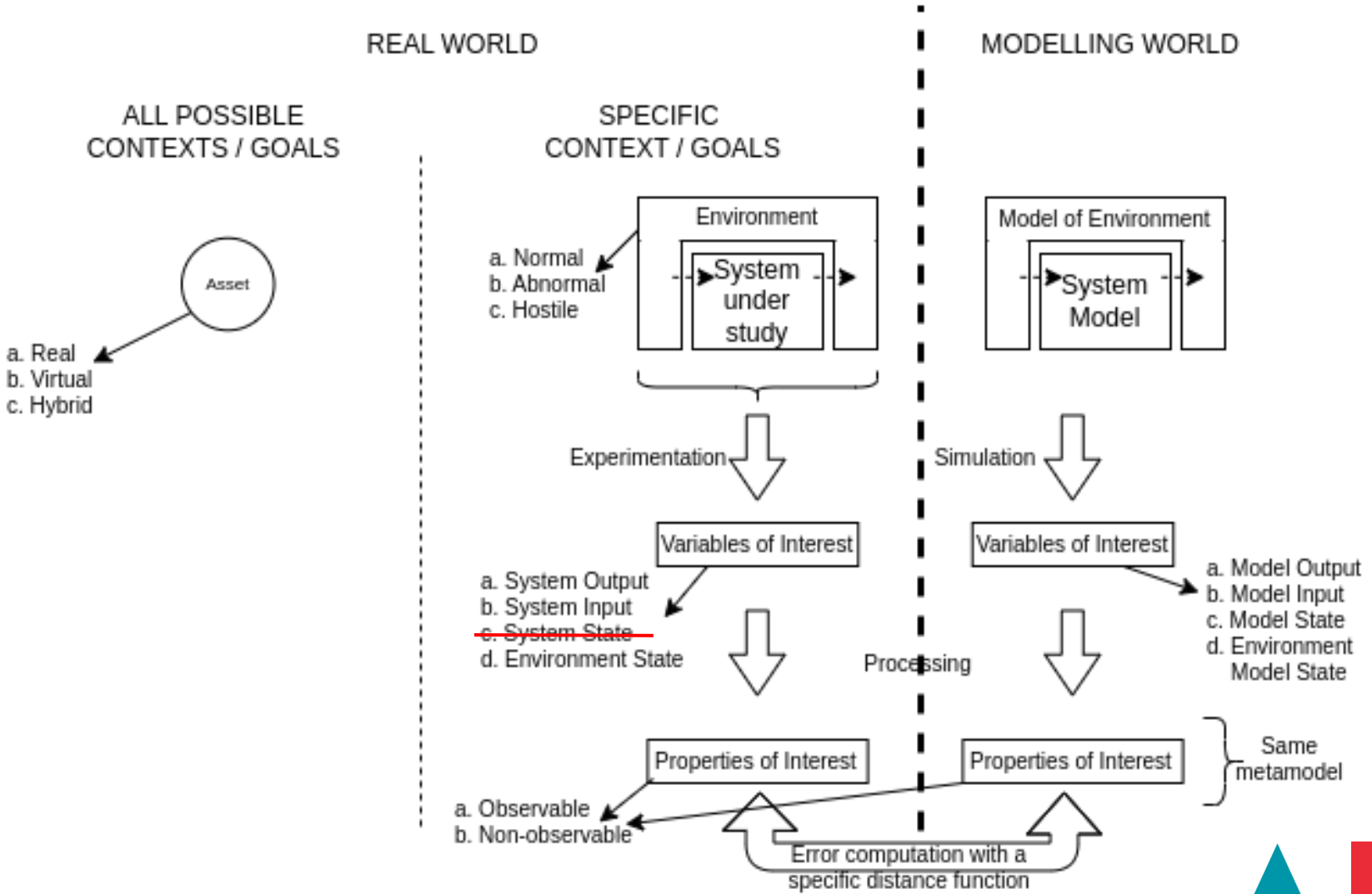
a and b are experimental specification parameters

Validity

- The state of a system that satisfies certain goals
  - goals measured as properties of interest.

- Representational Validity
  - The measure of how closely a model represents the system it models.
  - Function of the distance between the PoIs of the model and the system.
    - So, it is a function of:
      - The system-specific properties of interest
      - The distance function
      - The distance threshold

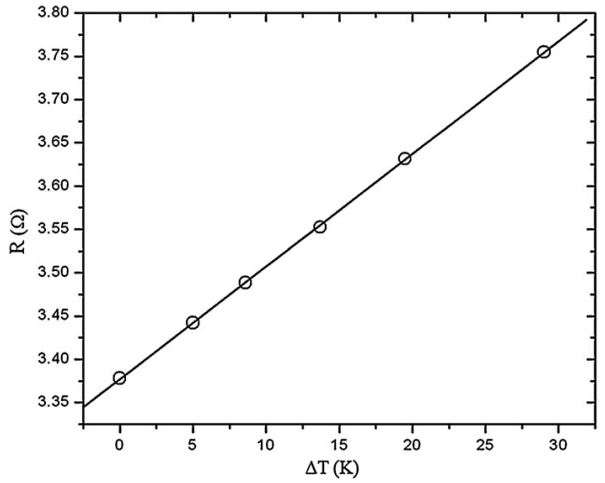
Validation

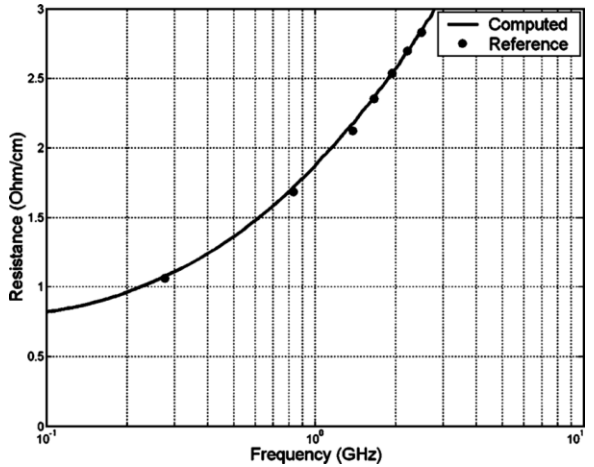

REAL WORLD
MODELLING WORLD
ALL POSSIBLE CONTEXTS / GOALS
SPECIFIC CONTEXT / GOALS
Asset
a. Real
b. Virtual
c. Hybrid
Environment
a. Normal
b. Abnormal
c. Hostile
System under study
Model of Environment
System Model
Experimentation
Simulation
Variables of Interest
Variables of Interest
a. System Output
b. System Input
c. System State
d. Environment State
a. Model Output
b. Model Input
c. Model State
d. Environment Model State
Processing
Properties of Interest
Properties of Interest
Same metamodel
a. Observable
b. Non-observable
Error computation with a specific distance function

But that is not all!

- The validity is also a function of the experimental frame / context!

- For example, the Ohmic resistor model fails:
  - In higher or lower temperature than the reference temperature
  - At higher power
  - At higher frequency





So?

Validity is not a singular Boolean!

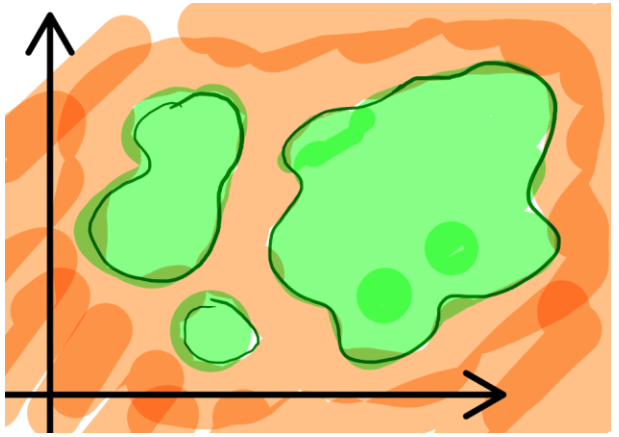
Okay, then what?

Answer: Validity Frames instead of Boolean validity

Validity Frames

- A validity frame is a construct that is explicitly a function of the model/system's:
  - Properties of interest
  - The distance function
  - The distance threshold
  - And other modular activities
    - For example, the processVoI activity may be modular i.e. there may be different ways to compute the same PoI from the same VoI.
- It is a set of experimental frames

Abstract Frames



- Abstract Validity Frame (AVF)
  - The (possibly infinite) set of all experimental frames in which a model is valid

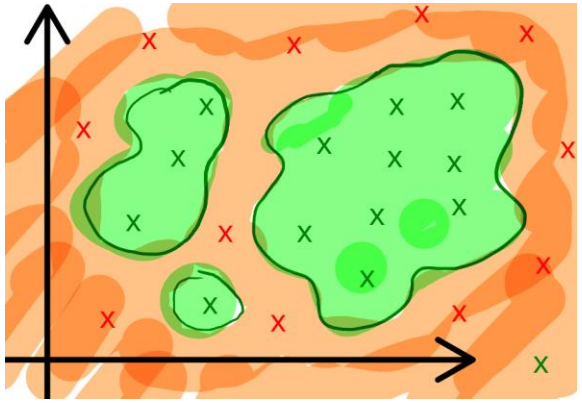
- Abstract Invalidity Frame (AIF)
  - The (possibly infinite) set of all experimental frames in which a model is invalid

$$AVF_{\mu_n} \cup AIF_{\mu_n} = \mathbb{U}_{\mu_n}$$

$$AVF_{\mu_n} \cap AIF_{\mu_n} = \varnothing$$

Acknowledgements to Rhys Goldstein, Autodesk Research

Concrete Frames

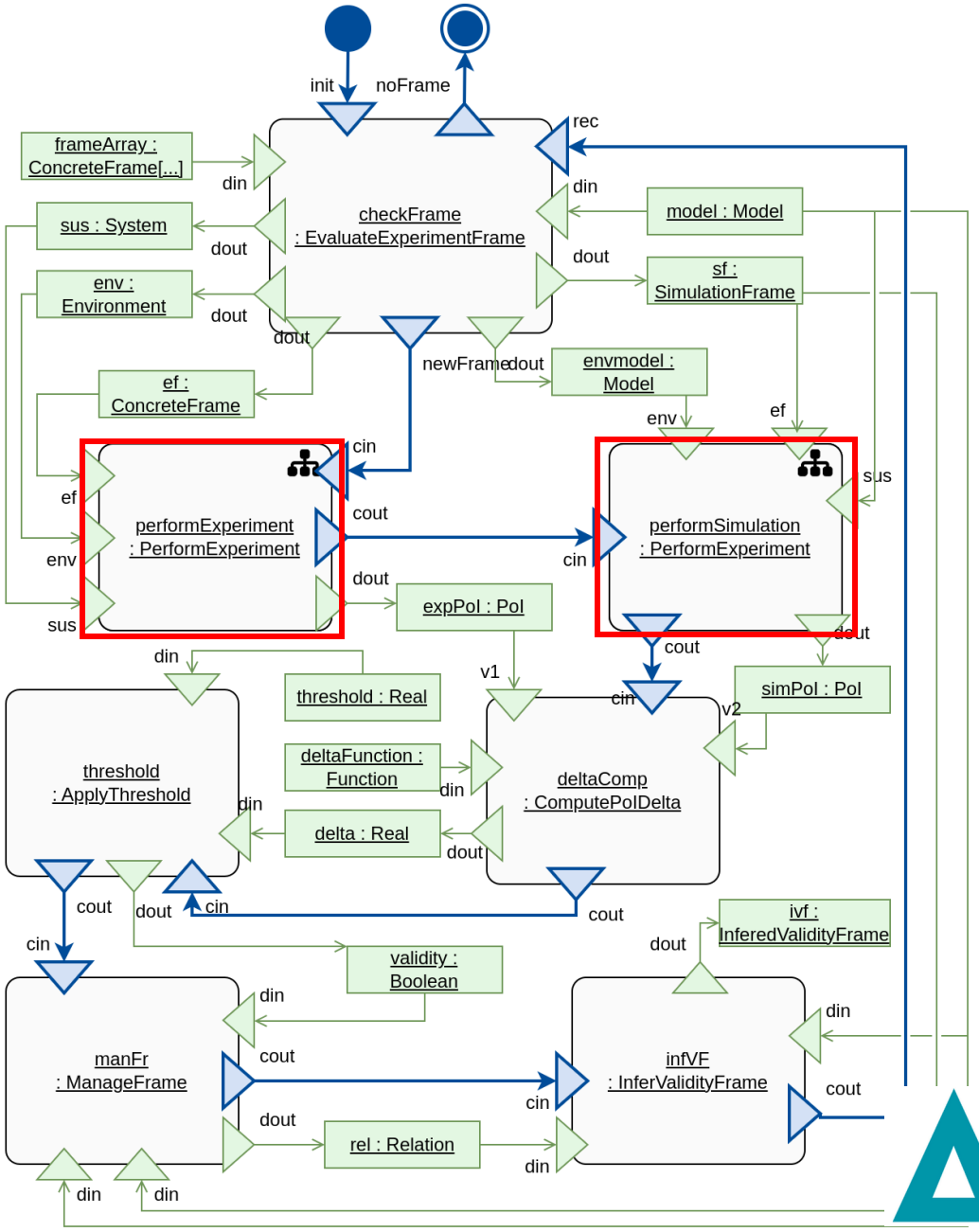


- Concrete Validity Frame (CVF)
  - The finite set of performed experimental frames in which a model is deemed valid
- Concrete Invalidity Frame (CIF)
  - The finite set of performed experimental frames in which a model is deemed invalid

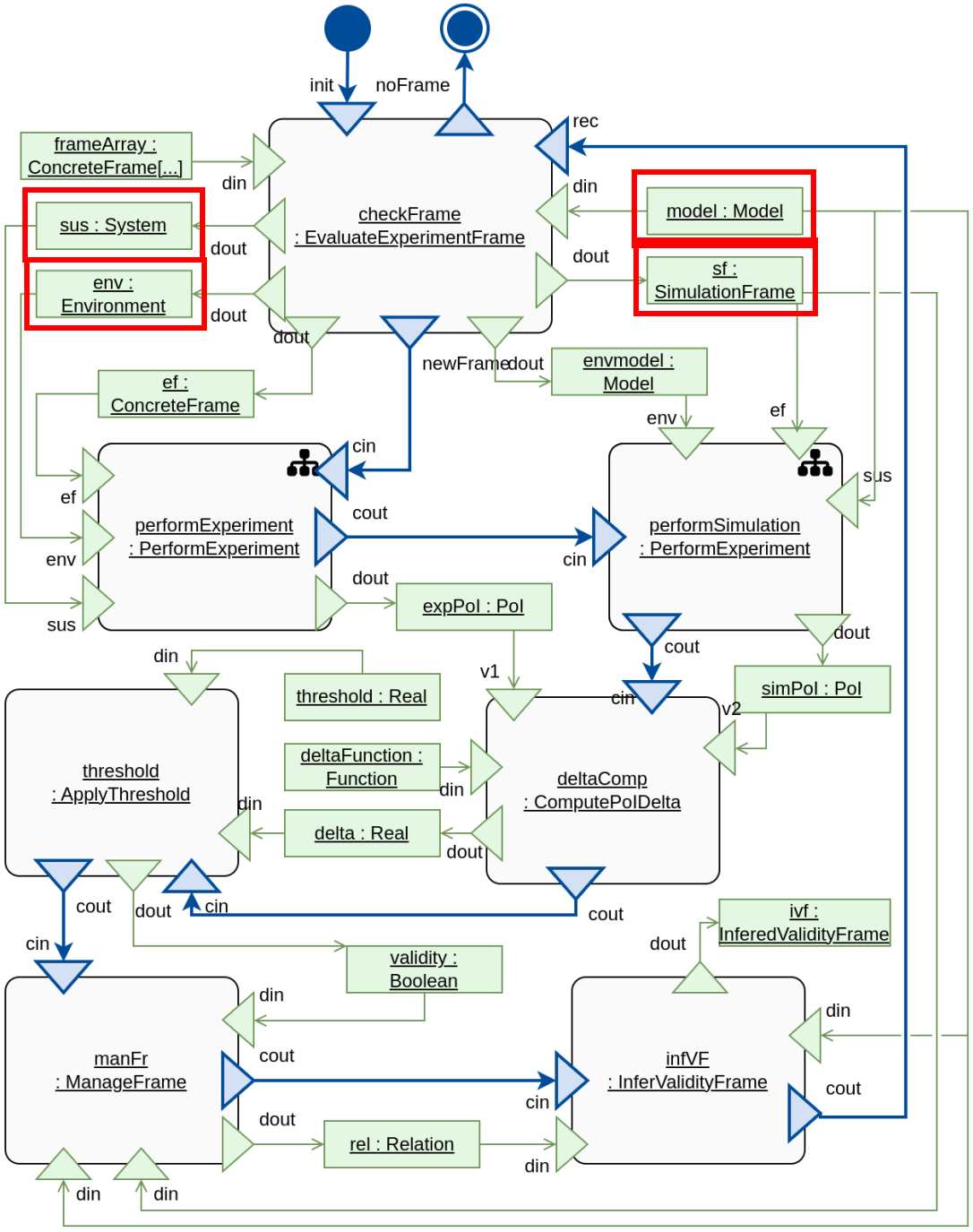
$$CVF_{\mu_n} \cap CIF_{\mu_n} = \varnothing$$

Validation experiment

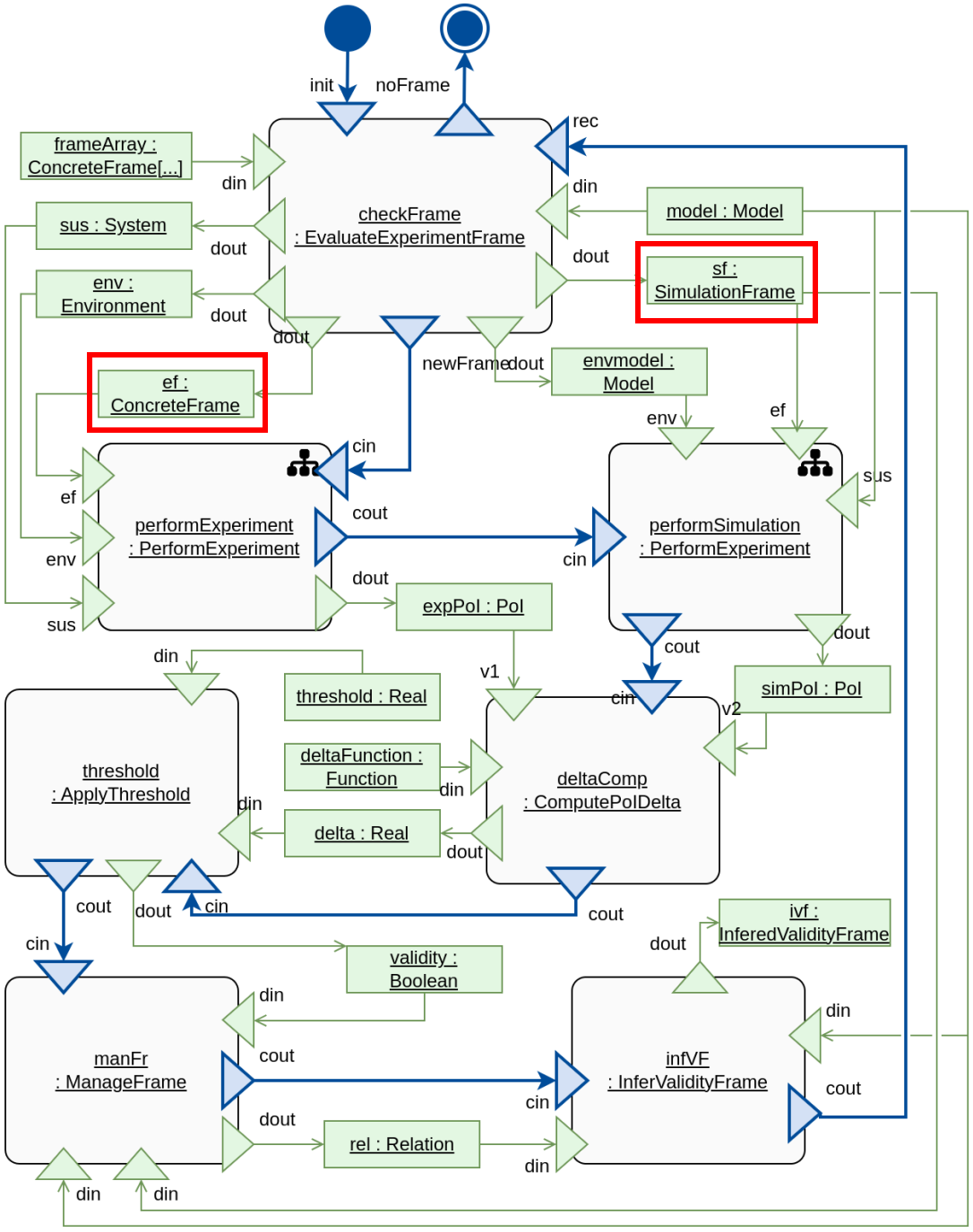
- At least one experiment and one simulation



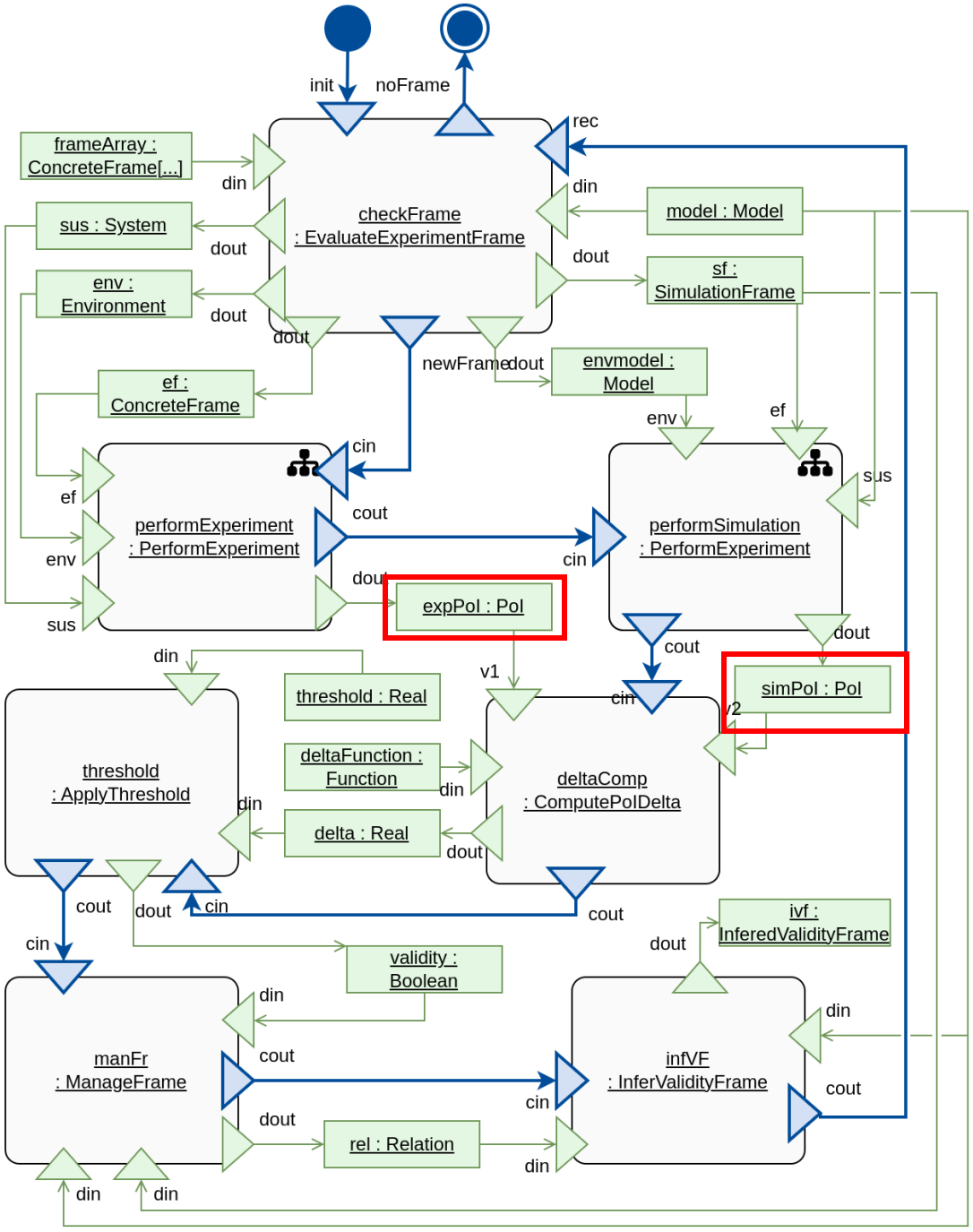
- At least one experiment and one simulation
- The model models the system



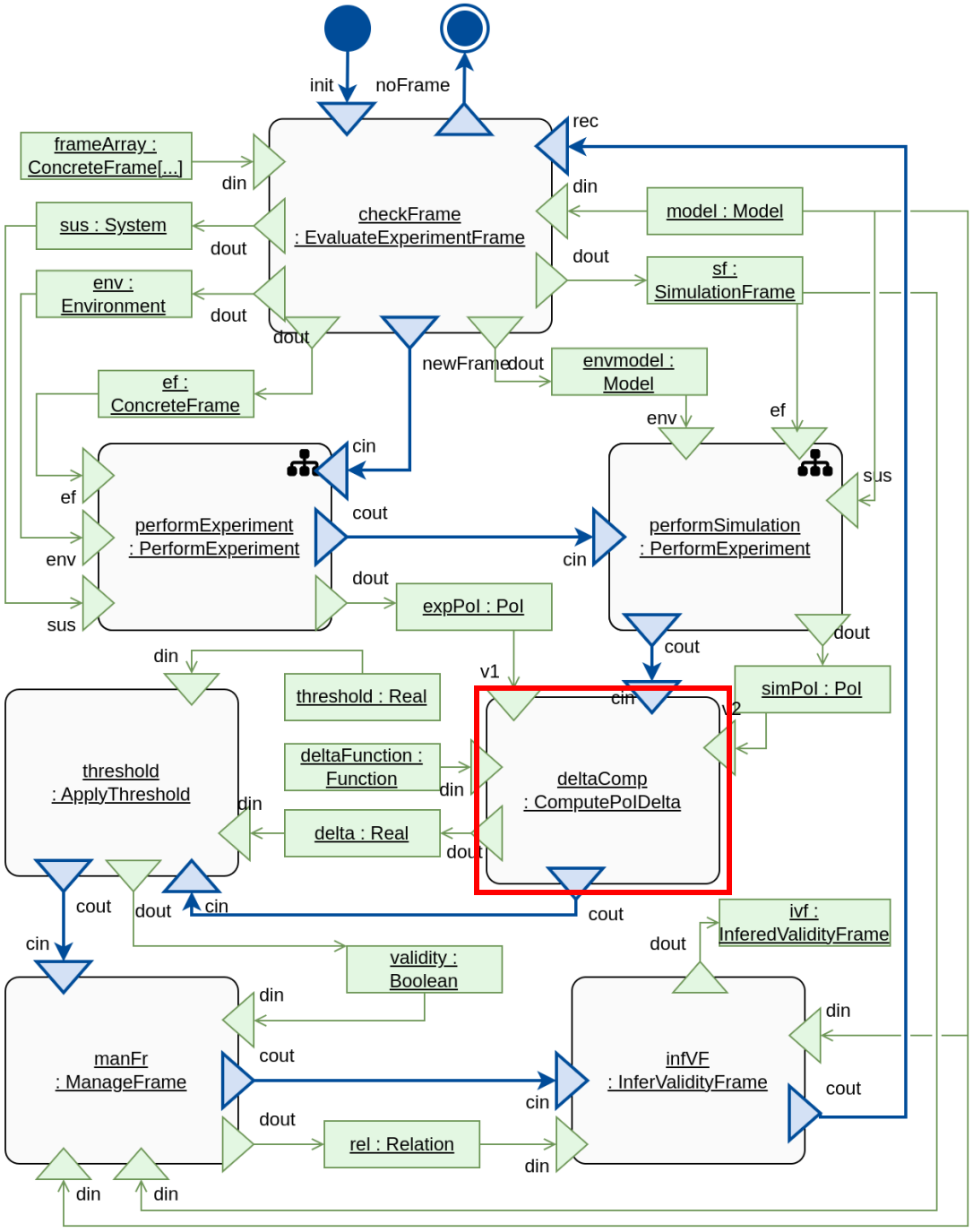
- At least one experiment and one simulation
- The model models the system
- The experimental and simulation frame should correspond



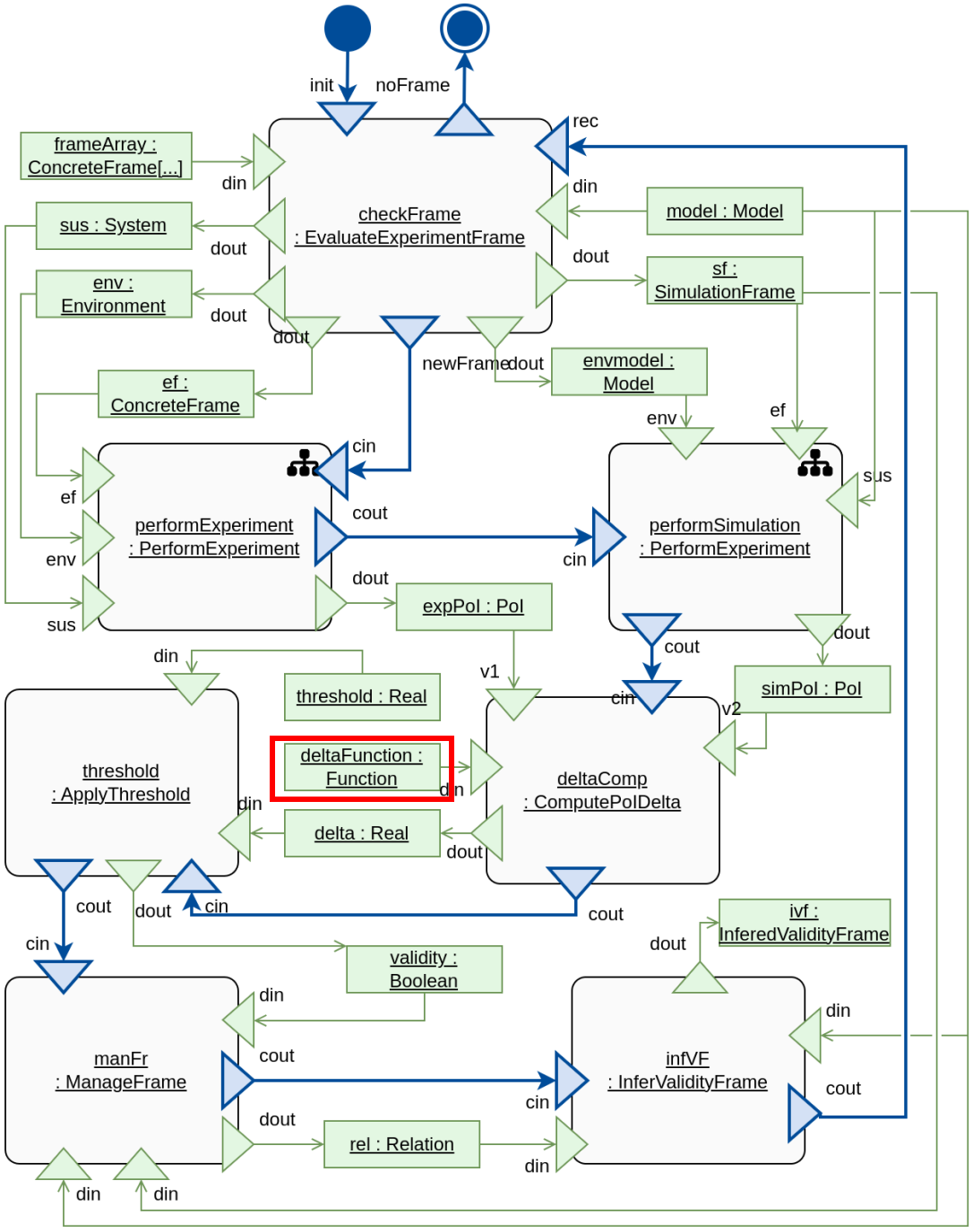
- At least one experiment and one simulation
- The model models the system
- The experimental and simulation frame should correspond
- The types of the PoIs should be the same



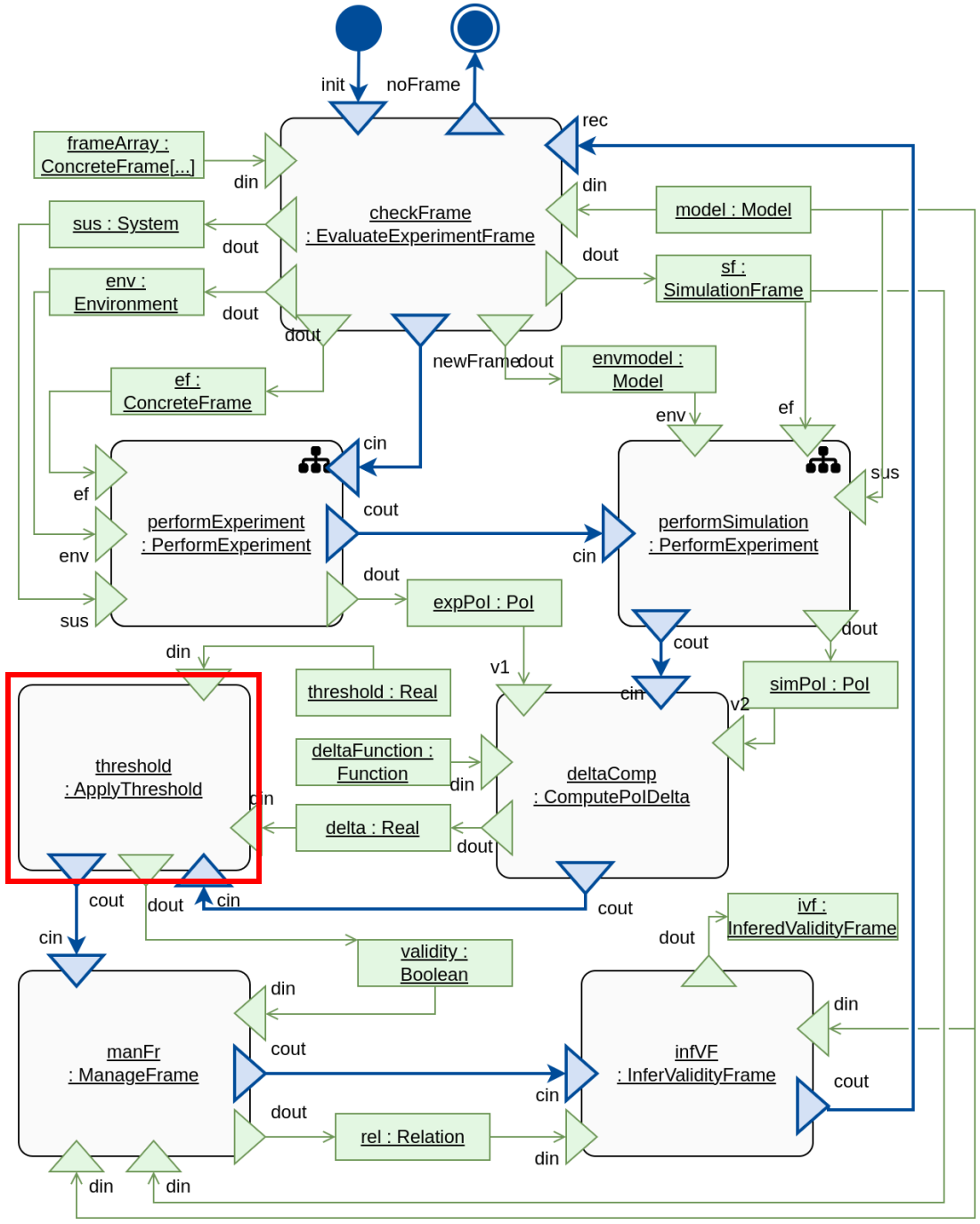
- Compute Delta from PoI



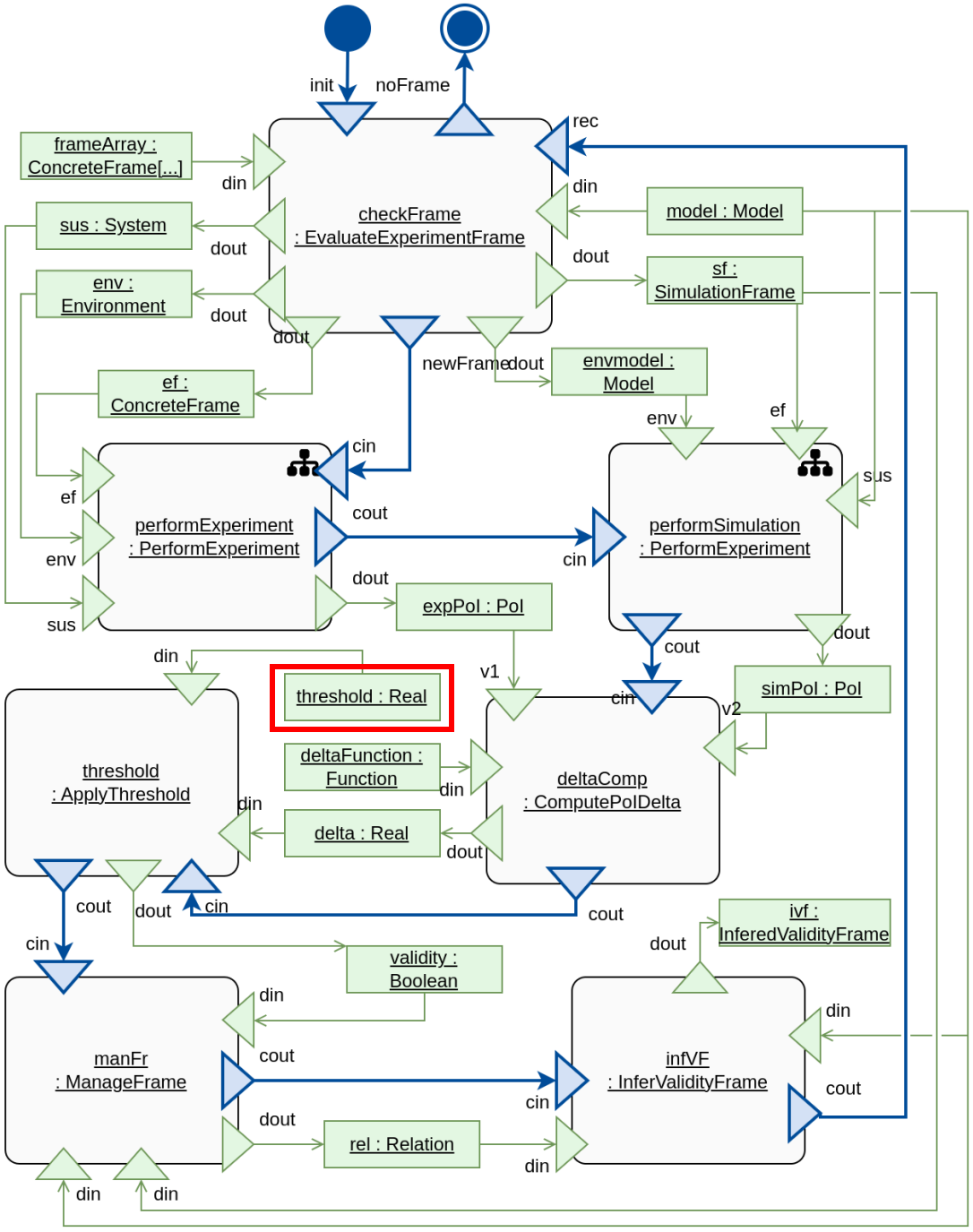
- Compute Delta from PoI
  - Based on a delta function



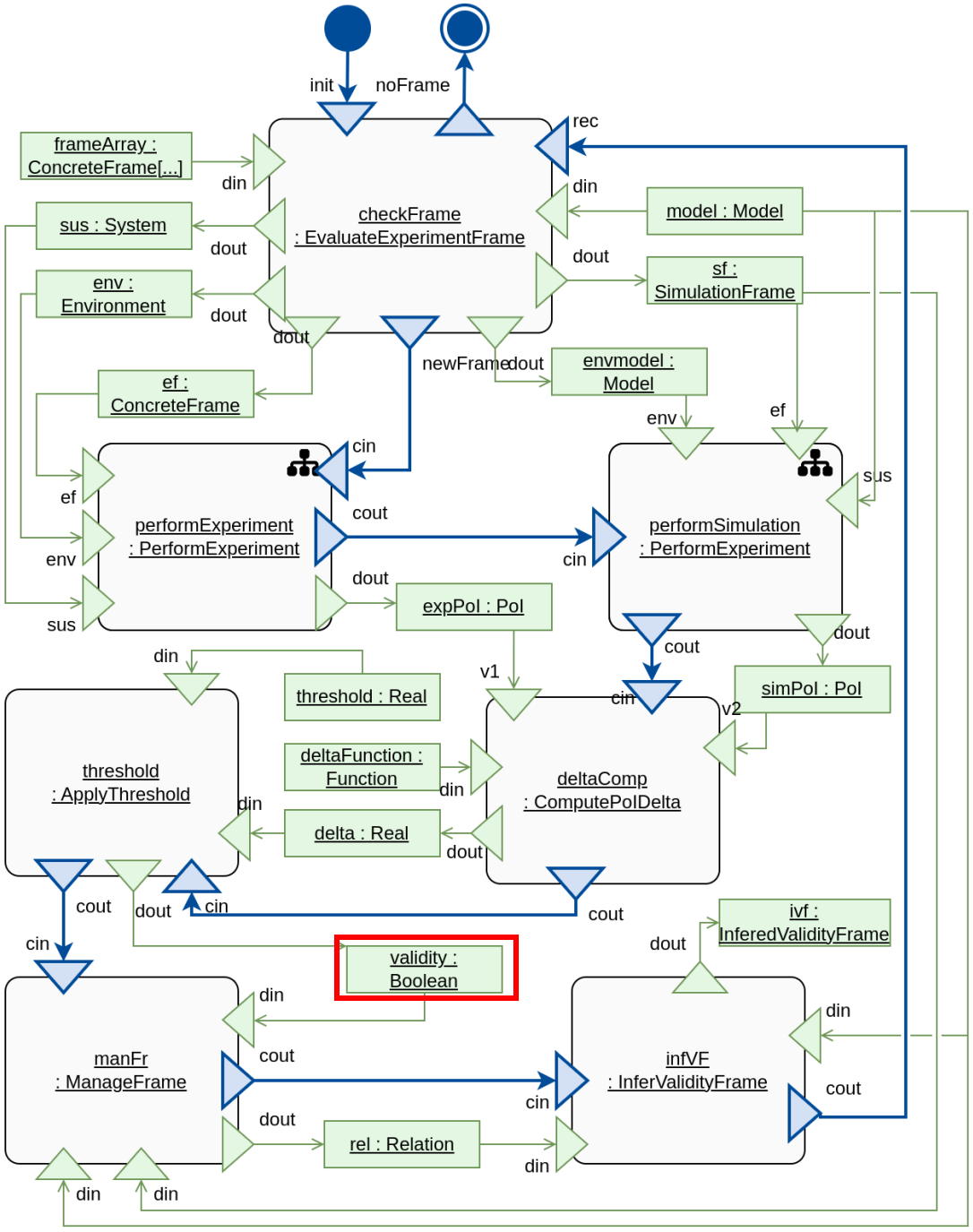
- Compute Delta from PoI
  - Based on a delta function
- Apply Threshold
  - Non-negative threshold



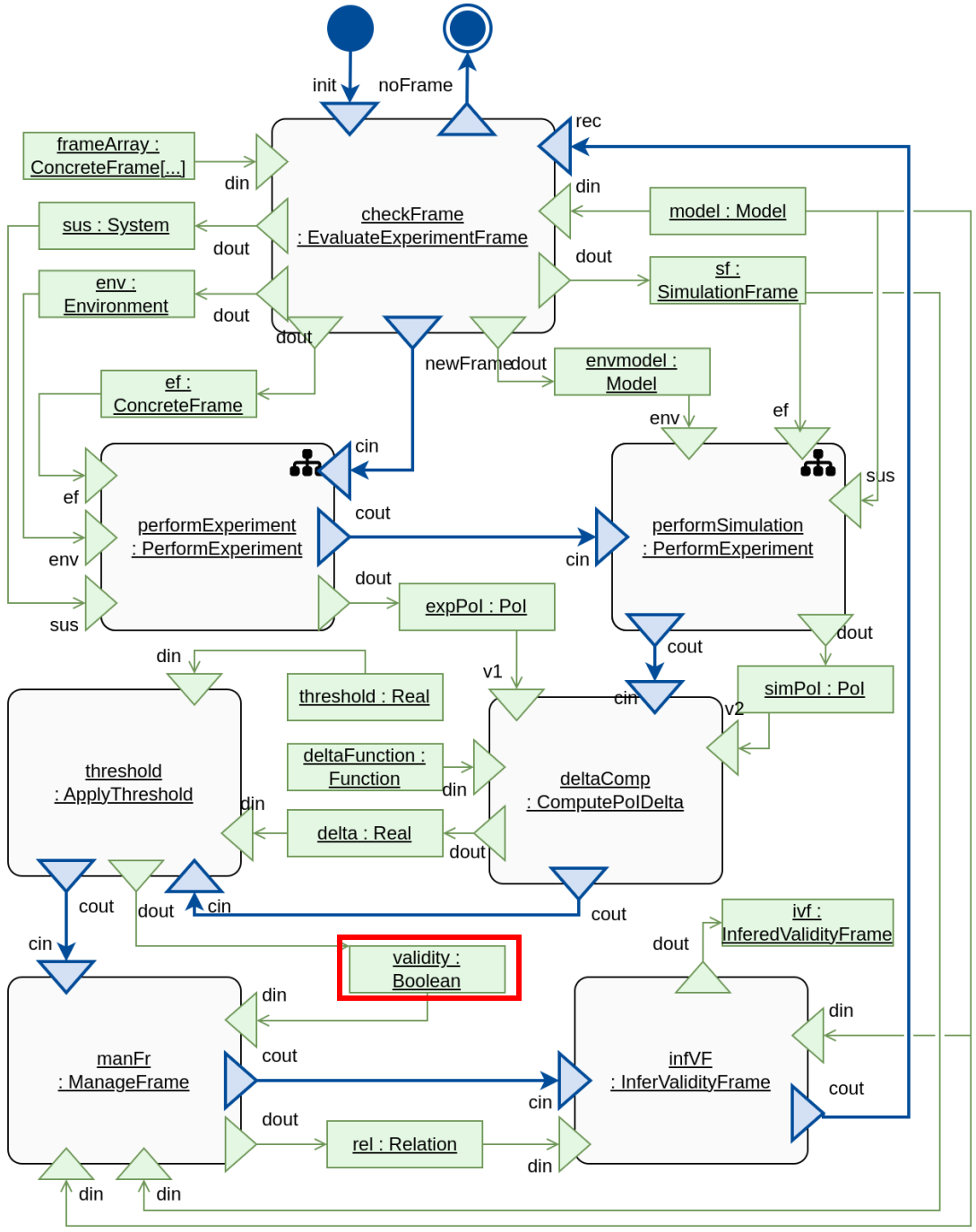
- Compute Delta from PoI
  - Based on a delta function
- Apply Threshold
  - Non-negative threshold
  - Decides the amplitude of delta



- Compute Delta from PoI
  - Based on a delta function
- Apply Threshold
  - Non-negative threshold
  - Decides the amplitude of delta
  - Boolean validity output

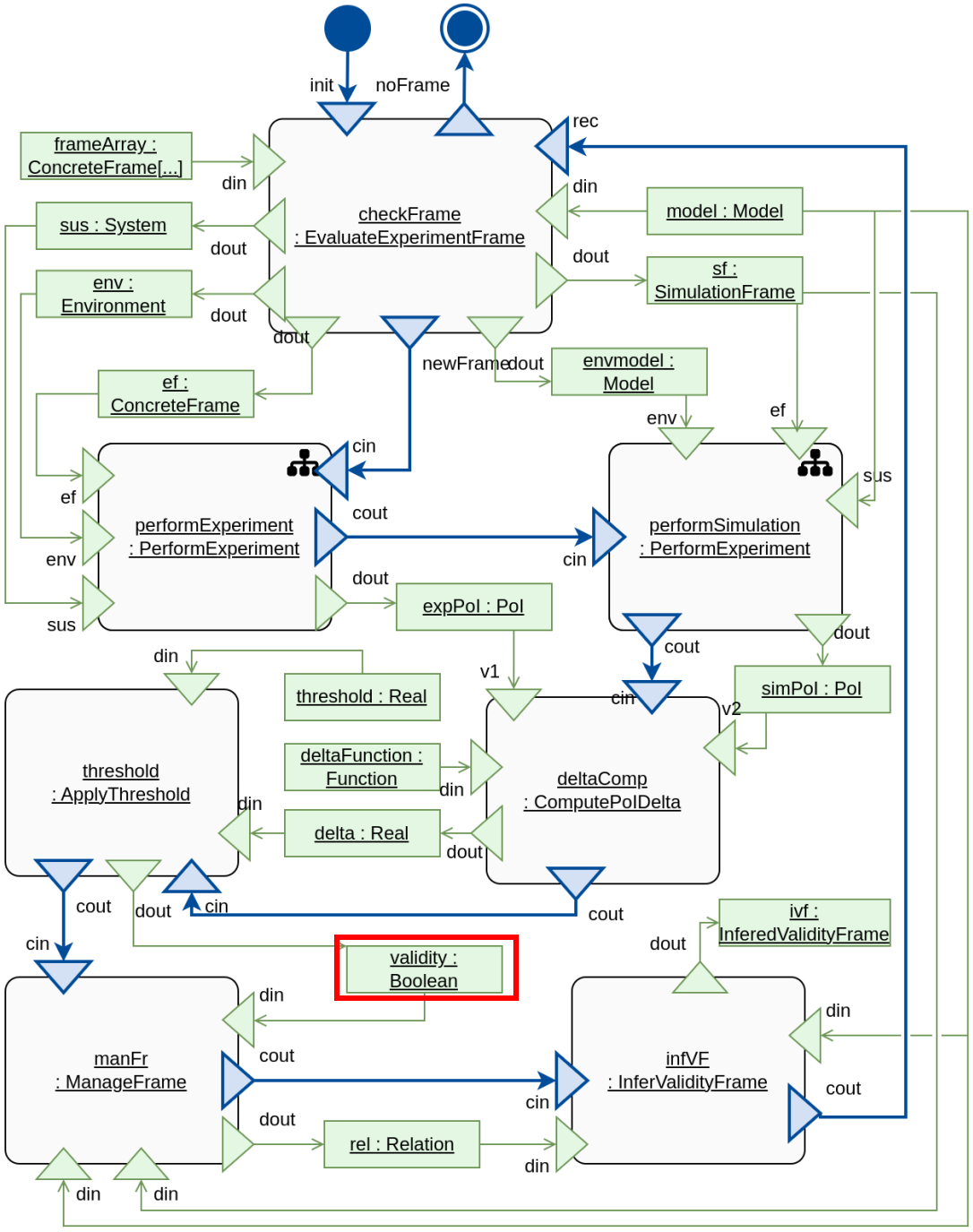


- Compute Delta from PoI
  - Based on a delta function
- Apply Threshold
  - Non-negative threshold
  - Decides the amplitude of delta
  - Boolean validity output
  - YAY! TASK COMPLETE!

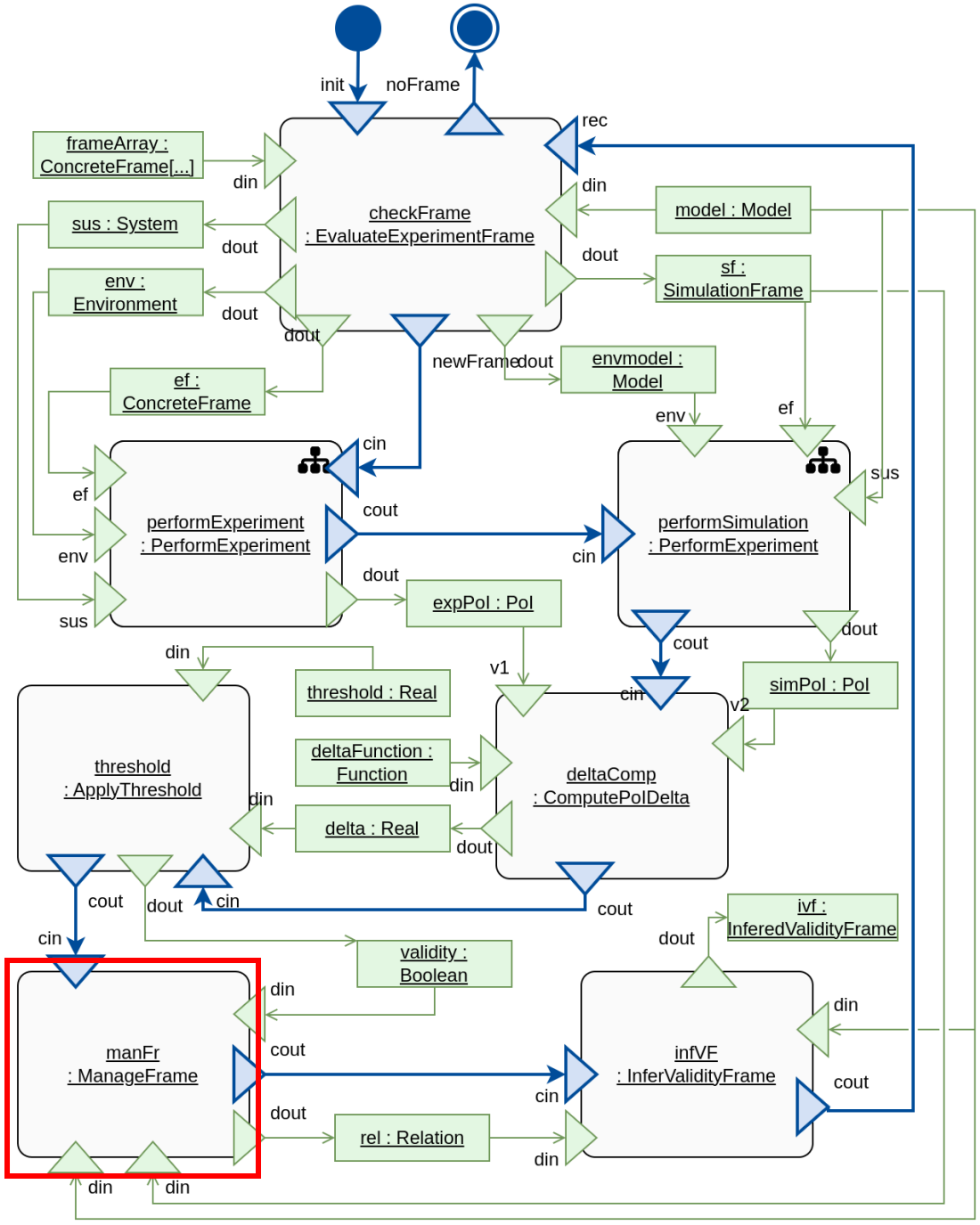


- Compute Delta from PoI
  - Based on a delta function
- Apply Threshold
  - Non-negative threshold
  - Decides the amplitude of delta
  - Boolean validity output
  - YAY! TASK COMPLETE!

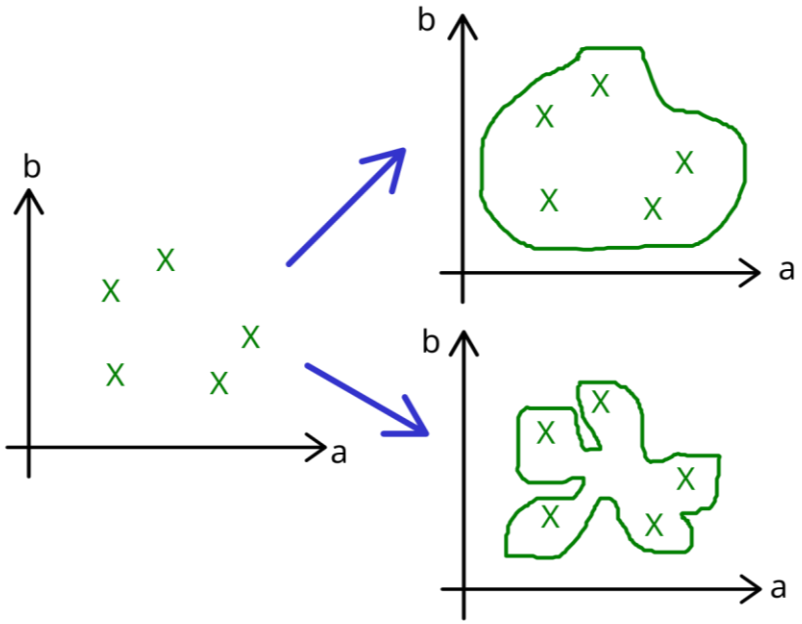
NO!

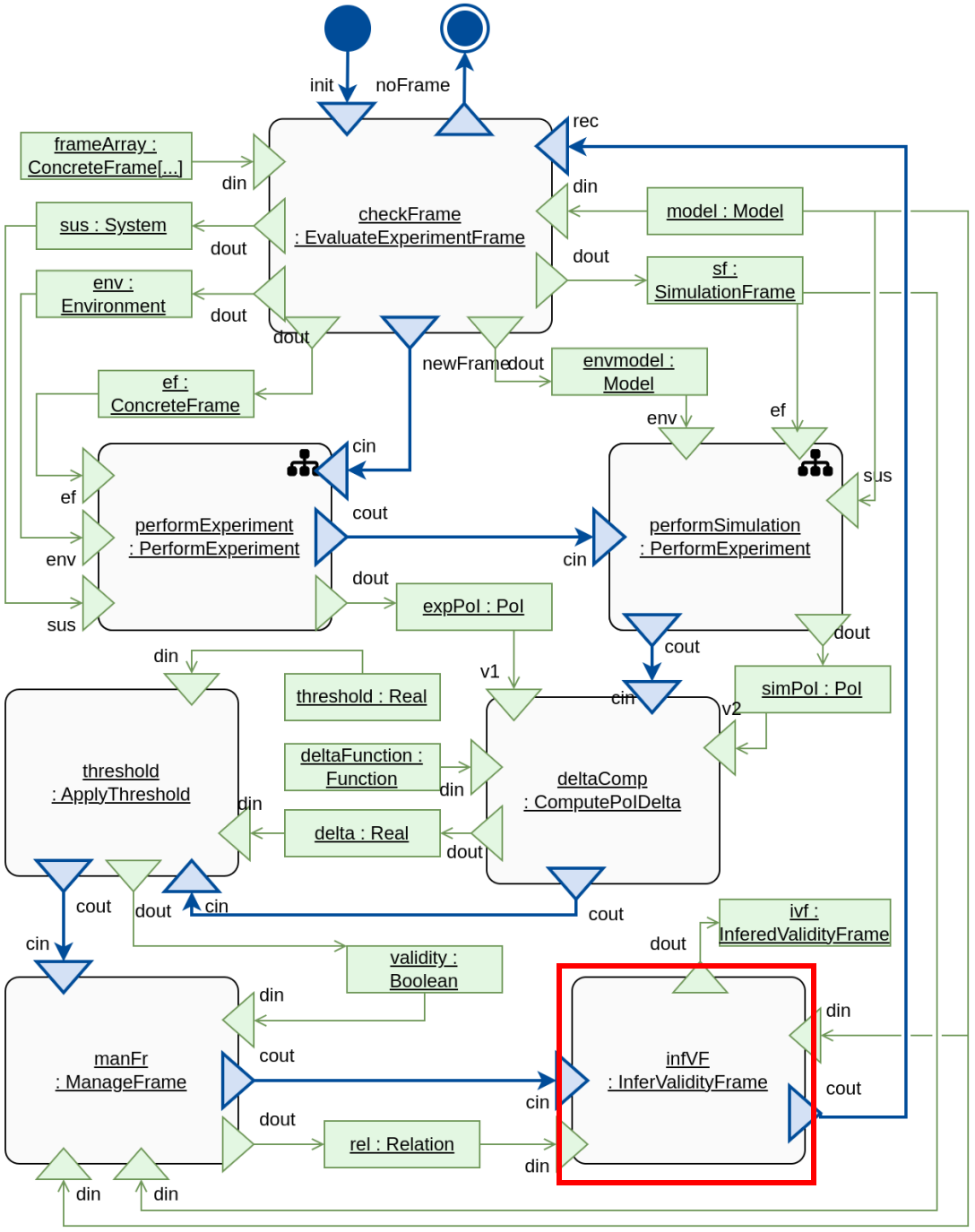


- Need to manage the information that we have! (Model management)
  - Create relations between
    - Concrete frame
    - Model
    - with references to
      - Variable Processing function
      - Distance function
      - Threshold
  - The relation is a validity or invalidity relation



- Make inferences from the concrete frame data





Inferred Frames

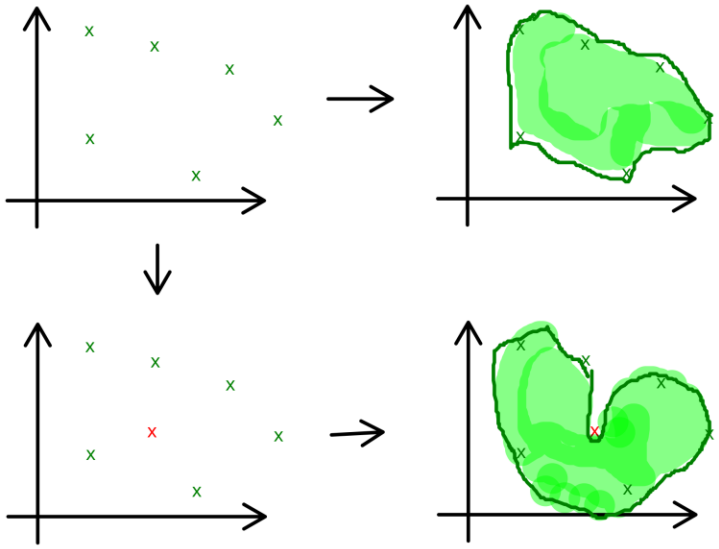
- Inferred Validity Frame (IVF)
  - The (possibly infinite) set of all experimental frames in which a model is assumed to be valid based on CVF information and an inferencing algorithm (based on domain-specific knowledge).

- Inferred Invalidity Frame (IIF)
  - The (possibly infinite) set of all experimental frames in which a model is assumed to be valid based on CIF information and an inferencing algorithm.

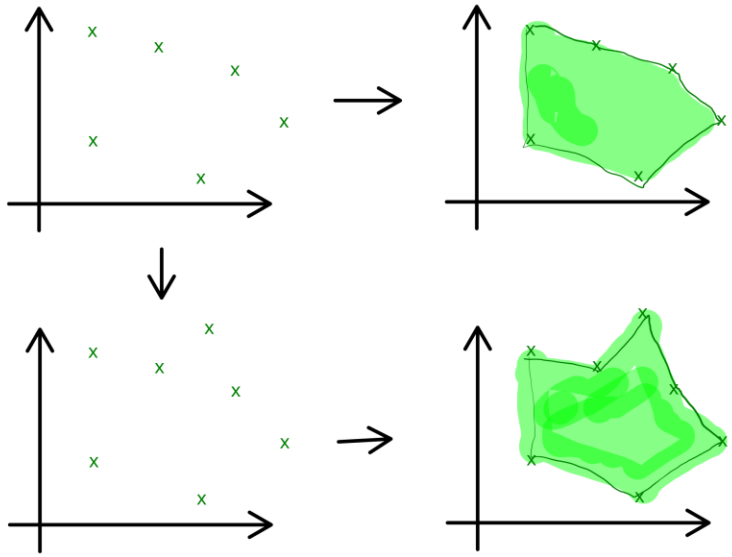
$$CVF_{\mu_n} \subset IVF_{\mu_n} \qquad CIF_{\mu_n} \subset IIF_{\mu_n}$$

$$CIF_{\mu_n} \cap IVF_{\mu_n} = \varnothing \qquad CVF_{\mu_n} \cap IIF_{\mu_n} = \varnothing$$

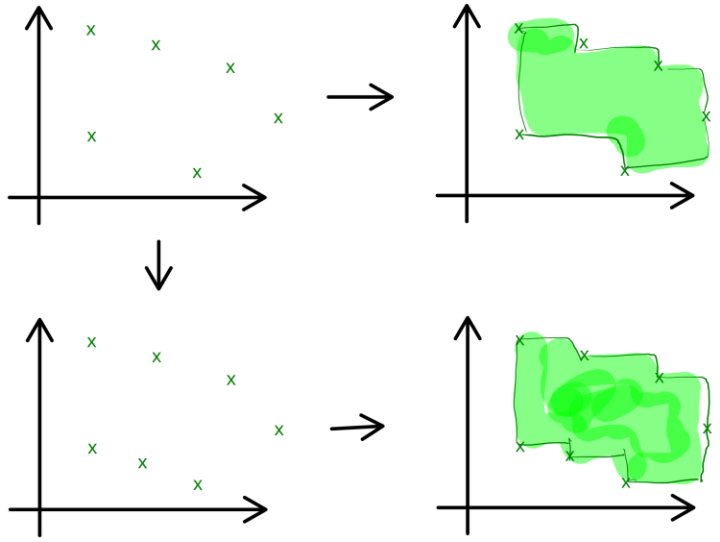
Updating the Inferred Frame



Case 1: Something wrong with inference proven by new experiment



Case 2: Extending the boundary of a conservative inferencing algorithm



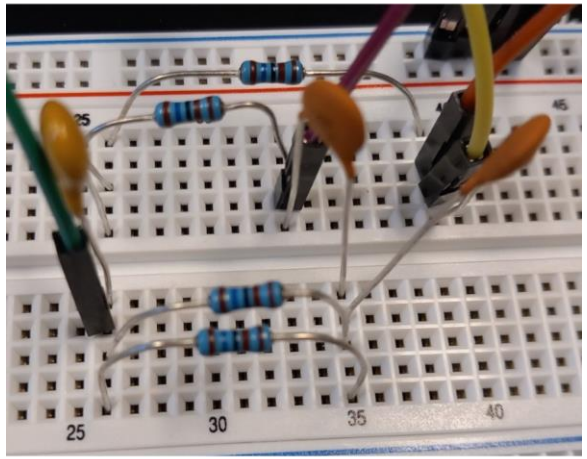
Case 3: Increasing the resolution of validity frame w.r.t parameters

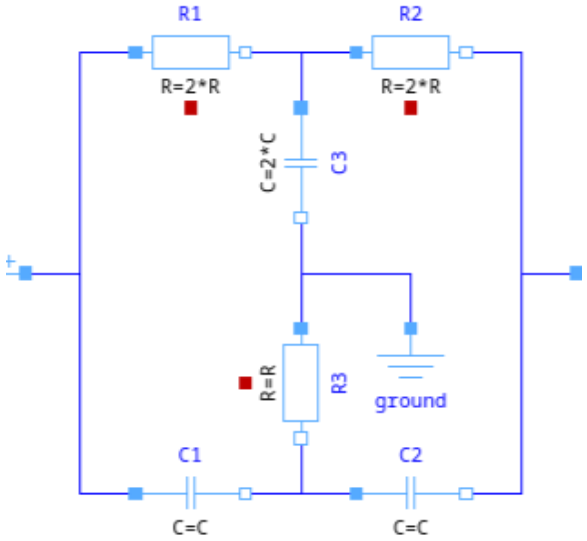
Updating the Inferred Frame

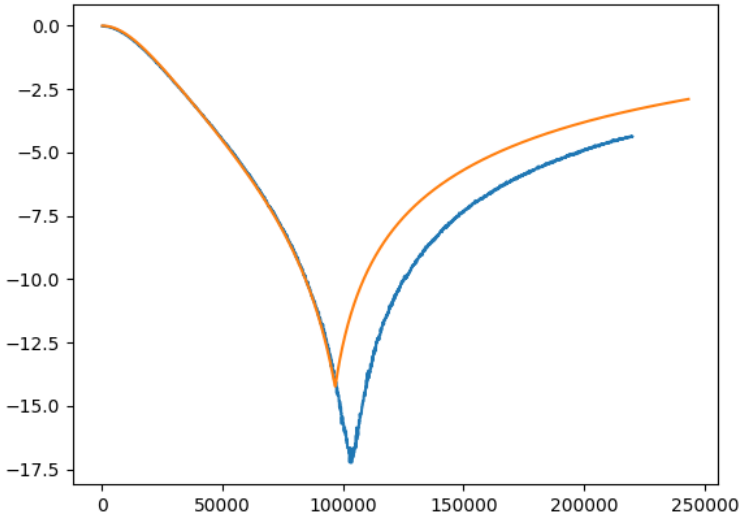
The task of the validation engineer becomes: "compute an inferred validity frame as close to the abstract validity frame as possible"

Case-study

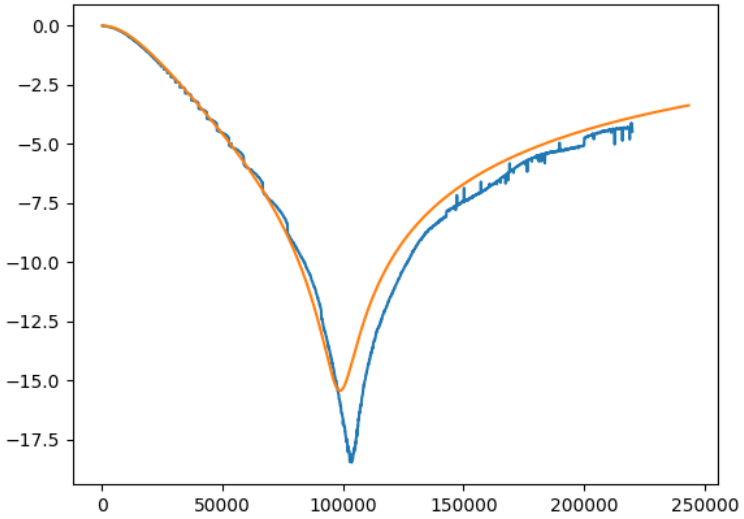
- Twin-T notch filter







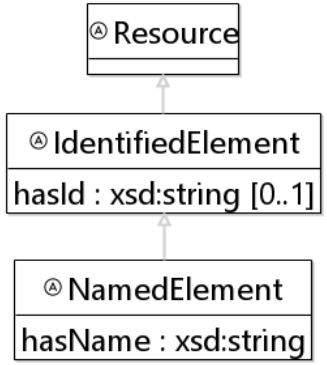
Gain =
Ratio of
amplitudes

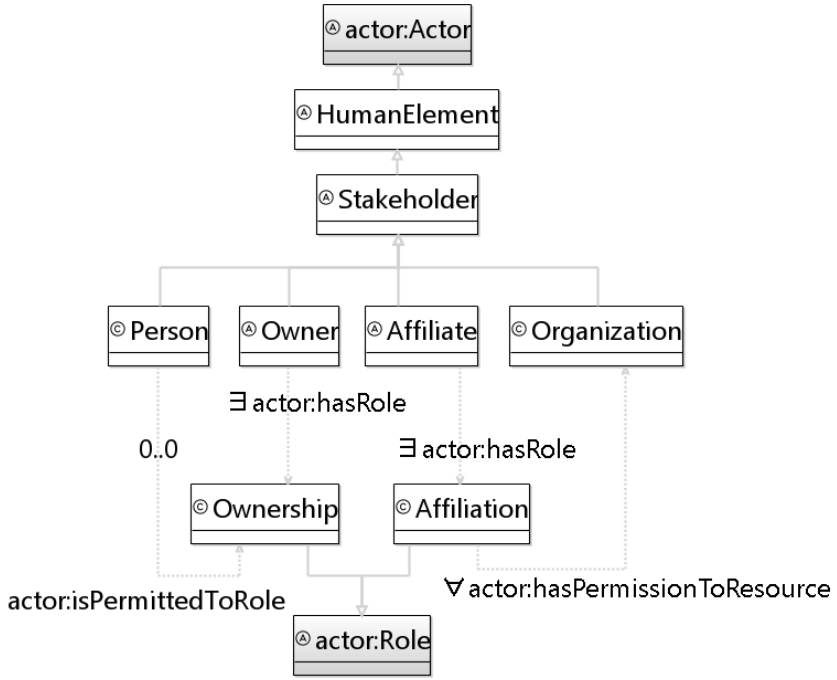


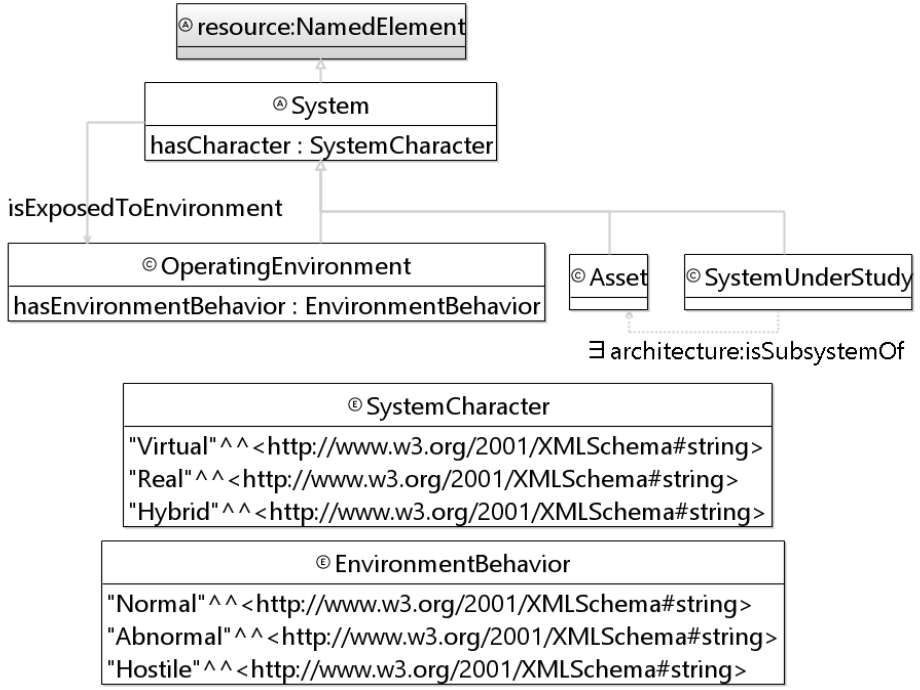
Gain =
Ratio of
RMS (power)

Automatically conducted and analysed ~10k experiments and simulations so far
(to go back and conduct many more!)

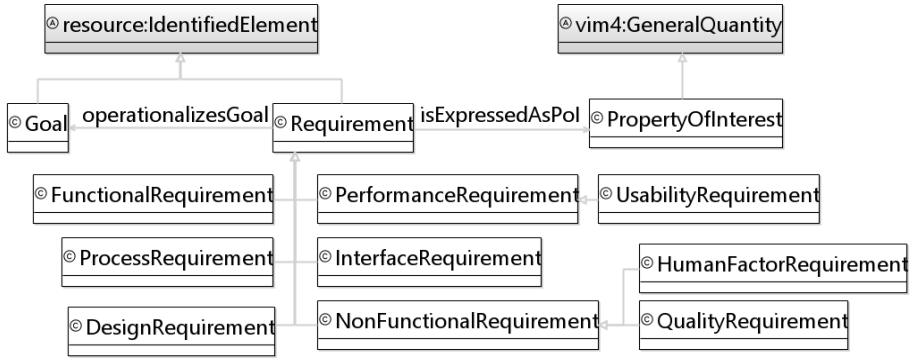
Ontology

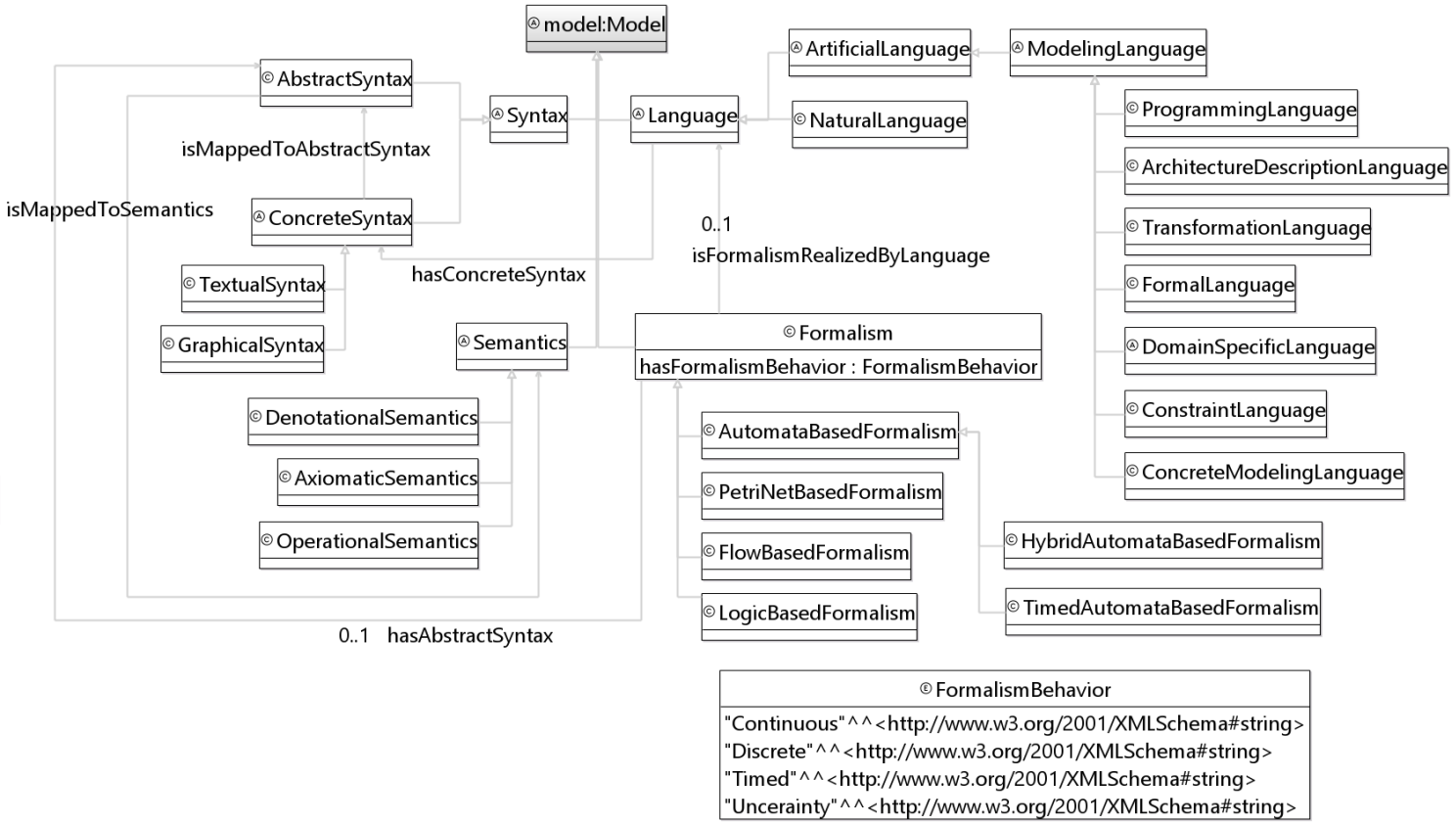

Resource
IdentifiedElement
hasId : xsd:string [0..1]
NamedElement
hasName : xsd:string


actor:Actor
HumanElement
Stakeholder
Person
Owner
Affiliate
Organization
∃ actor:hasRole
0..0
∃ actor:hasRole
Ownership
Affiliation
actor:isPermittedToRole
∀ actor:hasPermissionToResource
actor:Role


resource:NamedElement
System
hasCharacter : SystemCharacter
isExposedToEnvironment
OperatingEnvironment
hasEnvironmentBehavior : EnvironmentBehavior
Asset
SystemUnderStudy
∃ architecture:isSubsystemOf
SystemCharacter
"Virtual"^^<http://www.w3.org/2001/XMLSchema#string>
"Real"^^<http://www.w3.org/2001/XMLSchema#string>
"Hybrid"^^<http://www.w3.org/2001/XMLSchema#string>
EnvironmentBehavior
"Normal"^^<http://www.w3.org/2001/XMLSchema#string>
"Abnormal"^^<http://www.w3.org/2001/XMLSchema#string>
"Hostile"^^<http://www.w3.org/2001/XMLSchema#string>

Ontology


resource:IdentifiedElement
vim4:GeneralQuantity
Goal
operationalizesGoal
Requirement
isExpressedAsPoI
PropertyOfInterest
FunctionalRequirement
PerformanceRequirement
UsabilityRequirement
ProcessRequirement
InterfaceRequirement
HumanFactorRequirement
DesignRequirement
NonFunctionalRequirement
QualityRequirement


model:Model
AbstractSyntax
Syntax
Language
ArtificialLanguage
ModelingLanguage
NaturalLanguage
ProgrammingLanguage
ArchitectureDescriptionLanguage
isMappedToAbstractSyntax
isMappedToSemantics
ConcreteSyntax
TransformationLanguage
0..1
isFormalismRealizedByLanguage
hasConcreteSyntax
TextualSyntax
FormalLanguage
GraphicalSyntax
Semantics
Formalism
hasFormalismBehavior : FormalismBehavior
DomainSpecificLanguage
ConstraintLanguage
DenotationalSemantics
AutomataBasedFormalism
ConcreteModelingLanguage
AxiomaticSemantics
PetriNetBasedFormalism
OperationalSemantics
FlowBasedFormalism
HybridAutomataBasedFormalism
LogicBasedFormalism
TimedAutomataBasedFormalism
0..1
hasAbstractSyntax
FormalismBehavior
"Continuous"^^<http://www.w3.org/2001/XMLSchema#string>
"Discrete"^^<http://www.w3.org/2001/XMLSchema#string>
"Timed"^^<http://www.w3.org/2001/XMLSchema#string>
"Uncerainty"^^<http://www.w3.org/2001/XMLSchema#string>

DSL

- VaFL 'waffle' :
  - Validity Frame Language
- VESSL 'vessel' :
  - VaFL Experiment Specification Sub-Language

```
vessl_grammar = """
start: specification+

specification: "specification" CNAME [extends] "{" specifier+ "};"

extends: "extends" CNAME

specifier: "time_base" time_base ";"            -> time_base
         | "assumptions" "{" observable+ "};"   -> assumptions
         | "sys_in" "{" observed+ "};"          -> sys_in
         | "sys_out" "{" observed+ "};"         -> sys_out
         | "sys_state" "{" observed+ "};"       -> sys_state
         | "env_state" "{" observed+ "};"       -> env_state
         | "environment" CNAME ";"              -> environment

time_base: ("R"|"Z"|NUMBER"*Z")["+"]

observed: "{"observable_name","dimensions","[unit]","resolution"}"

resolution: FLOAT

unit: CNAME

observable: "{"observable_name","dimensions"}"

dimensions:"["[time]","[length]","[mass]","[current]","[temperature]","[amount]","[luminosity]"]"

time: SIGNED_NUMBER

length: SIGNED_NUMBER

mass: SIGNED_NUMBER

current: SIGNED_NUMBER

temperature: SIGNED_NUMBER

amount: SIGNED_NUMBER

luminosity: SIGNED_NUMBER

observable_name: CNAME

%import common.CNAME
%import common.SIGNED_NUMBER
%import common.FLOAT
%import common.WS
%import common.NUMBER
%ignore WS
"""
```

Thanks

Questions?

Rakshit.Mittal@uantwerpen.be

end