(SysML 2)
Semantics without ~~Tears~~ Math

Conrad Bock
U.S. National Institute of Standards and Technology

Ed Seidewitz
Model Driven

Overview

- Motivation / Problem
  - Modeling Languages and Analysis
  - Interpreting Models (Semantics)
- Solution
  - Standardizing Semantics
  - Logical Classification
  - Semantics, Without Math
  - SysML 2 Semantics
- Summary

Overview

- **Motivation / Problem**
  - **Modeling Languages and Analysis**
  - Interpreting Models (Semantics)
- Solution
  - Standardizing Semantics
  - Logical Classification
  - Semantics, Without Math
  - SysML 2 Semantics
- Summary

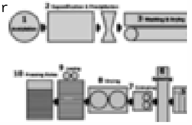
Modeling

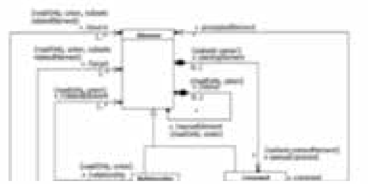
Language Developers

(using *example models*)

```
    assoc BinaryLink specializes Link {
 feature participant: Anything[2] nonunique
  end feature source: Anythi
  end feature target: Anythi

 feature sourceParticipant:
super
```





What are they imagining for system operation?

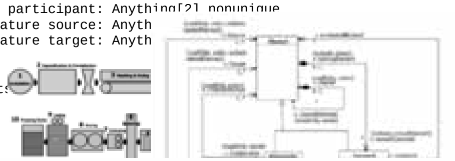
Modeling

Language Developers

(using *example models*)

```
     assoc BinaryLink specializes Link {
feature participant: Anything[2] nonunique
 end feature source: Anyth
 end feature target: Anyth

feature
supersets
```



What are they imagining for system operation?

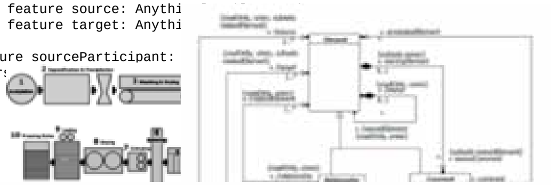
Analysis

Analysis Tool Builders

(incl execution, simulation, reasoning, etc)

```
     assoc BinaryLink specializes Link {
feature participant: Anything[2] nonunique
 end feature source: Anythi
 end feature target: Anythi

feature sourceParticipant:
supers
```



What should tools predict for system operations?

Modeling and Analysis

Language Developers

(using *example models*)

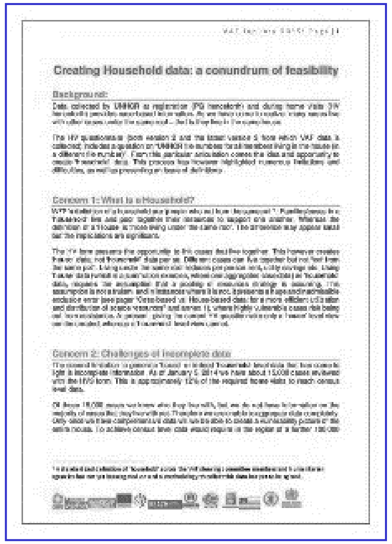
```
    assoc BinaryLink specializes Link {
feature participant: Anything[2] nonunique
 end feature source: Anyth
 end feature target: Anyth

feature
supersets
```

What is imagined for system operation?

Don’t know each other

Communicate only
through a standards spec



VAF [illegible] Page | 1

Creating Household data: a conundrum of feasibility

Background:

[illegible]

Concern 1: What is a Household?

[illegible]

Concern 2: Challenges of incomplete data

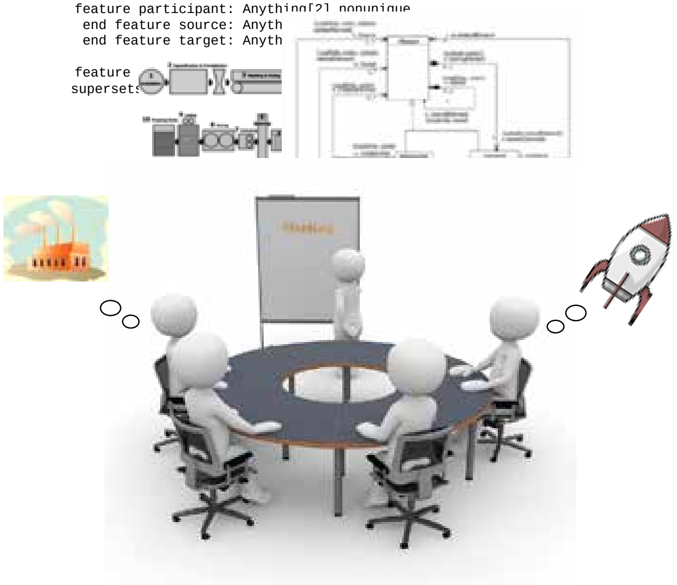
[illegible]

Analysis Tool Builders

(incl execution, simulation, reasoning, etc)

```
    assoc BinaryLink specializes Link {
feature participant: Anything[2] nonunique
 end feature source: Anyth
 end feature target: Anyth

feature
supersets
```



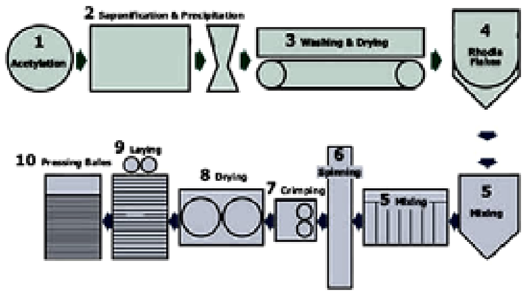
What should tools predict for system operations?

Overview

- Motivation / Problem
  - Modeling Languages and Analysis
  - Interpreting Models (Semantics)
- Solution
  - Standardizing Semantics
  - Logical Classification
  - Semantics, Without Math
  - SysML 2 Semantics
- Summary

Problem: Interpreting Models

Engineering model

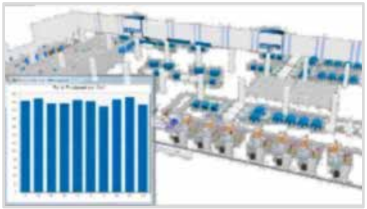


Interpreting a

Model

Things being specified

(as constructed & operated)



? Yes / No

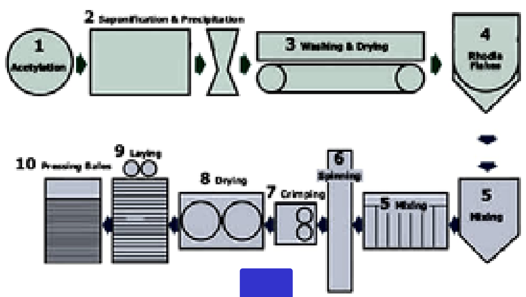
(Semantics)

- How do we know whether real or virtual things built & operated to a model follow the model?
- = Semantics (a boolean check)

Producing Real/Virtual Things from Models

Model

Engineering model



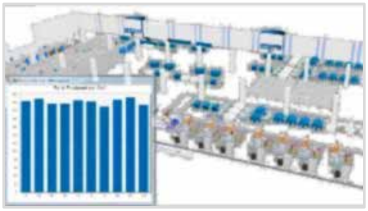
Interpreting a Model

Produce Real or Virtual Operated Thing

? Yes / No (Semantics)

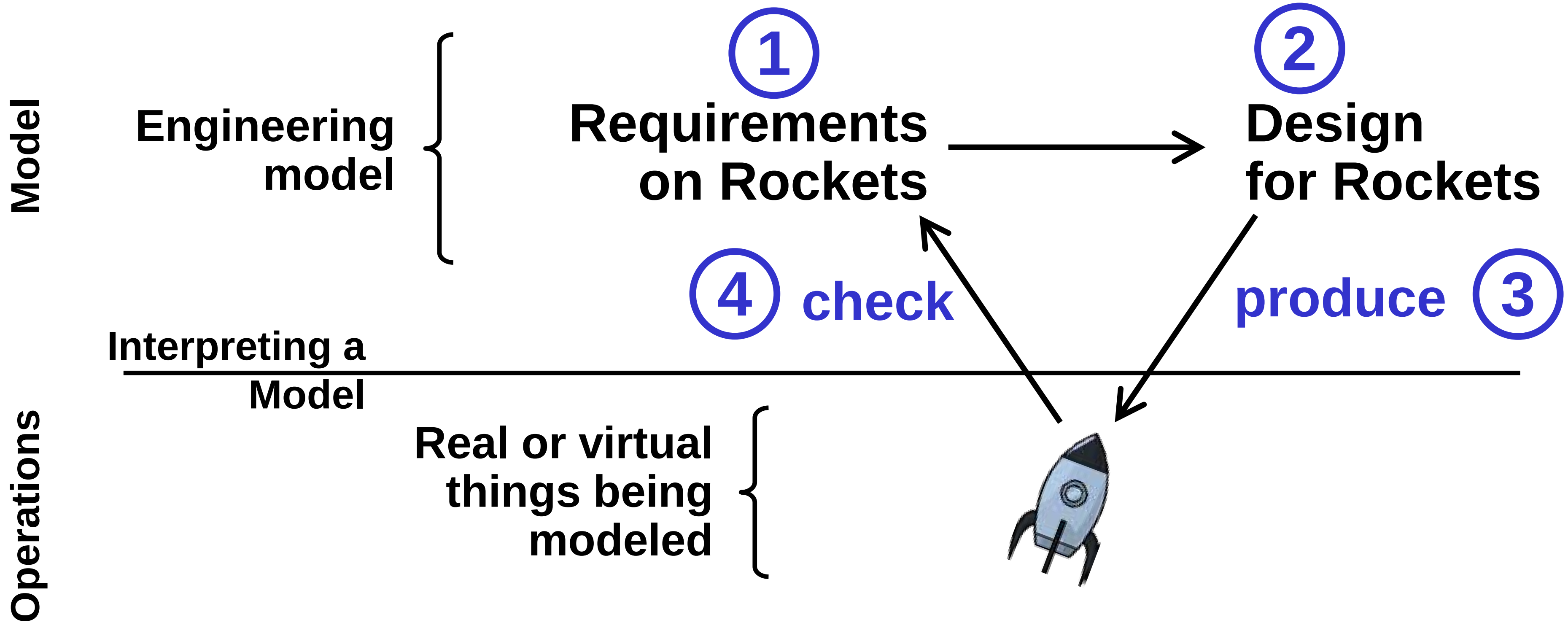
Operations

Things being modeled



§ **Check resulting things/operations using semantics.**

Requirements & Designs



Model

Engineering model

1 Requirements on Rockets

2 Design for Rockets

4 check

produce 3

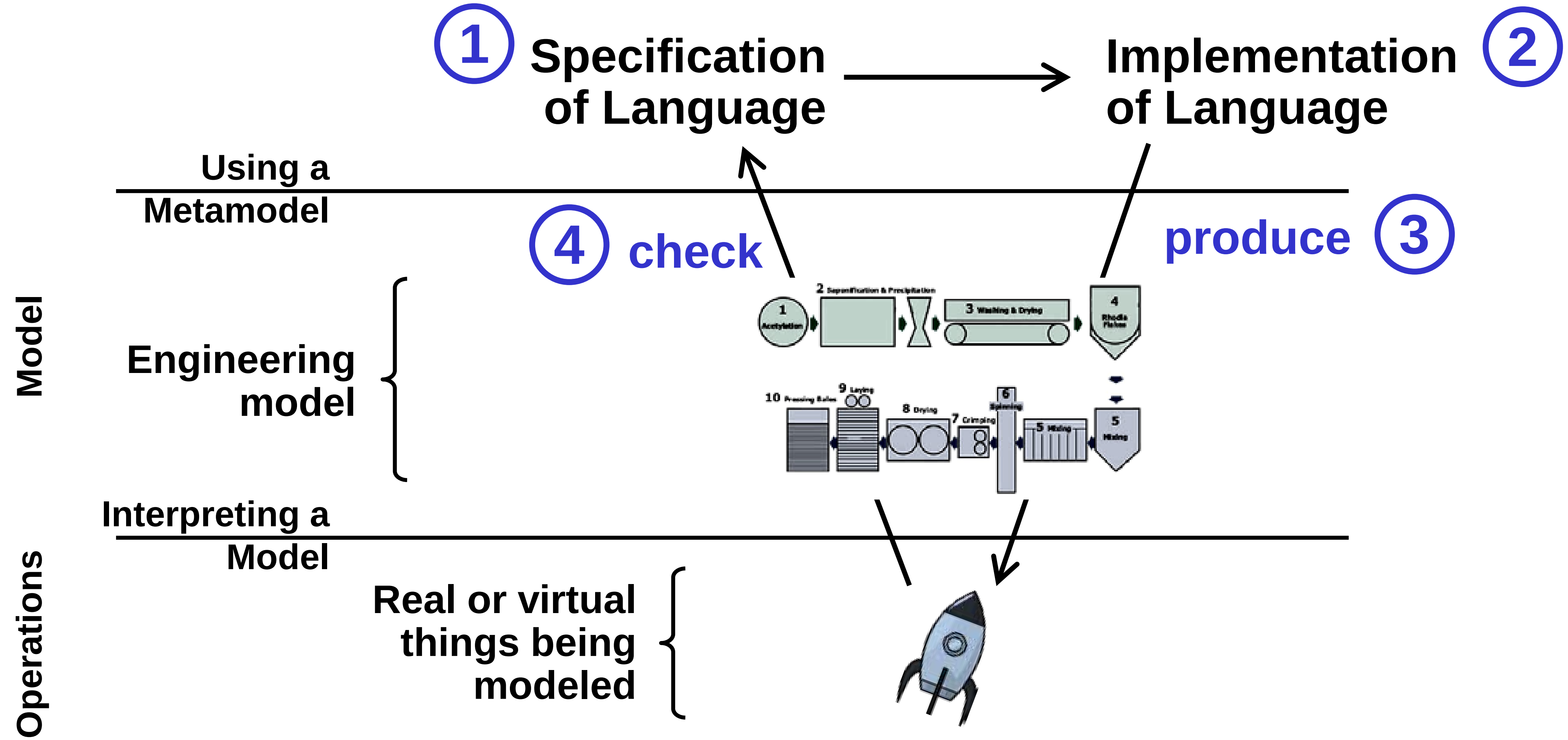
Interpreting a Model

Operations

Real or virtual things being modeled

- Do real/virtual systems **built** meet requirements?

Language Specs & Implementations



1 Specification of Language → Implementation of Language 2

Using a Metamodel

4 check

produce 3

Model

Engineering model

Interpreting a Model

Operations

Real or virtual things being modeled

- Do modeling/analysis tools meet the language spec?

Systems Engineering for Languages

- **SE involves multiple kinds of specifications:**
  - **Intended effects of a system (requirements)**
  - **How the system will bring about the effects (designs)**
  - **Procedures for testing real or virtual systems built and operated according to a design (tests).**

Systems Engineering	**Modeling Languages**
Requirements	Semantics
Designs	Analysis Tools
Tests	Semantic Conformance

Logical Terms: Inference and Semantics

- **Produce** real of virtual things
  - Execution
    - Incremental creation, usually deterministic and time ordered.
  - Simulation
    - Less deterministic execution.
    - Aggregate measures of probable executions.
  - Reasoning
    - Search based directly on semantics.

Kinds of inference
(logically speaking)

- **Check results** based on model + language semantics.

Semantics
(logically speaking)

See Section 3.1 (Intro to Reasoning) in Bock, et al, “Evaulating Resoning Systems, NIST 7310 https://www.nist.gov/publications/evaluating-reasoning-systems

The “S” Word

- One meaning used here: how to tell when …
  - a real or virtual thing (as contructed and operated) …
  - “follows” (conforms to) a model …
  - … written in a particular language.
- “How to tell” =
  - procedure resulting in true or false when applied to real or virtual thing/operation.
  - Conditions that must be met by operated thing.
- Compare to
  - Application vocabulary (lathes, drills, etc).
  - Model development methods (requirements, designs).

Overview

- Motivation / Problem
  - Modeling Languages and Analysis
  - Interpreting Models (Semantics)
- **Solution**
  - **Standardizing Semantics**
  - Logical Classification
  - Semantics, Without Math
  - SysML 2 Semantics
- Summary

Standardizing Conformance, Syntactic

Standard

Modeling language and libraries

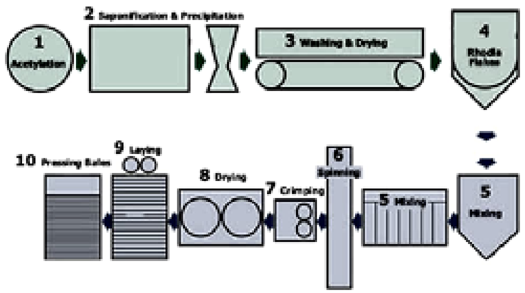
- Graphics:
  - Circles
  - Lines
  - Rectangles
- Domain terms:
  - Lathes, Feeders
  - Drying, Shaping
- Using terms:
  - Connect a feeder to a lathe

Syntactic conformance

Using a standard

Model

Engineering model



- Typical “instance checking”
  - between metamodel and model
  - specified in the usual way (classes, properties, constraints)

Standardizing Conformance, Semantic

Standard

Modeling language and libraries

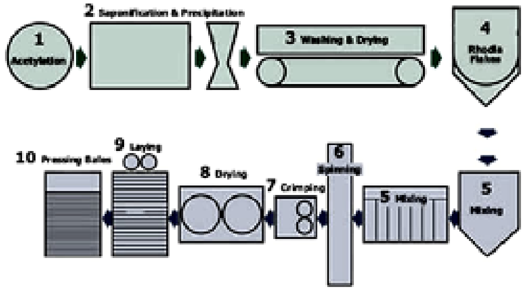
- Graphics:
  - Circles
  - Lines
  - Rectangles
- Domain terms:
  - Lathes, Feeders
  - Drying, Shaping
- Using terms:
  - Connect a feeder to a lathe

- **What happens**:
  - Geometry changed
  - Pieces mounted onto machine
  - Water removed

Using a standard

Specification

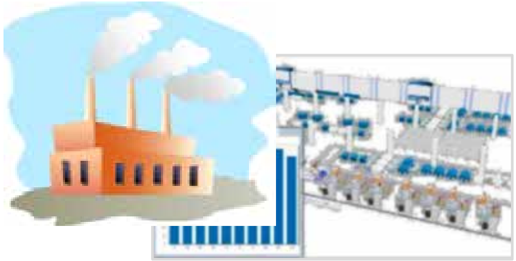
Engineering model



Interpreting a specification

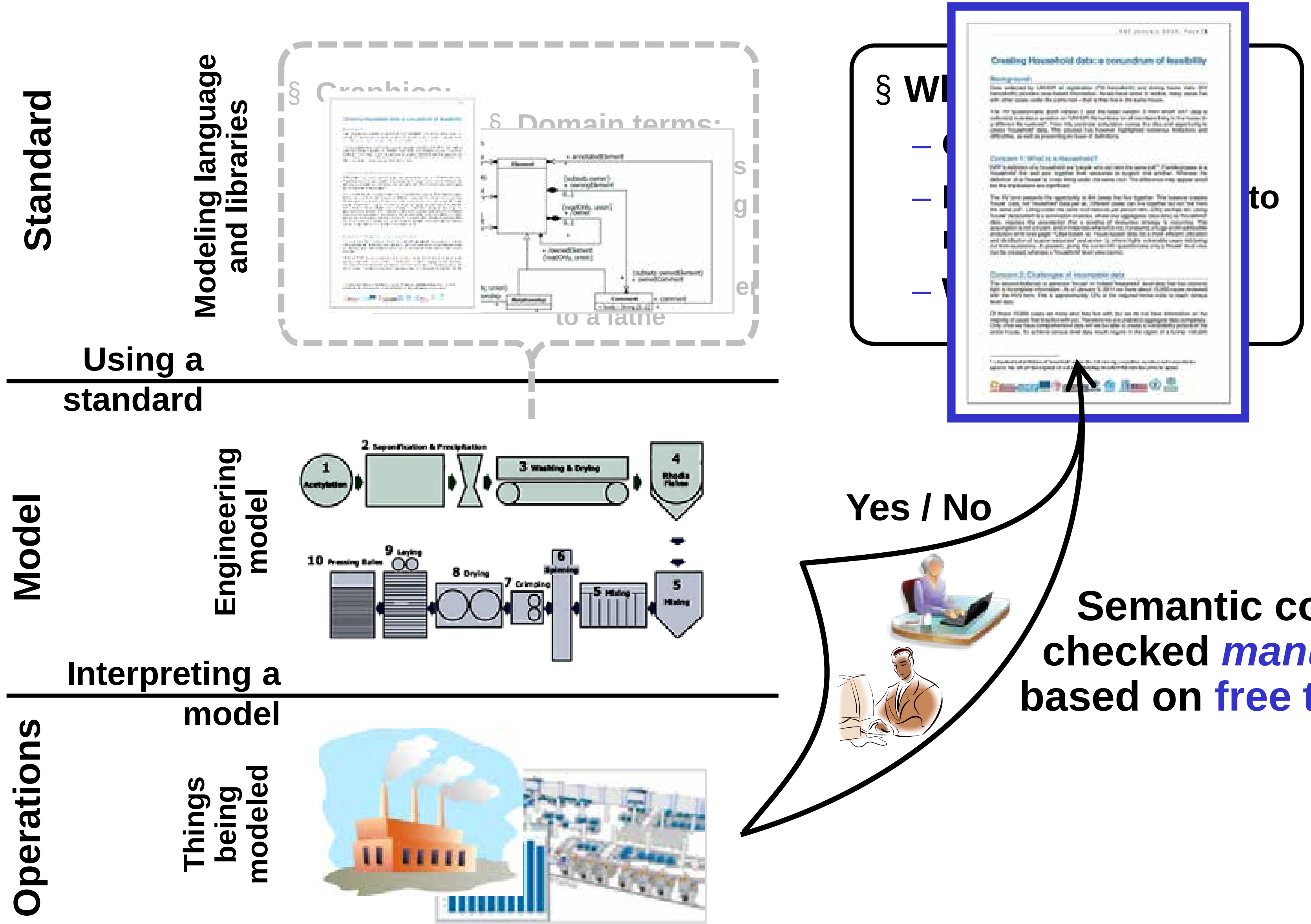
Operations

Things being modeled



Semantic conformance

Checking Semantic Conformance, Manual

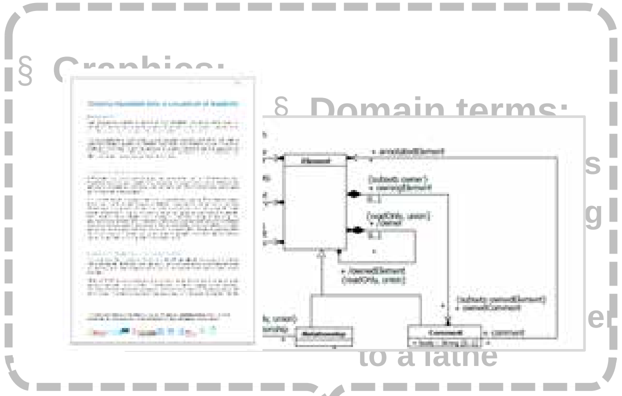


Semantic conformance checked *manually* based on free text

Checking Semantic Conformance, Autoish

Standard

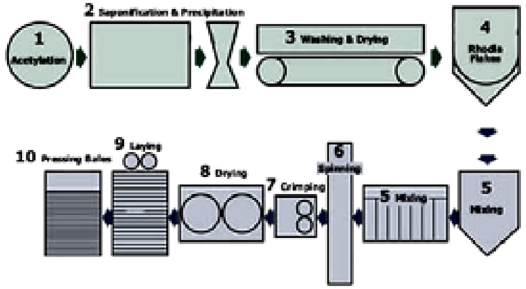
Modeling language and libraries



Using a standard

Model

Engineering model



Interpreting a model

Operations

Things being modeled

§ Wh

– C

– I to

I

– W

Creating Household data: a conundrum of feasibility

Background

[illegible]

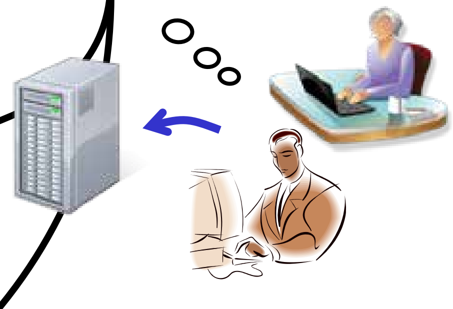
Concern 1: What is a Household?

[illegible]

Concern 2: Challenges of incomplete data

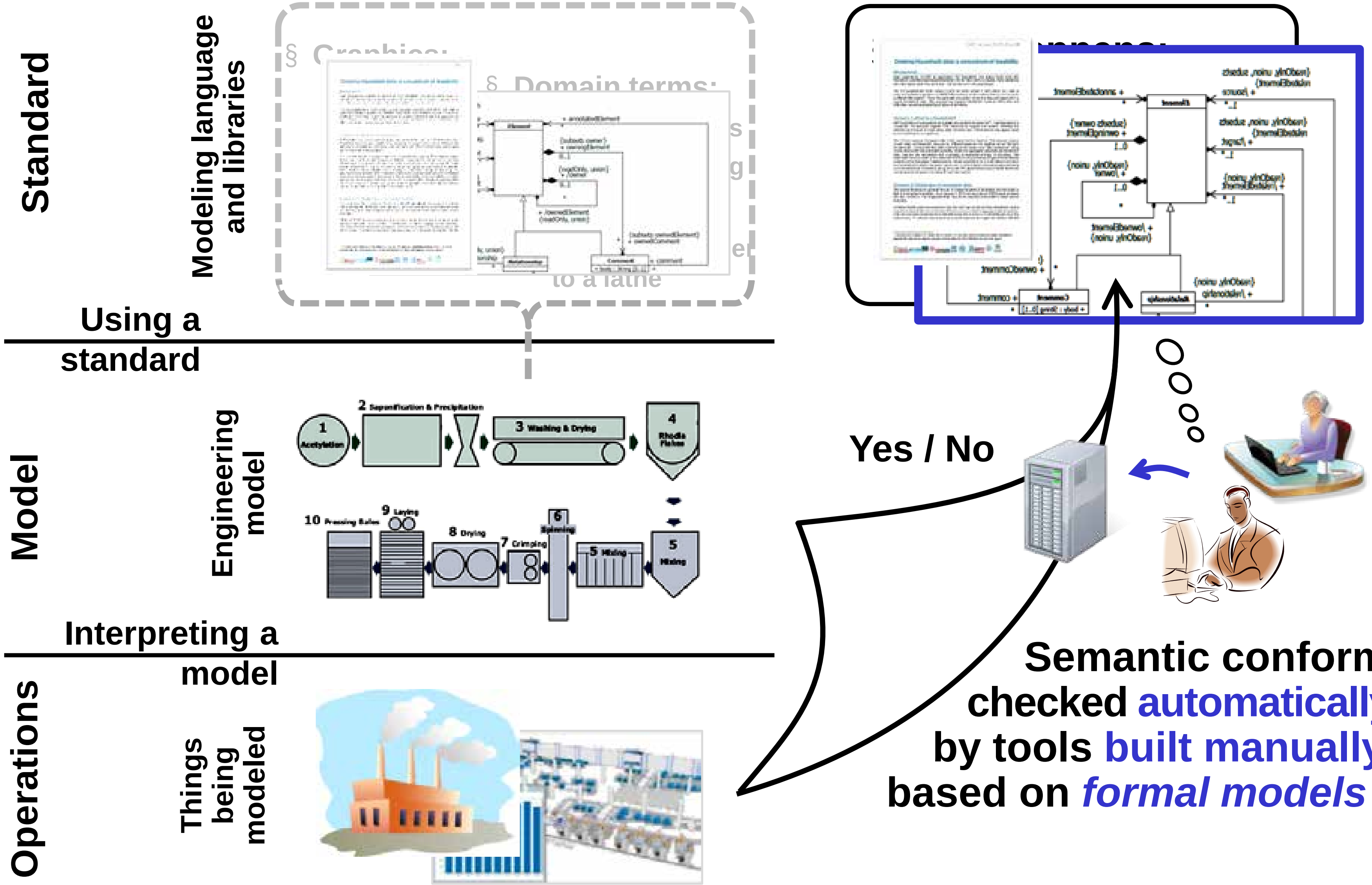
[illegible]

Yes / No



Semantic conformance checked *automatically* by tools built manually based on free text

Checking Semantic Conformance, More Auto



Standard

Modeling language and libraries

Using a standard

Model

Engineering model

Interpreting a model

Operations

Things being modeled

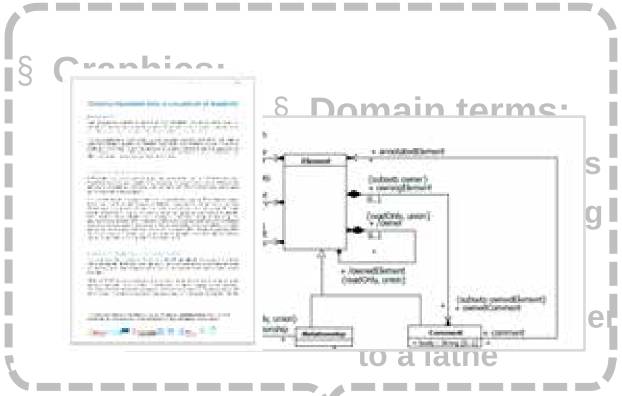
Yes / No

Semantic conformance checked automatically by tools built manually based on *formal models*

Checking Semantic Conformance, Most Auto

Standard

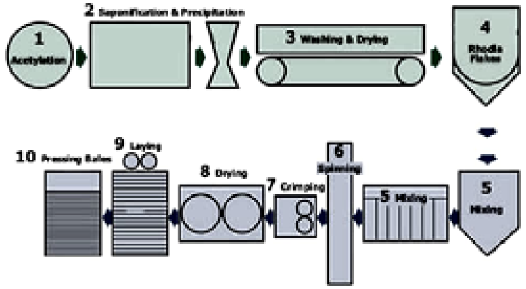
Modeling language and libraries



Using a standard

Model

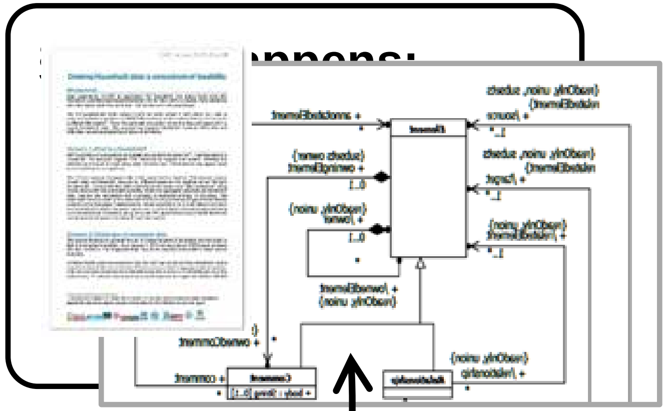
Engineering model

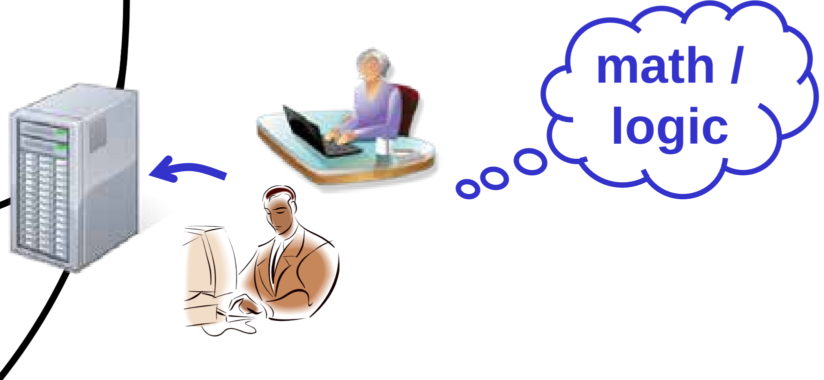


Interpreting a model

Operations

Things being modeled

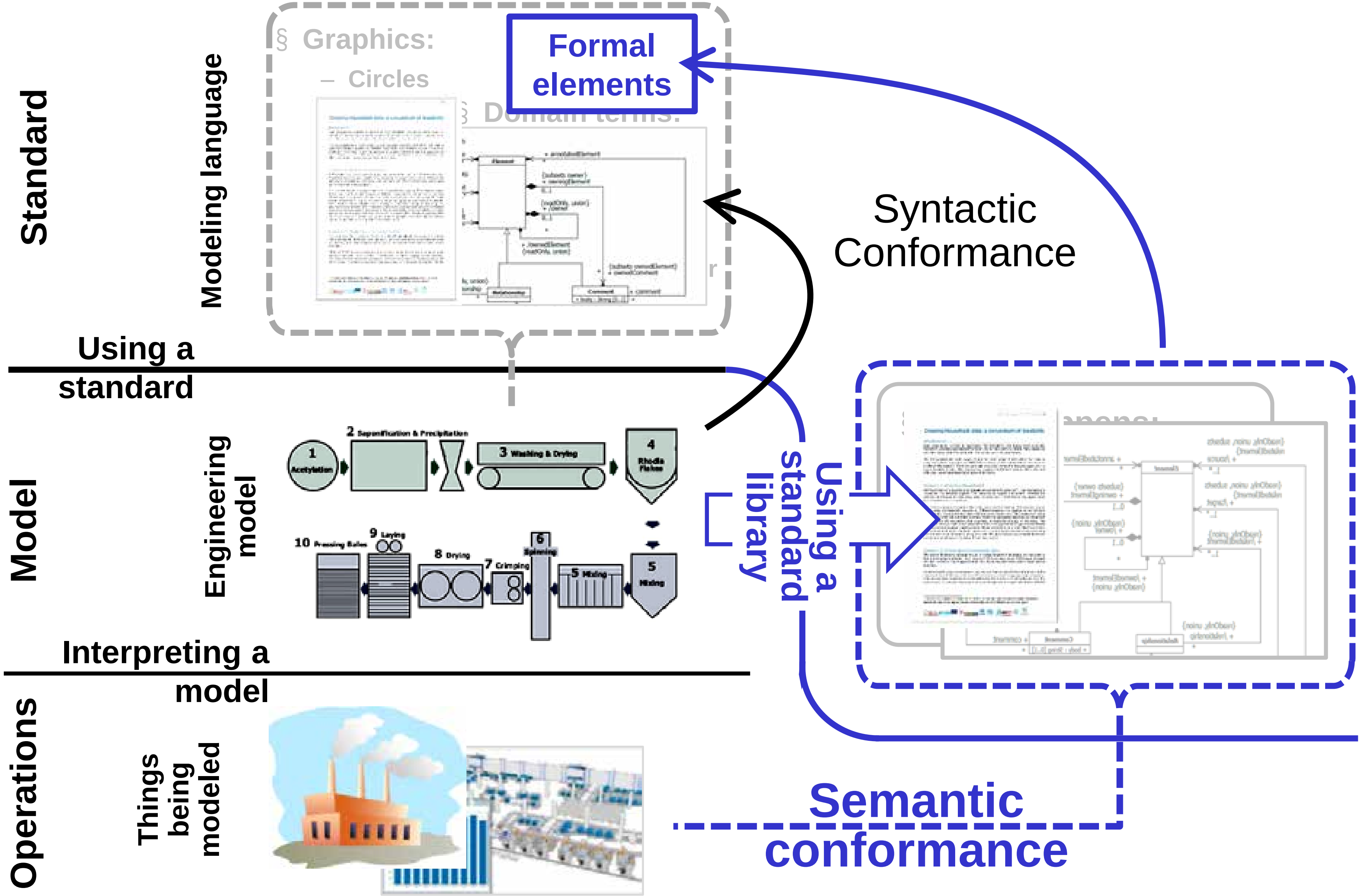




Yes / No

Semantic conformance checked automatically by tools built manually for checking *all formal models*

Standard Semantic Models



Standard

Modeling language

§ Graphics:

– Circles

§ Domain terms:

Formal elements

Syntactic Conformance

Using a standard

Model

Engineering model

Interpreting a model

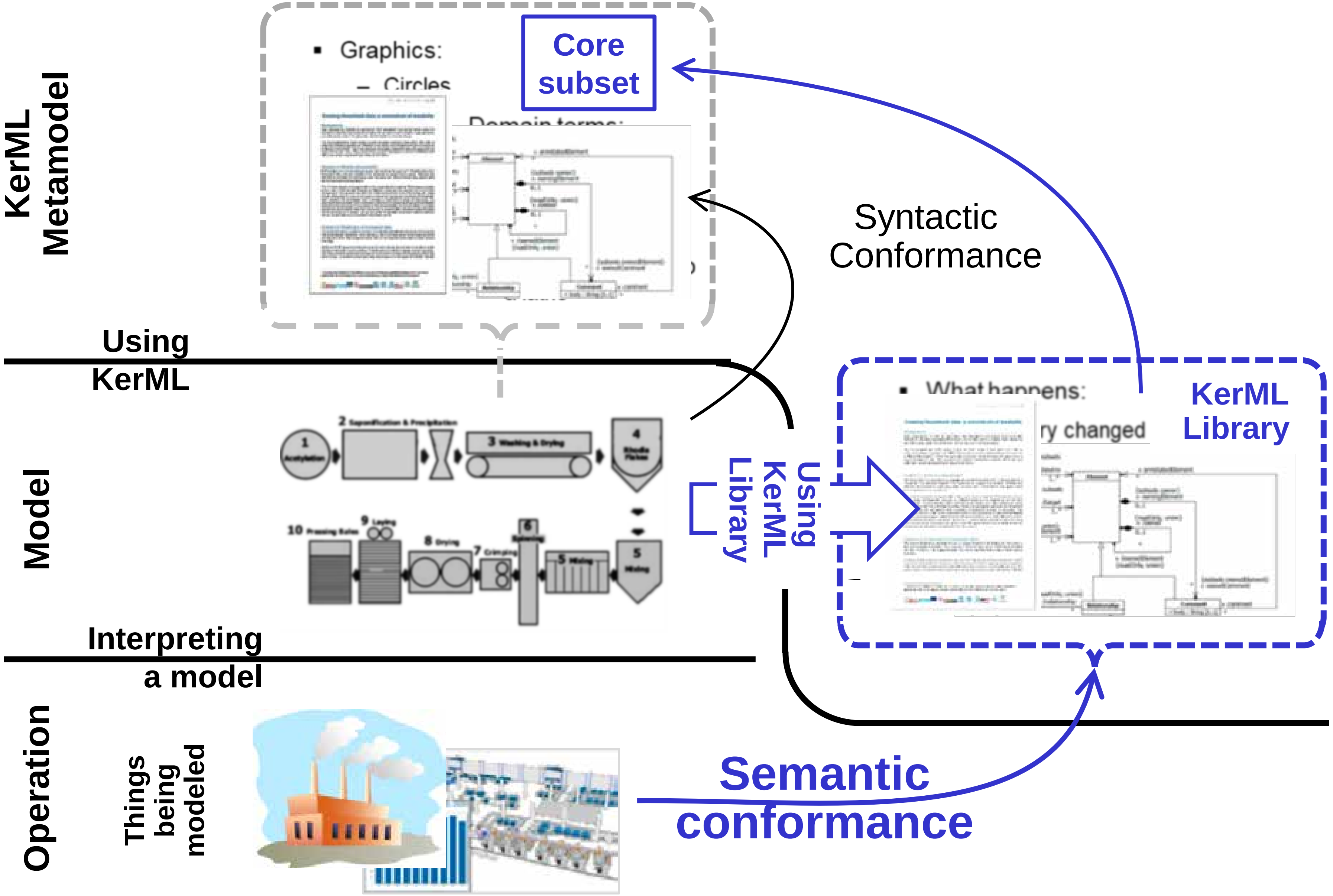
Operations

Things being modeled

Using a standard library

Semantic conformance

Semantic Conformance (SST)



KerML Metamodel

Core subset

Syntactic Conformance

Using KerML

Model

Using KerML Library

KerML Library

Interpreting a model

Operation

Things being modeled

Semantic conformance

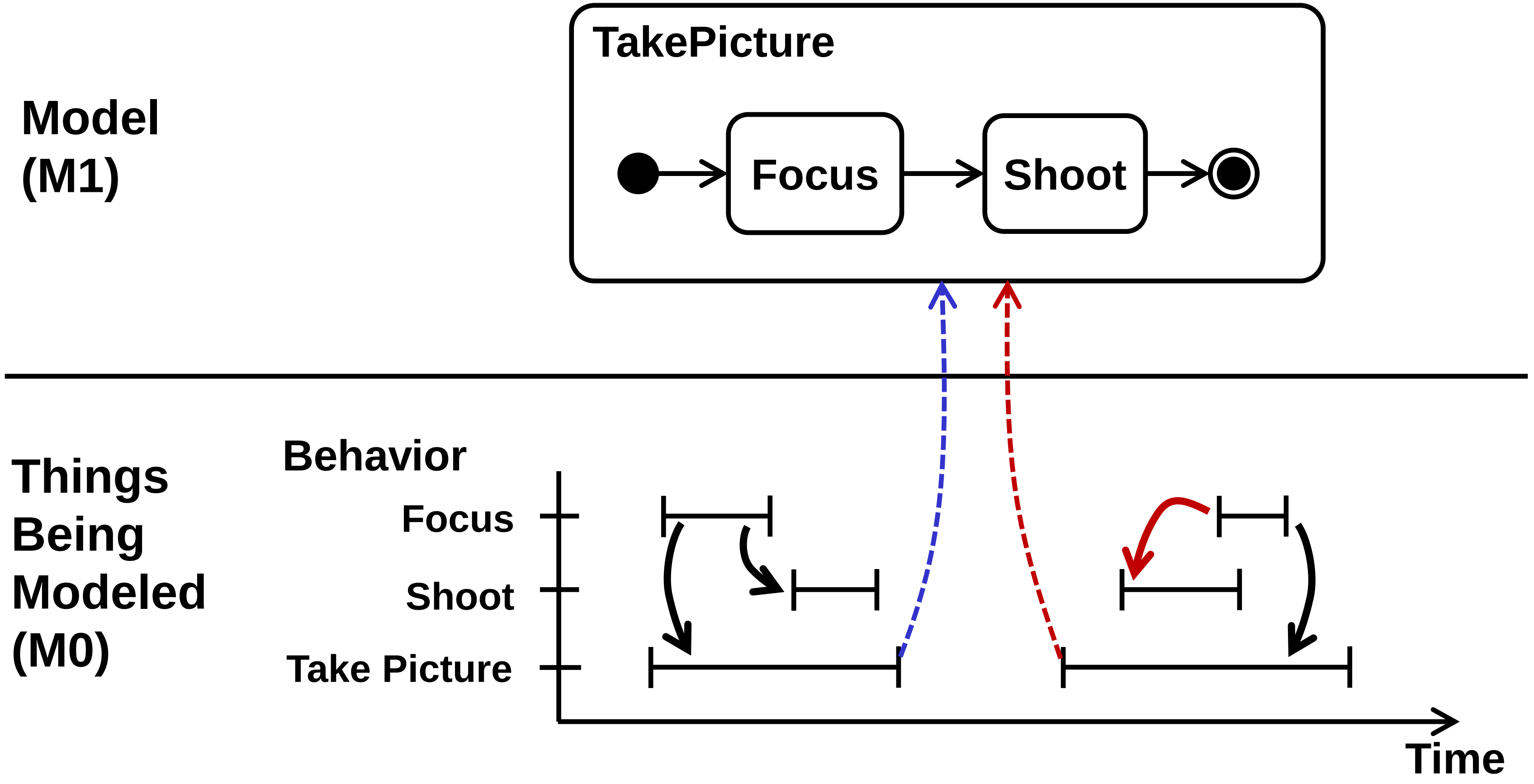
Overview

§ Motivation / Problem
  – Modeling Languages and Analysis
  – Interpreting Models (Semantics)

§ Solution
  – Standardizing Semantics
  – **Logical Classification**
  – Semantics, Without Math
  – SysML 2 Semantics

§ Summary

Conformance = Classification



TakePicture occurrences that do/not not conform to (are/not classified by) the behavior model.

Classification Synonyms

Classified by

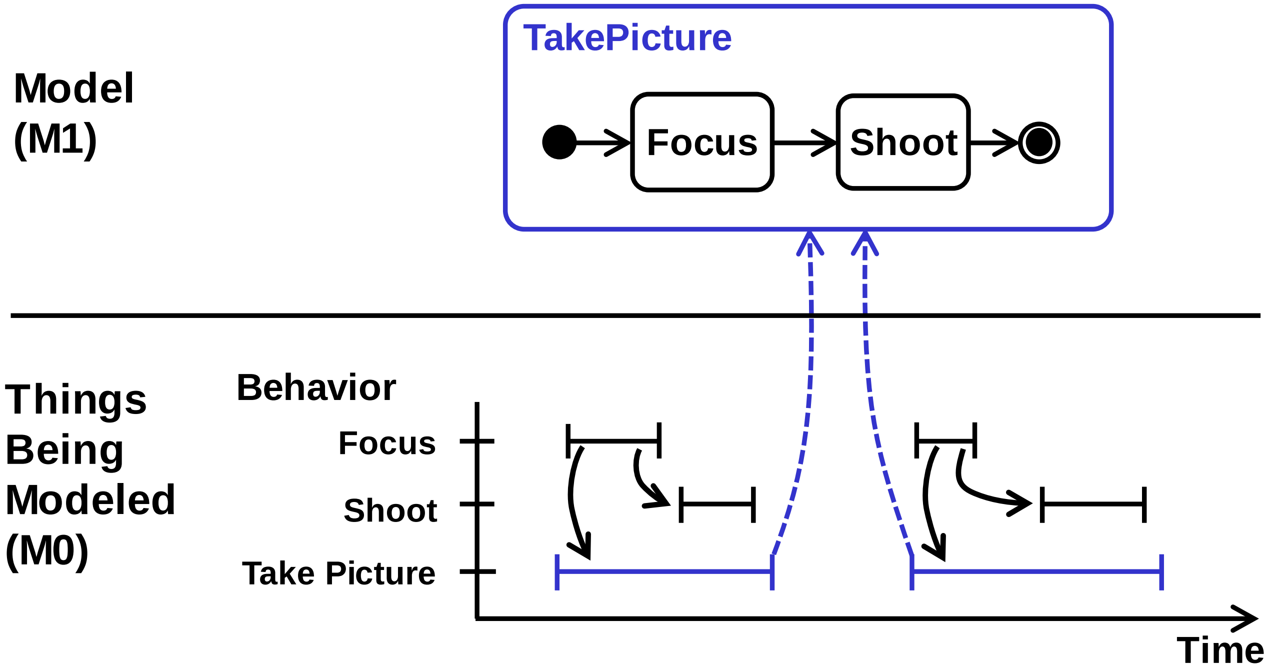
Modeled by

Specified by

Conforms to

Follows

Satisfies (logically)

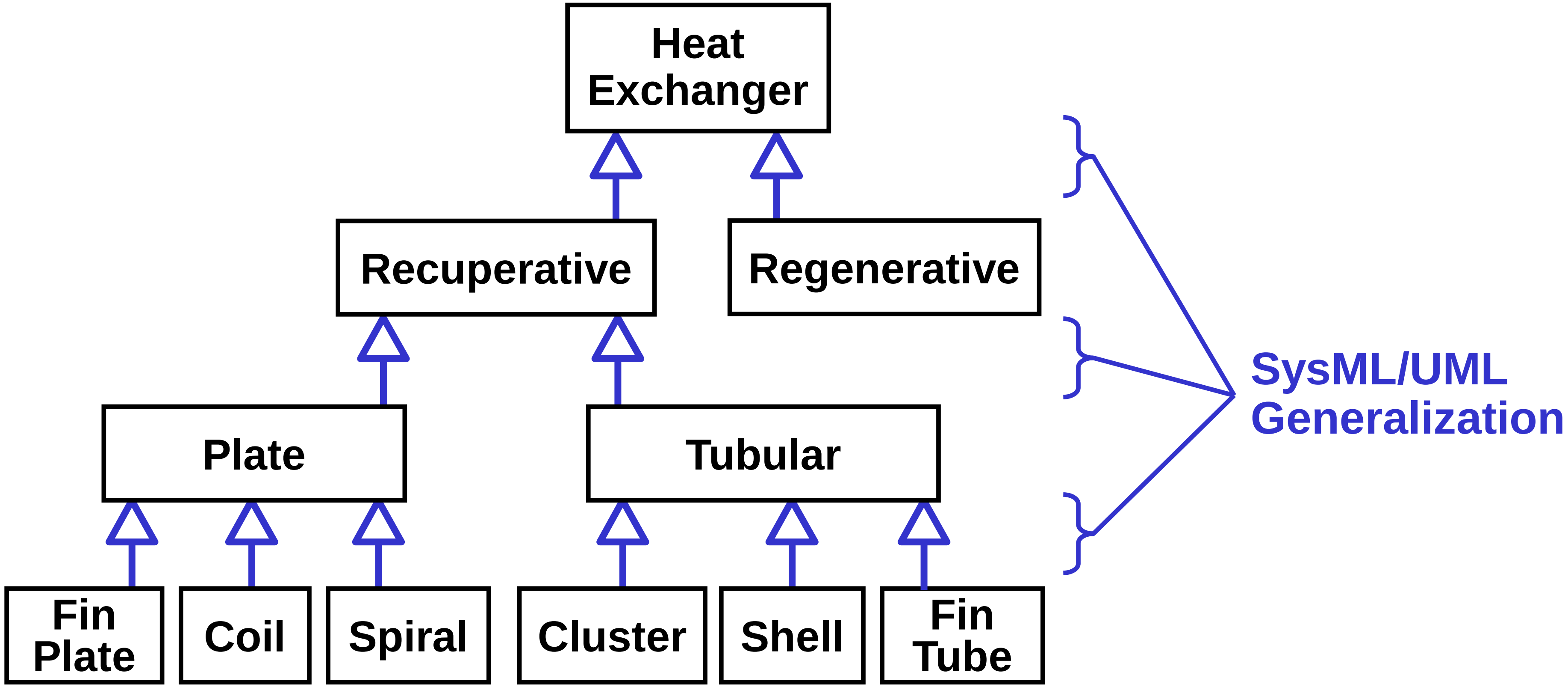


Not quite: "Instance of" (in the OO sense)

Not *at all* : "Execution of" (MES/software sense)

Taxonomies

Model
(M1)



- “Sub”classification …
- …of real or simulated things.

http://mechanicalgalaxy.blogspot.com/2011/05/heat-exchangers-classification-types.html

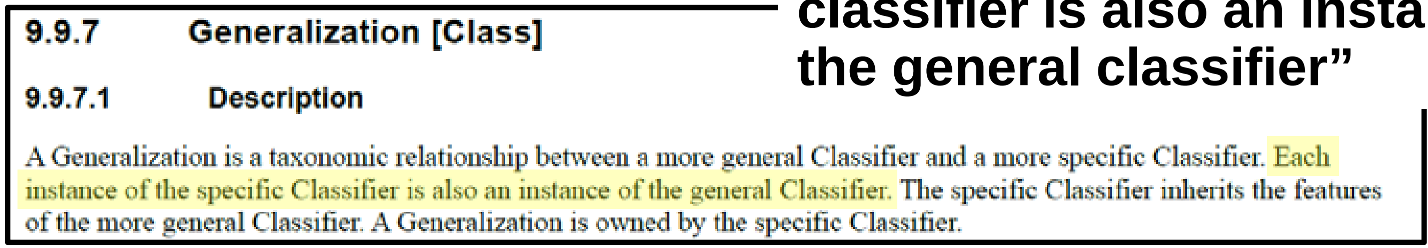
Overview

- Motivation / Problem
  - Modeling Languages and Analysis
  - Interpreting Models (Semantics)
- Solution
  - Standardizing Semantics
  - Logical Classification
  - **Semantics, Without Math**
  - SysML 2 Semantics
- Summary

Informal Semantics

UML Generalization

From UML 2.5 Specification:

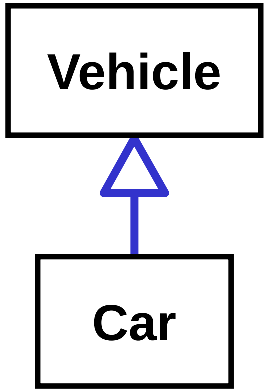


9.9.7 Generalization [Class]

9.9.7.1 Description

A Generalization is a taxonomic relationship between a more general Classifier and a more specific Classifier. Each instance of the specific Classifier is also an instance of the general Classifier. The specific Classifier inherits the features of the more general Classifier. A Generalization is owned by the specific Classifier.

“Each instance of the specific classifier is also an instance of the general classifier”



“Every instance of Car is an instance of Vehicle”

iow

“Every Car is a Vehicle”

iow

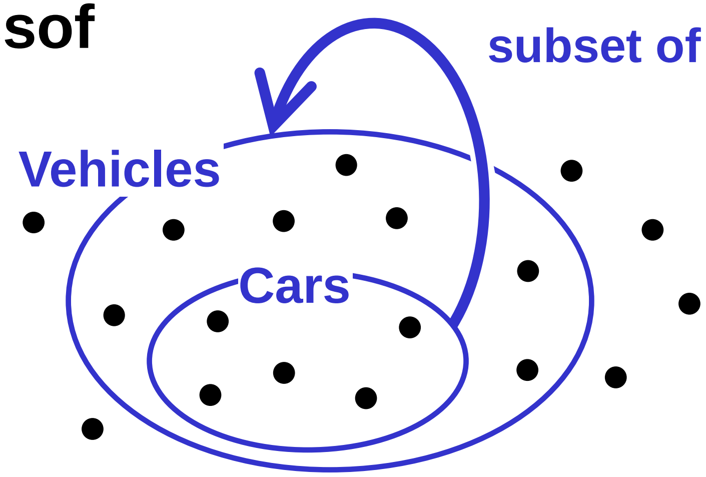
“Cars are vehicles”

How can this be specified more precisely?

Mathematical Semantics

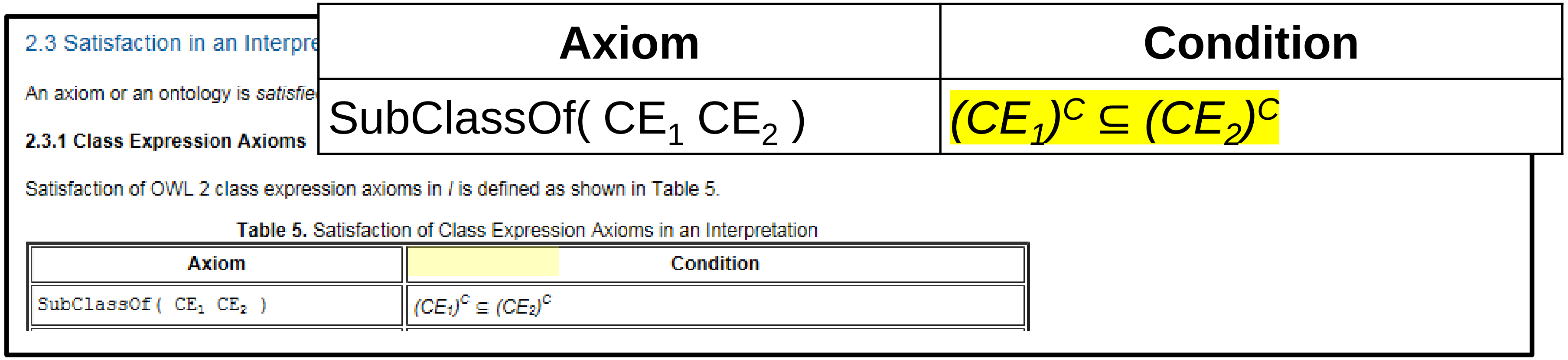
OWL SubClassof

SubClassOf (Car, Vehicle)



● = a single real or virtual thing

From OWL 2 Direct Semantics:



Axiom	Condition
SubClassOf(CE_1 CE_2)	$(CE_1)^C \subseteq (CE_2)^C$

2.3 Satisfaction in an Interpre

An axiom or an ontology is *satisfie*

2.3.1 Class Expression Axioms

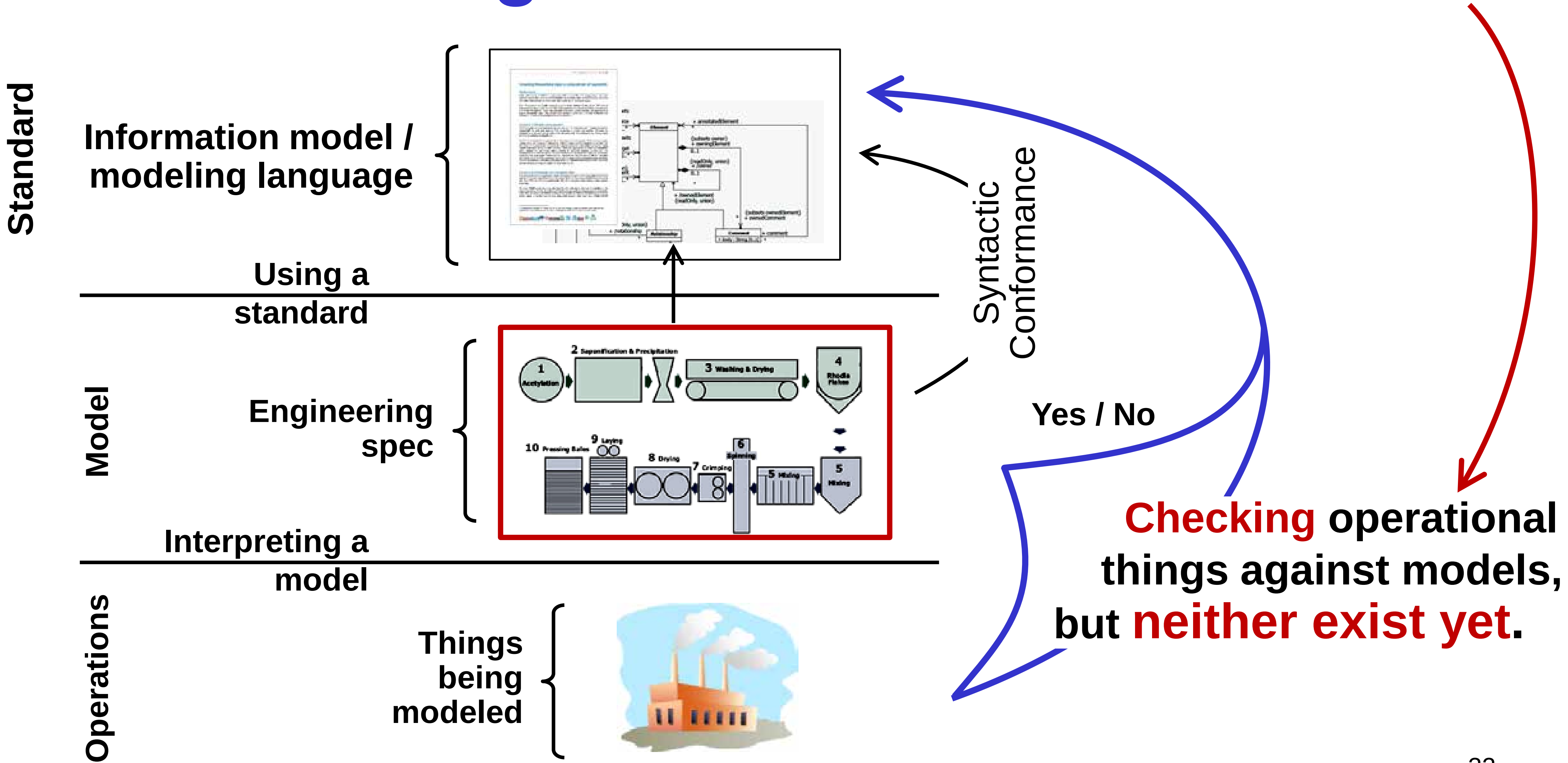
Satisfaction of OWL 2 class expression axioms in *I* is defined as shown in Table 5.

Table 5. Satisfaction of Class Expression Axioms in an Interpretation

Axiom	Condition
`SubClassOf( CE₁ CE₂ )`	$(CE_1)^C \subseteq (CE_2)^C$

https://www.w3.org/TR/owl2-direct-semantics/

Standardizing Semantic Conformance?



Standard

Information model / modeling language

Using a standard

Model

Engineering spec

Interpreting a model

Operations

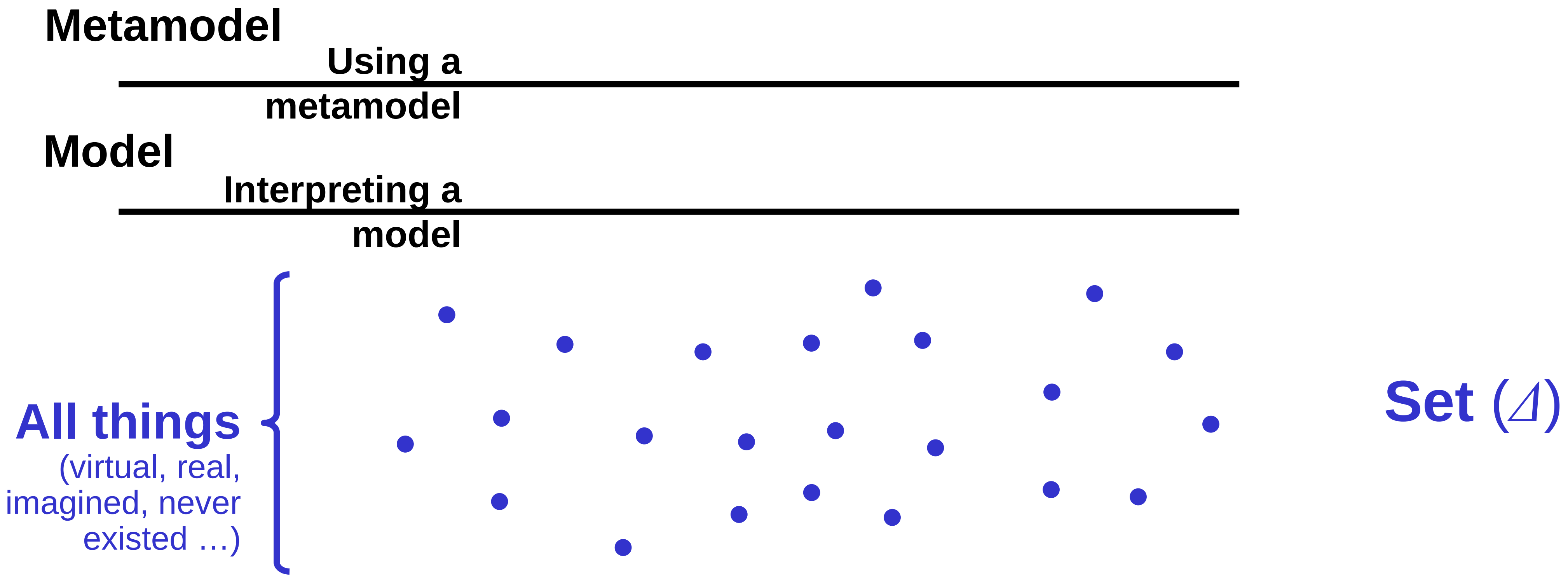
Things being modeled

Syntactic Conformance

Yes / No

Checking operational things against models, but neither exist yet.

Universe



Metamodel

Using a metamodel

Model

Interpreting a model

All things

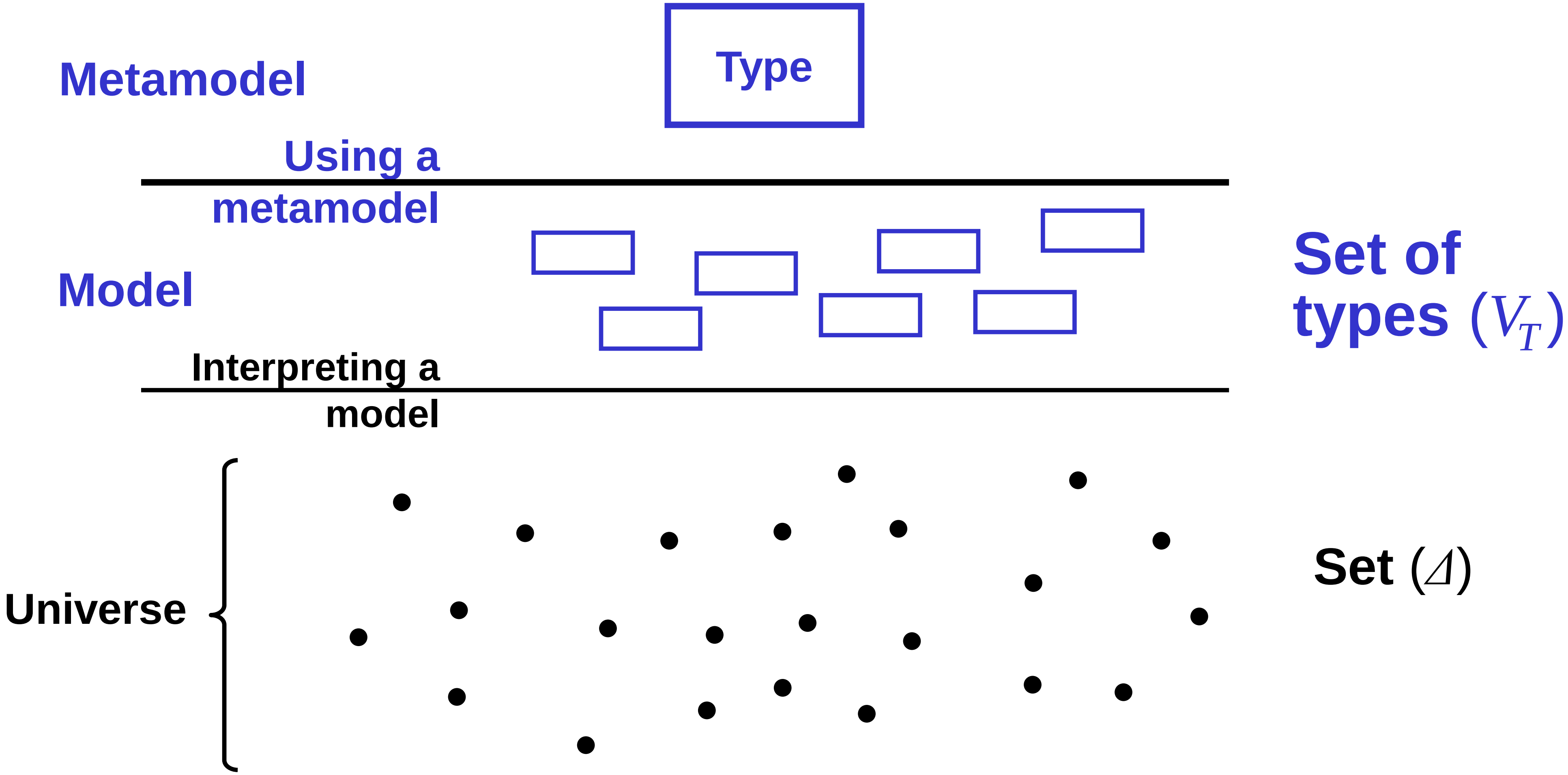
(virtual, real, imagined, never existed …)

Set (Δ)

- **Everything, anything, no restrictions, don’t know anything about them, how many, etc.**
- **For interpreting models.**

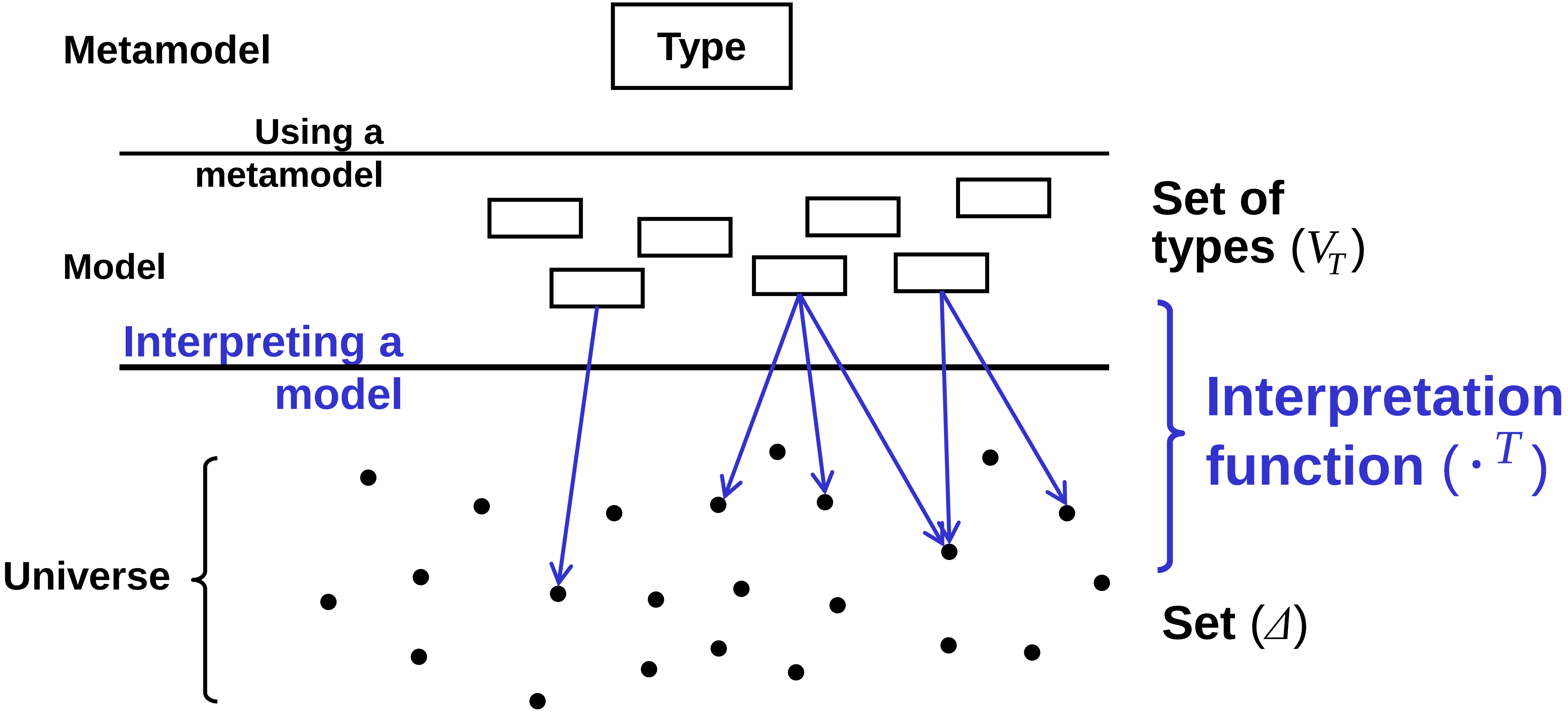
OWL 2 Direct Semantics https://www.w3.org/TR/owl2-direct-semantics/

Model Elements



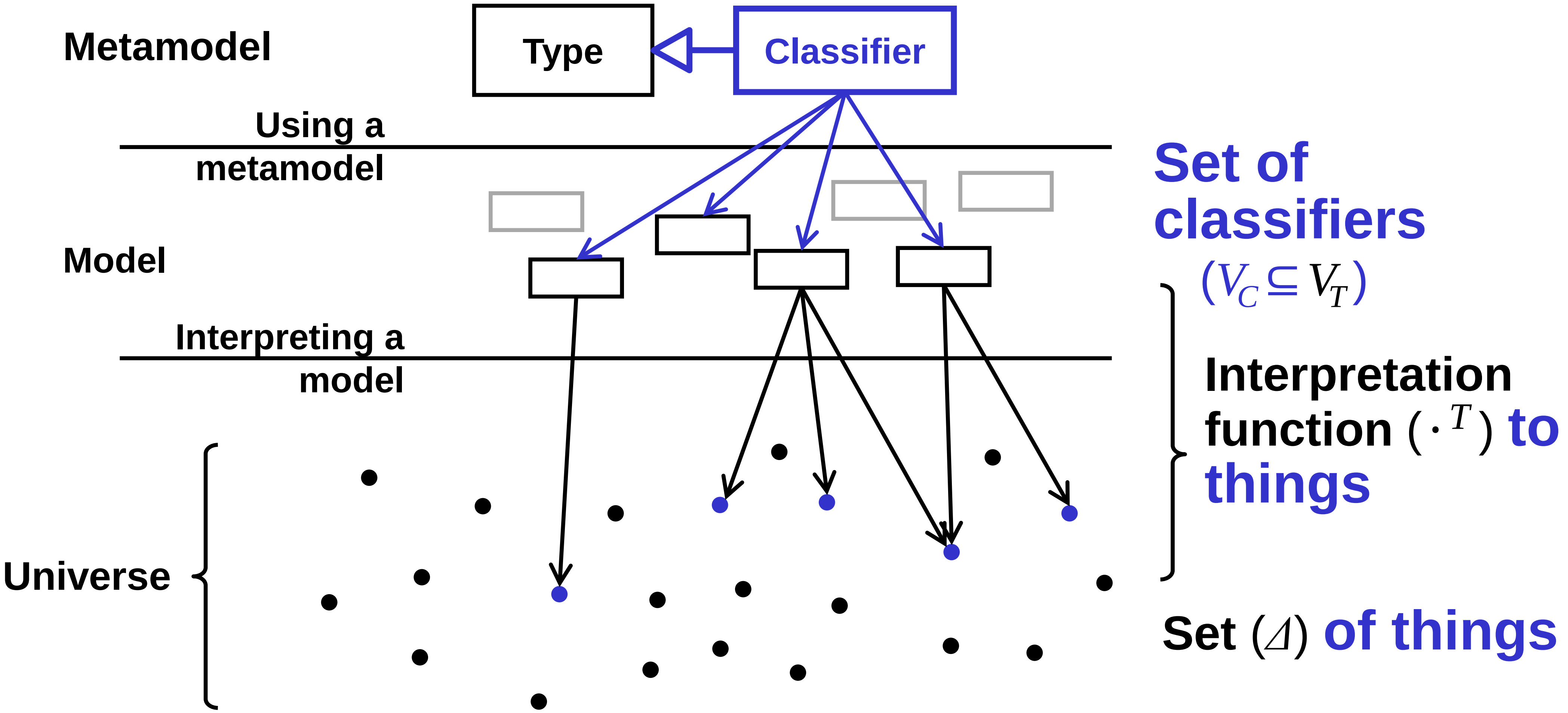
- Beginning of syntax.

Interpretation = Classification



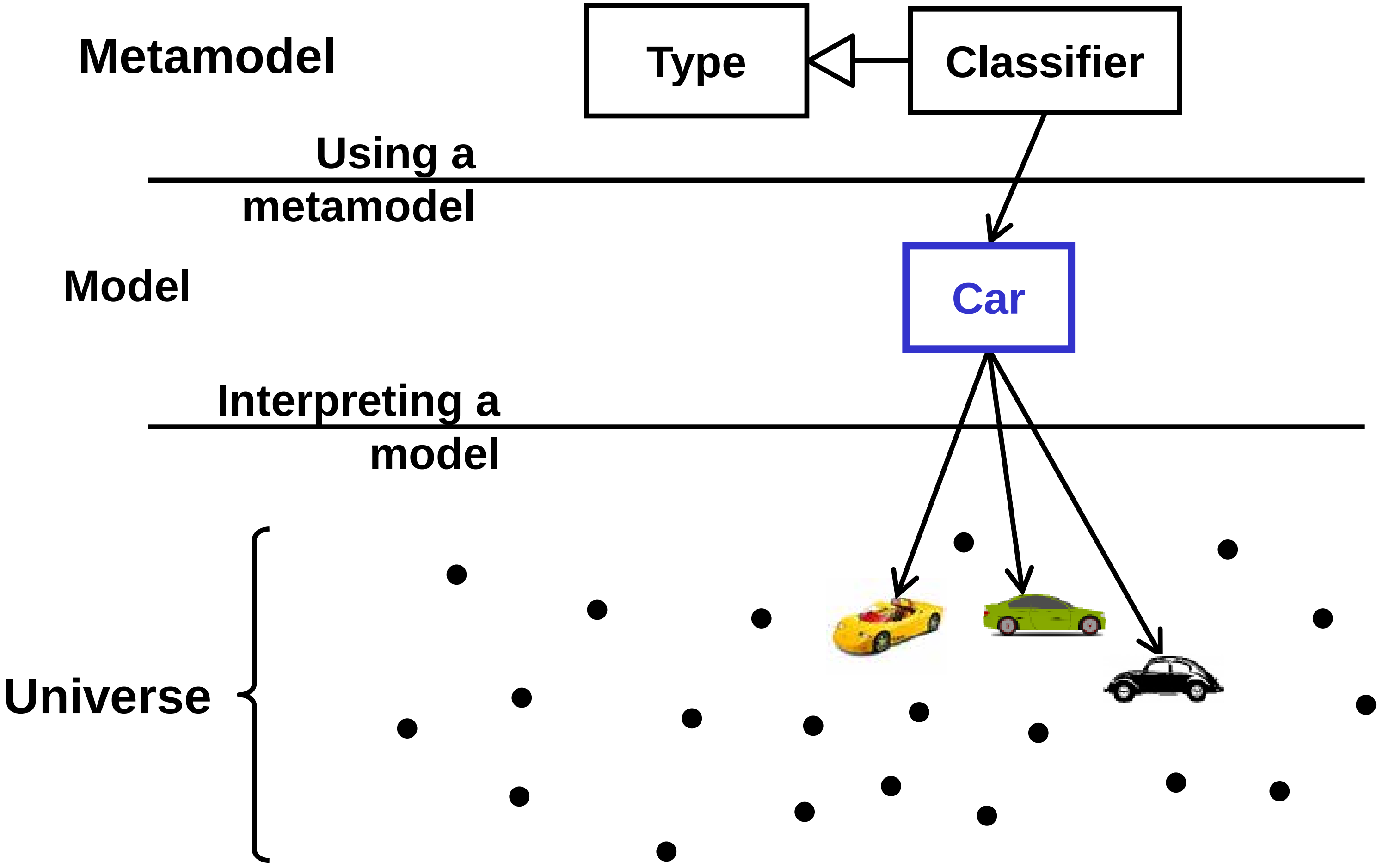
§ **Links model elements to things in the universe.**

Interpretation , Classifiers



- Classifiers are interpreted as (sets of) things in the universe.

Interpretation , Classifiers, Example



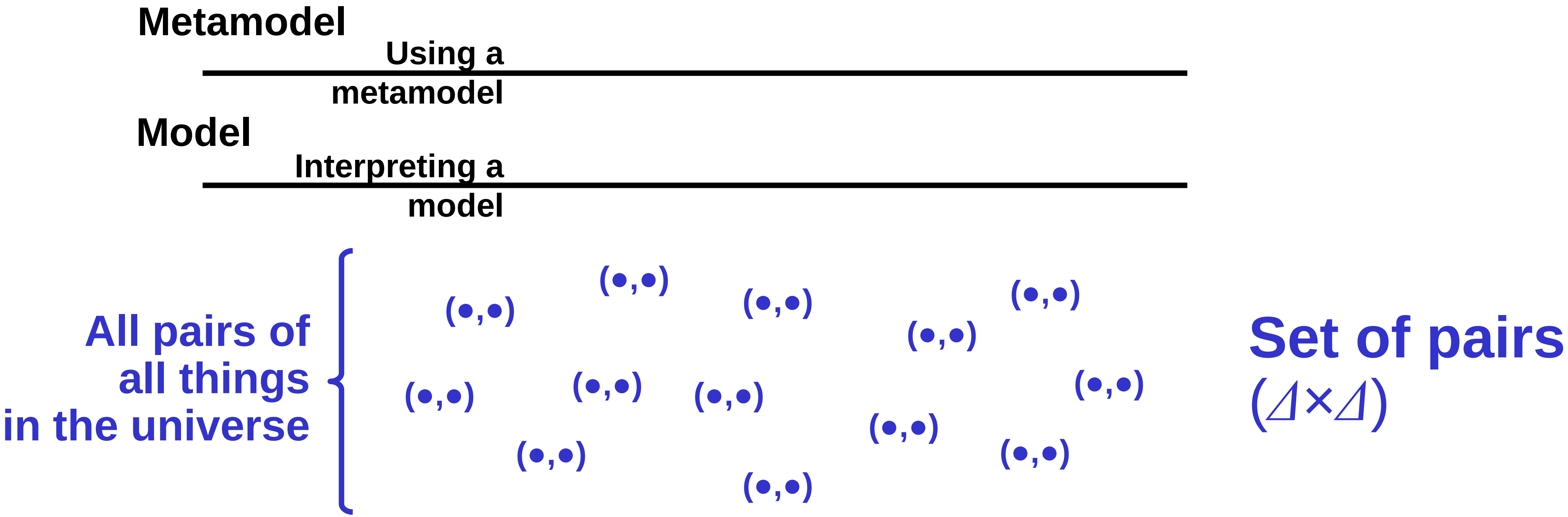
A classifier
(Car $\in V_C$)

Interpretation function $(\cdot^{\mathcal{I}})$ to real or virtual cars

Set of car things
((Car)$^{\mathcal{I}} \subseteq \Delta$)

- **Car is interpreted as some real or virtual things.**

Pairs of Things in the Universe



Metamodel

Using a metamodel

Model

Interpreting a model

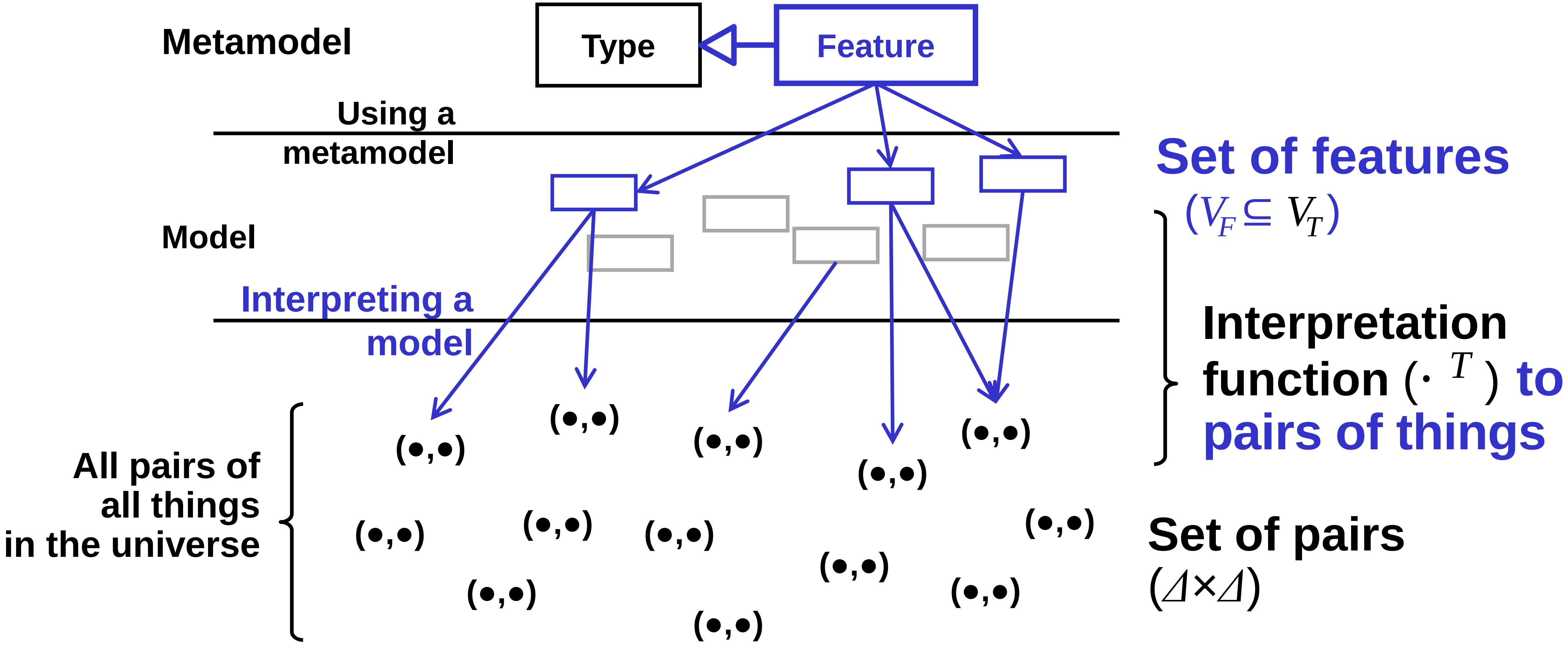
All pairs of all things in the universe

(•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•) (•,•)

Set of pairs $(\Delta\times\Delta)$

- **Every pair of anything, no restrictions on pairing, don’t know anything about the pairings, etc.**
- **For interpreting relationships between things.**

Interpretation, Features



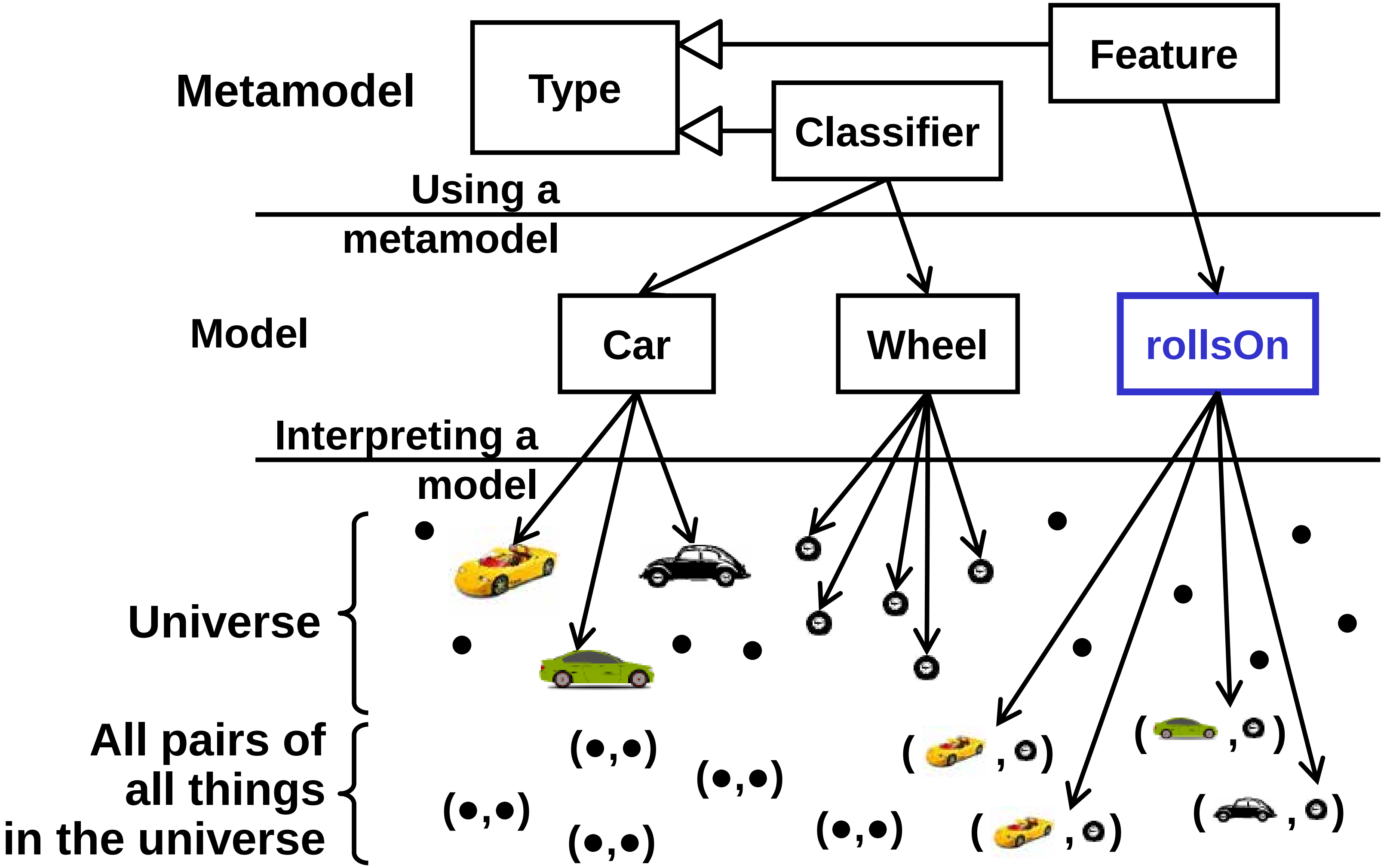
Set of features $(V_F \subseteq V_T)$

Interpretation function $(\cdot^{\mathcal{T}})$ to pairs of things

Set of pairs $(\Delta \times \Delta)$

§ **Features are interpreted as (sets of) pairs of things in the universe.**

Interpretation, Features, Example



Metamodel

Type

Feature

Classifier

Using a metamodel

Model

Car

Wheel

rollsOn

Interpreting a model

Universe

All pairs of all things in the universe

(•,•) (•,•) (•,•) (•,•) (•,•) (•,•)

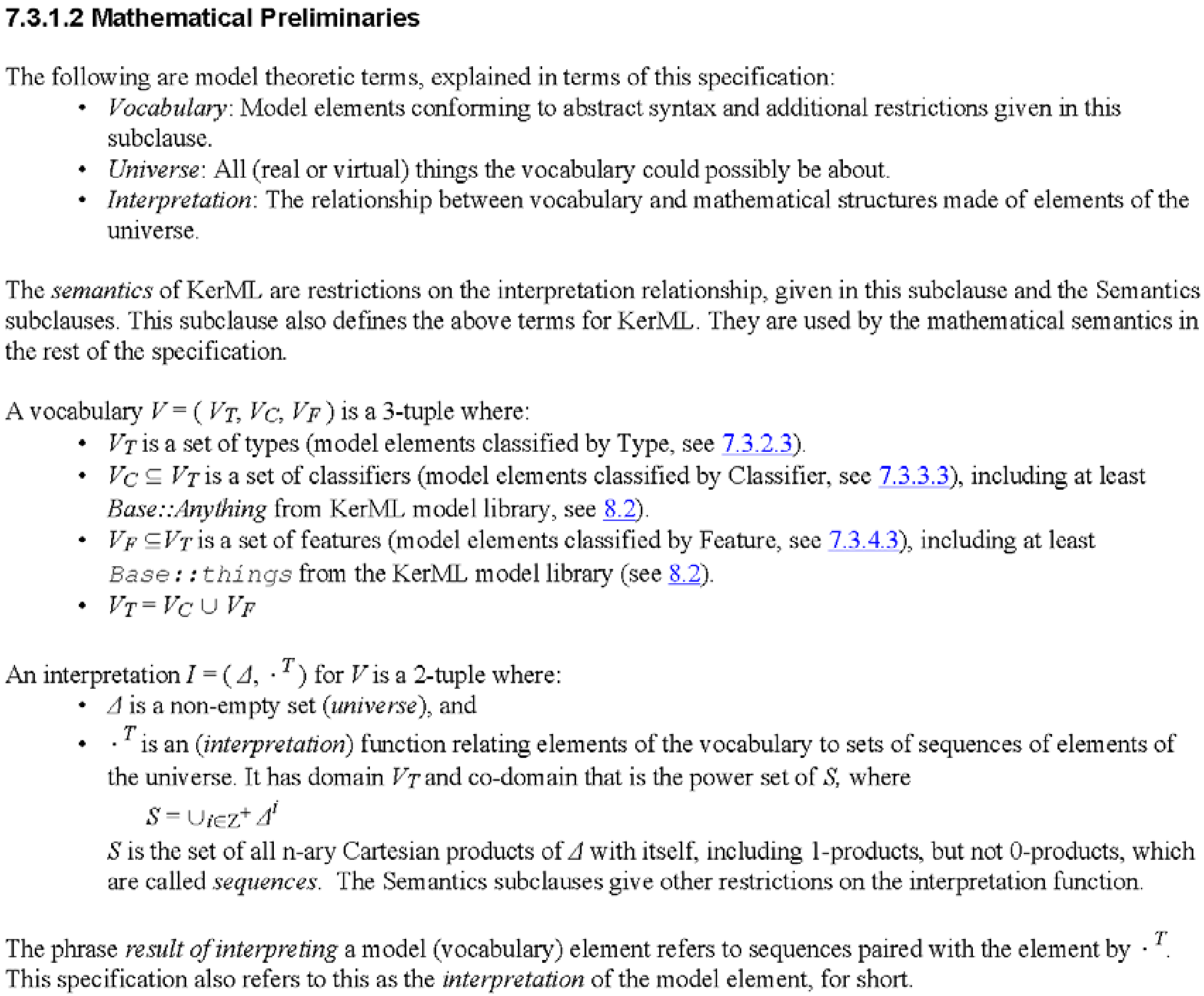
A feature
(rollsOn $\in V_F$)

Interpretation function $(\cdot^T)$ **to pairs of real or virtual cars and wheels**

Set of rollsOn pairs
((rollsOn)$^T \subseteq (\Delta\times\Delta)$)

- § **rollsOn is interpreted as (sets of) pairs of real or virtual cars and wheels.**

SysML 2 Math Terms



7.3.1.2 Mathematical Preliminaries

The following are model theoretic terms, explained in terms of this specification:

- *Vocabulary*: Model elements conforming to abstract syntax and additional restrictions given in this subclause.
- *Universe*: All (real or virtual) things the vocabulary could possibly be about.
- *Interpretation*: The relationship between vocabulary and mathematical structures made of elements of the universe.

The *semantics* of KerML are restrictions on the interpretation relationship, given in this subclause and the Semantics subclauses. This subclause also defines the above terms for KerML. They are used by the mathematical semantics in the rest of the specification.

A vocabulary $V = (V_T, V_C, V_F)$ is a 3-tuple where:

- V_T is a set of types (model elements classified by Type, see 7.3.2.3).
- $V_C \subseteq V_T$ is a set of classifiers (model elements classified by Classifier, see 7.3.3.3), including at least *Base::Anything* from KerML model library, see 8.2).
- $V_F \subseteq V_T$ is a set of features (model elements classified by Feature, see 7.3.4.3), including at least `Base::things` from the KerML model library (see 8.2).
- $V_T = V_C \cup V_F$

An interpretation $I = (\Delta, \cdot^T)$ for V is a 2-tuple where:

- Δ is a non-empty set (*universe*), and
- $\cdot^T$ is an (*interpretation*) function relating elements of the vocabulary to sets of sequences of elements of the universe. It has domain V_T and co-domain that is the power set of S, where

  $S = \cup_{i \in Z^+} \Delta^i$

  S is the set of all n-ary Cartesian products of Δ with itself, including 1-products, but not 0-products, which are called *sequences*. The Semantics subclauses give other restrictions on the interpretation function.

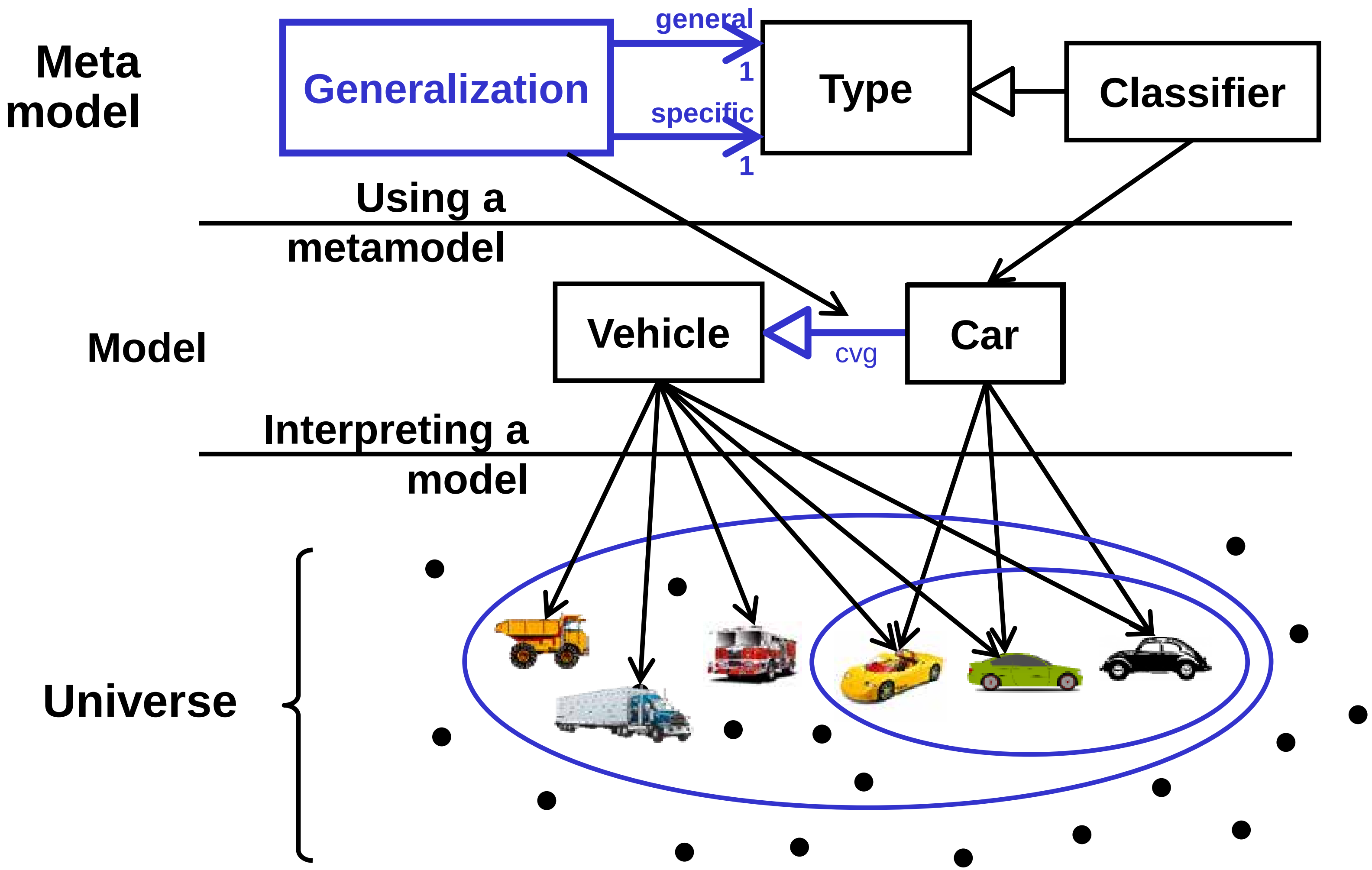
The phrase *result of interpreting* a model (vocabulary) element refers to sequences paired with the element by $\cdot^T$. This specification also refers to this as the *interpretation* of the model element, for short.

Mini-Glossary

Vocabulary

Interpretation

Interpretation, Generalization, Classifier



A generalization

(cvg.general = Vehicle ∧ cvg.specific = Car)

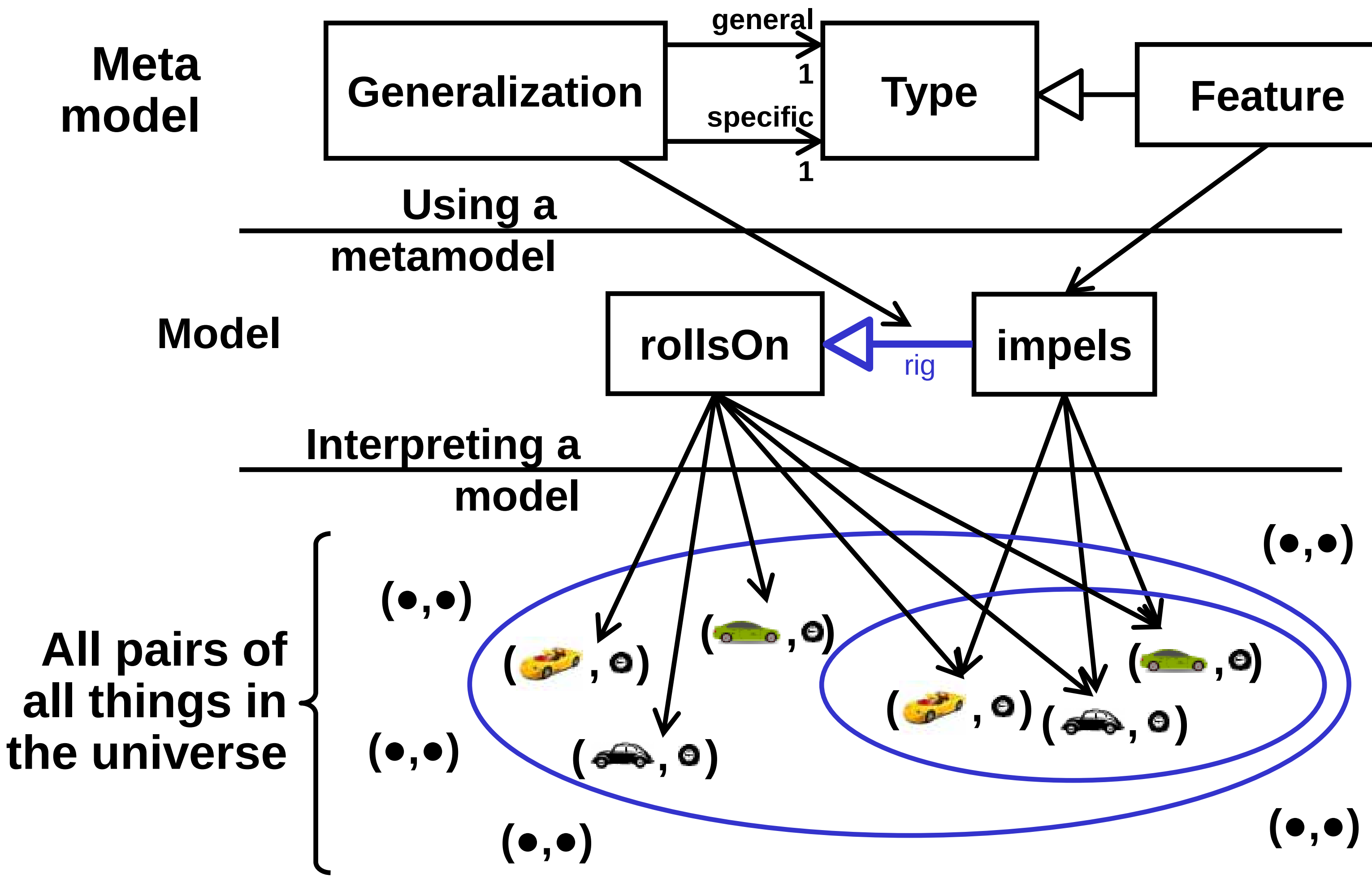
Interpretation function $(\cdot^{T})$ **to real or virtual vehicles, including cars**

Set of car things is a subset of the set of vehicle things

($(\mathrm{Car})^{T} \subseteq (\mathrm{Vehicle})^{T}$)

- **Car's interpretation is a subset of Vehicle's.**

Interpretation, Generalization, Feature



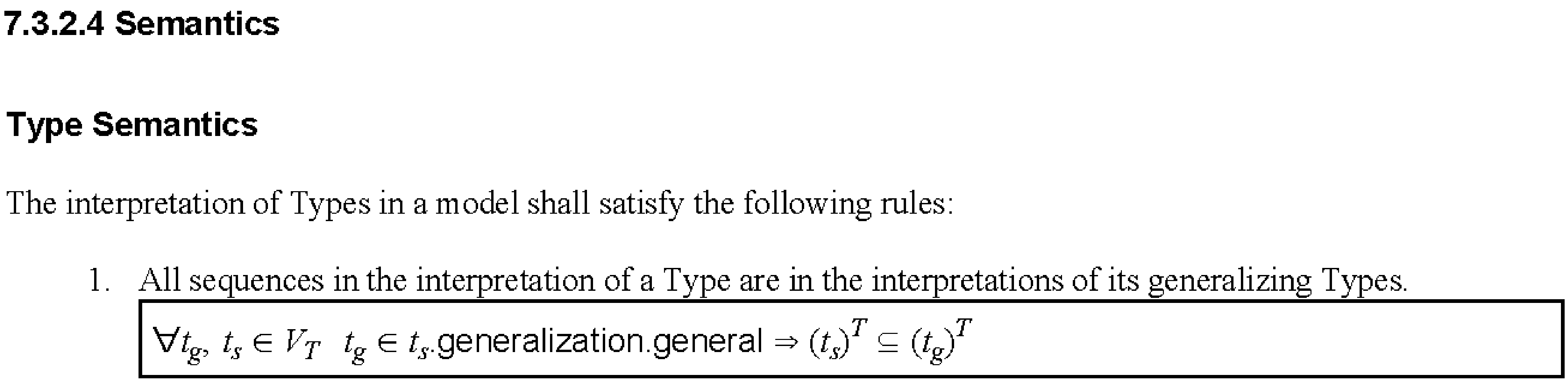
A generalization
(rig.general = rollsOn ∧ rig.specific = impels)

Interpretation function $(\cdot^{T})$ **to pairs of real or virtual cars and wheels**

Set of impels pairs is a subset of the set of rollsOn pairs
($(\text{impels})^{T} \subseteq (\text{rollsOn})^{T}$)

§ **impel’s interpretation is a subset of rollsOn’s.**

SysML 2 Generalization Math



7.3.2.4 Semantics

Type Semantics

The interpretation of Types in a model shall satisfy the following rules:

1. All sequences in the interpretation of a Type are in the interpretations of its generalizing Types.

$$\boxed{\forall t_g,\ t_s \in V_T \quad t_g \in t_s.\mathsf{generalization.general} \Rightarrow (t_s)^T \subseteq (t_g)^T}$$

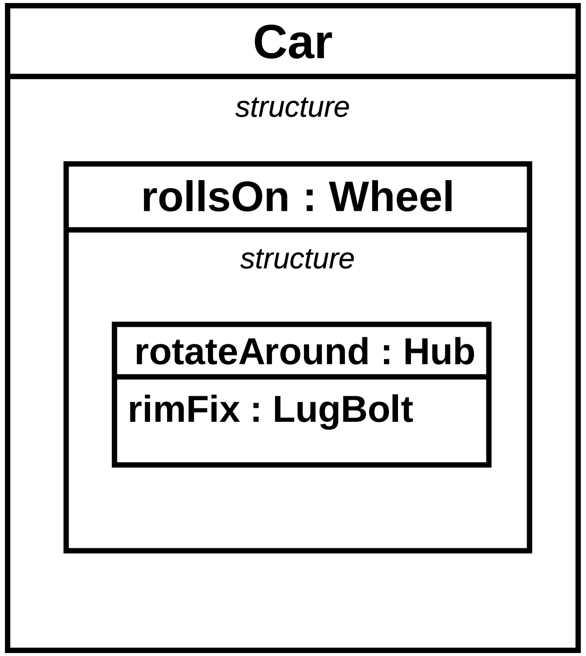
- **Generalization = subsetting of interpretations.**

Overview

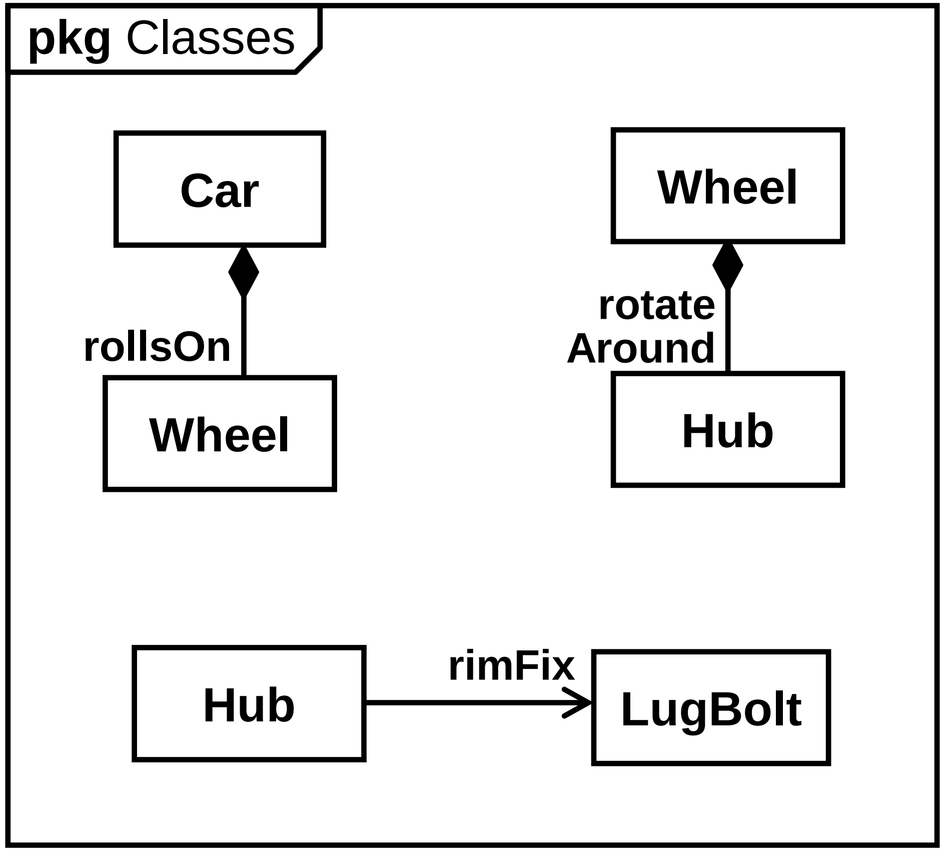
- Motivation / Problem
  - Modeling Languages and Analysis
  - Interpreting Models (Semantics)
- Solution
  - Standardizing Semantics
  - Logical Classification
  - Semantics, Without Math
  - **SysML 2 Semantics**
- Summary

Visual Nesting ≠ Class/Property Modeling

Model



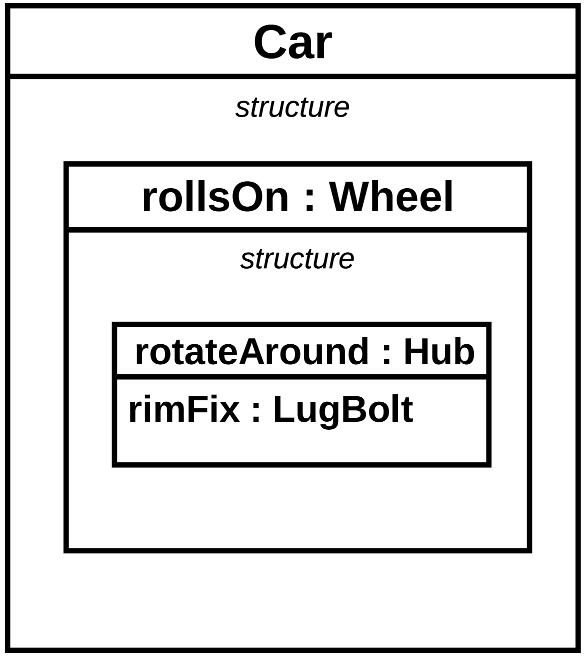
=



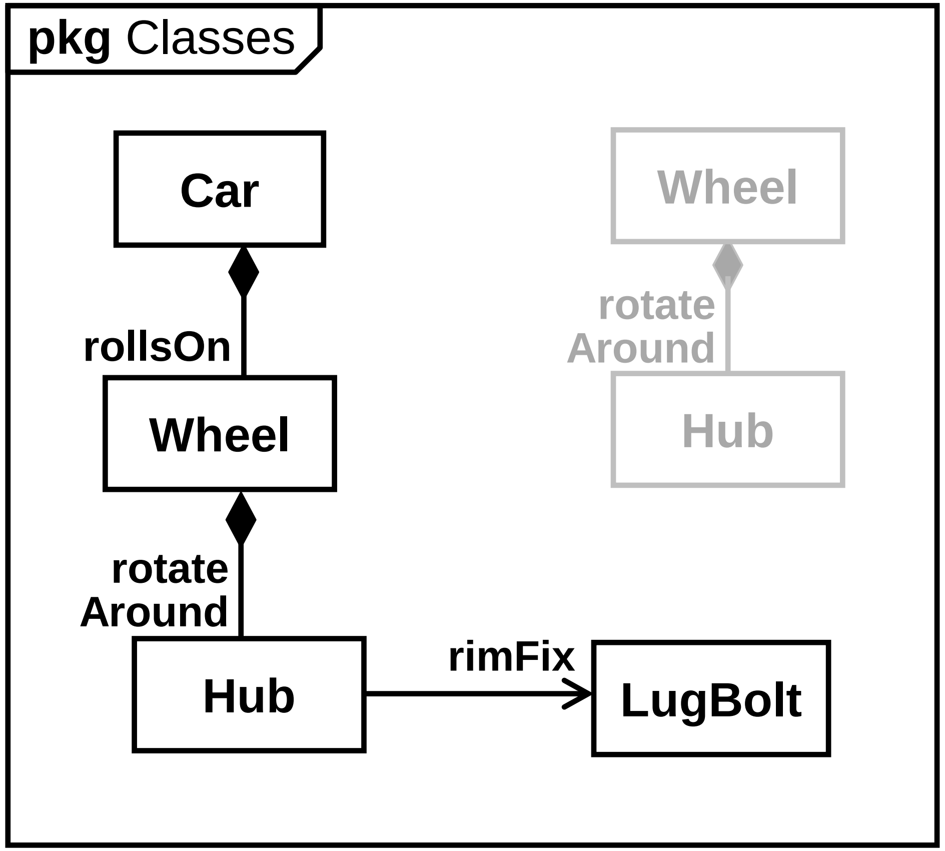
- **Structure diagrams same as class diagramns**
  - **as far as visual nesting goes.**

Visual Nesting ≠ Class/Property Modeling

Model

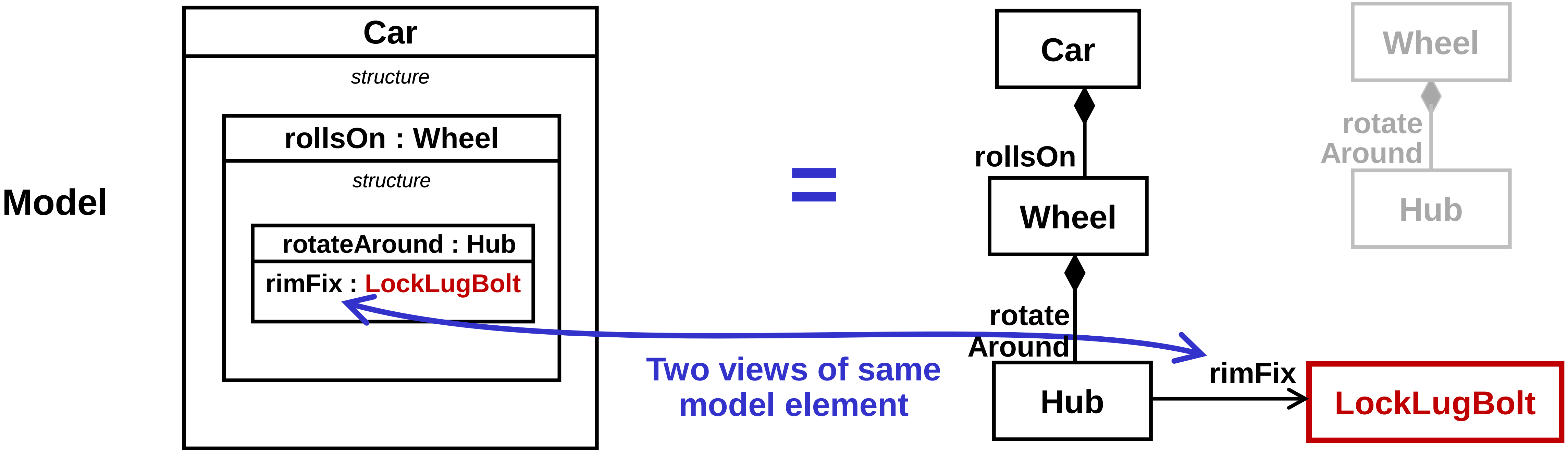


=



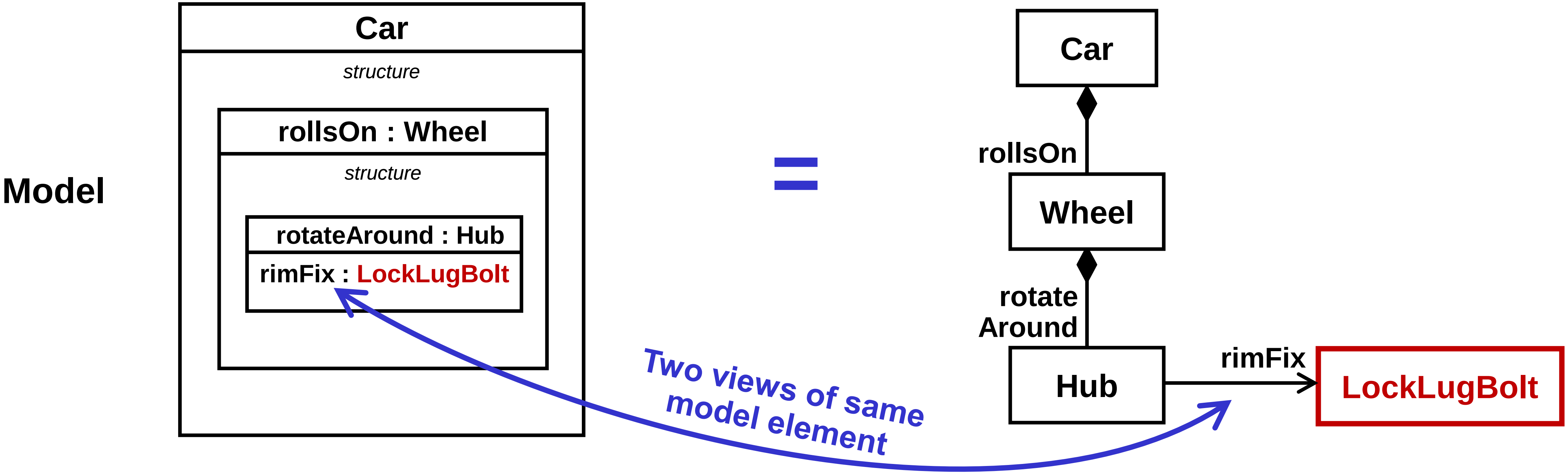
- No matter how class diagrams are drawn.

Visual Nesting ≠ Class/Property Modeling



- Don’t want
  - All hubs to use lock lugbolts.
  - All wheels to have hubs with lock lugbolts.
- Just the hubs in wheels that are in cars.

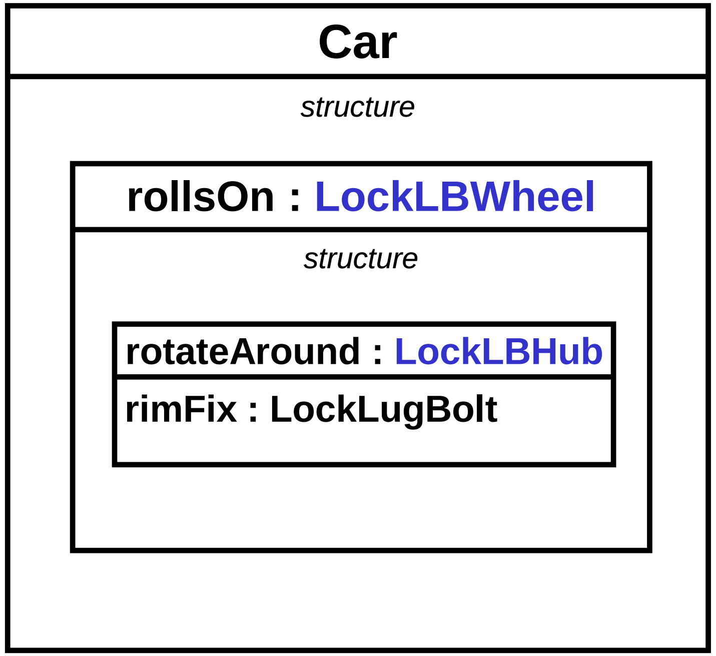
Visual Nesting ≠ Class/Property Nesting



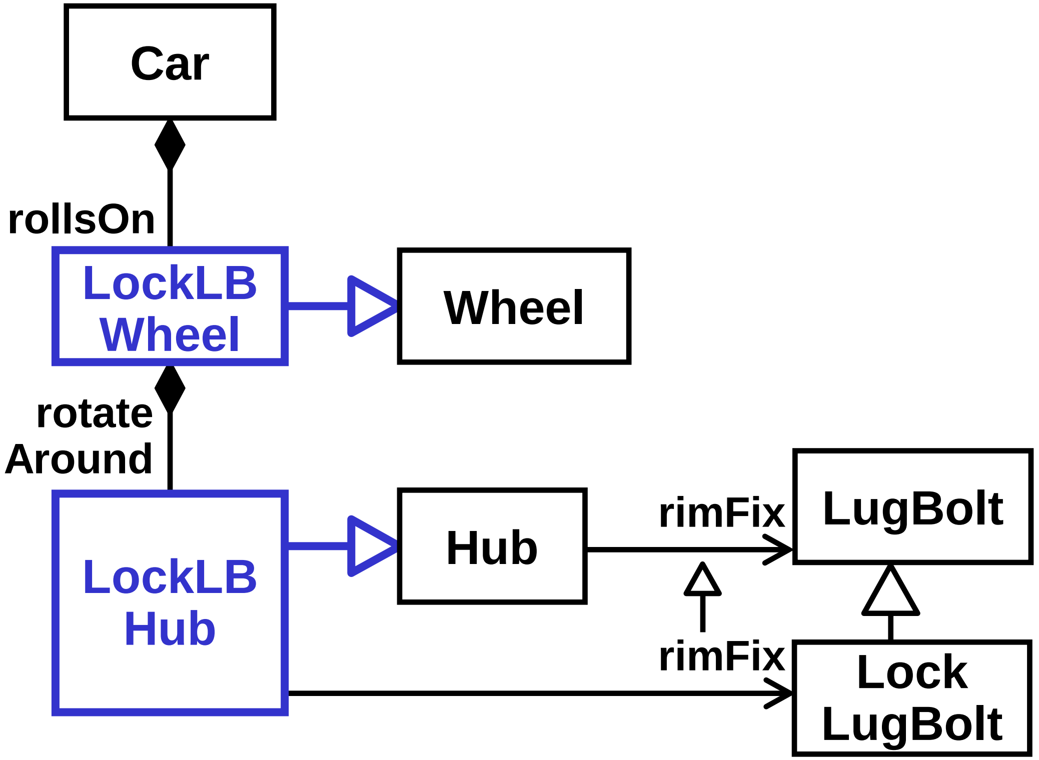
- Doesn’t matter how class diagrams are drawn.

Visual Nesting ≠ Class/Property Modeling

Model

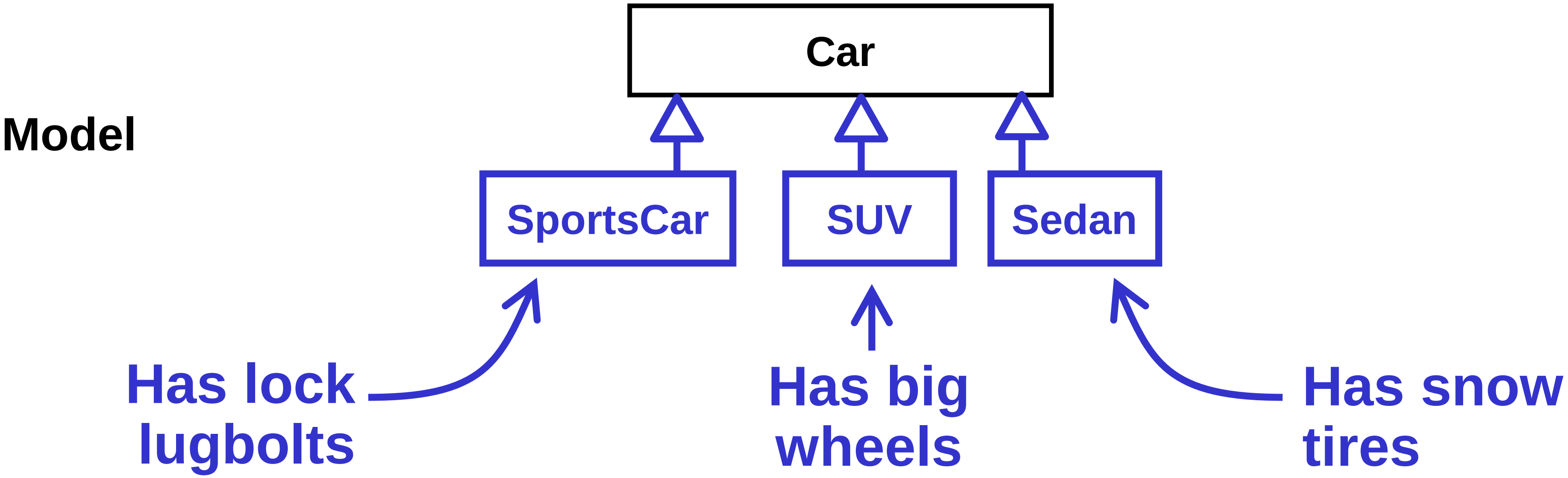


=



- Need new specialized classes all the way down the chain of properties.

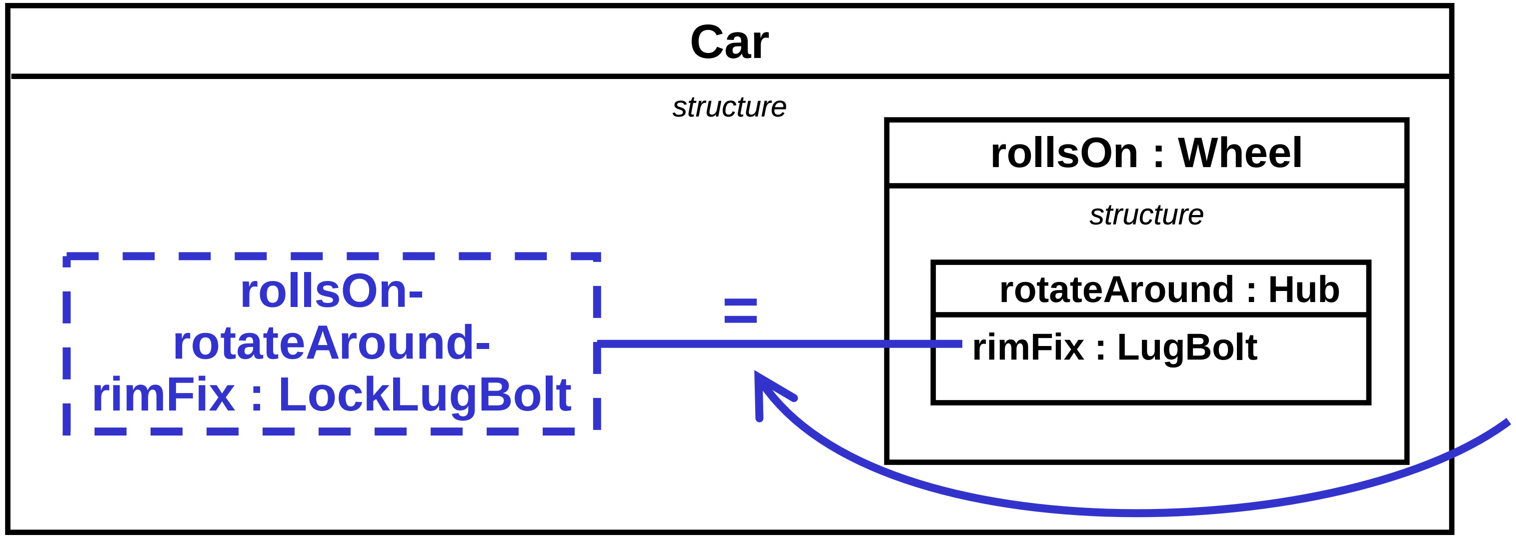
Variation Modeling



- Need classes all the way down **for all variants**.

SysML 1.4 Variant WG Archive http://www.omg.org/members/sysml-rtf-wiki/doku.php?id=rtf4:groups:variant:variants_modeling

SysML 1.x Bound References (= SST Feature Chains)

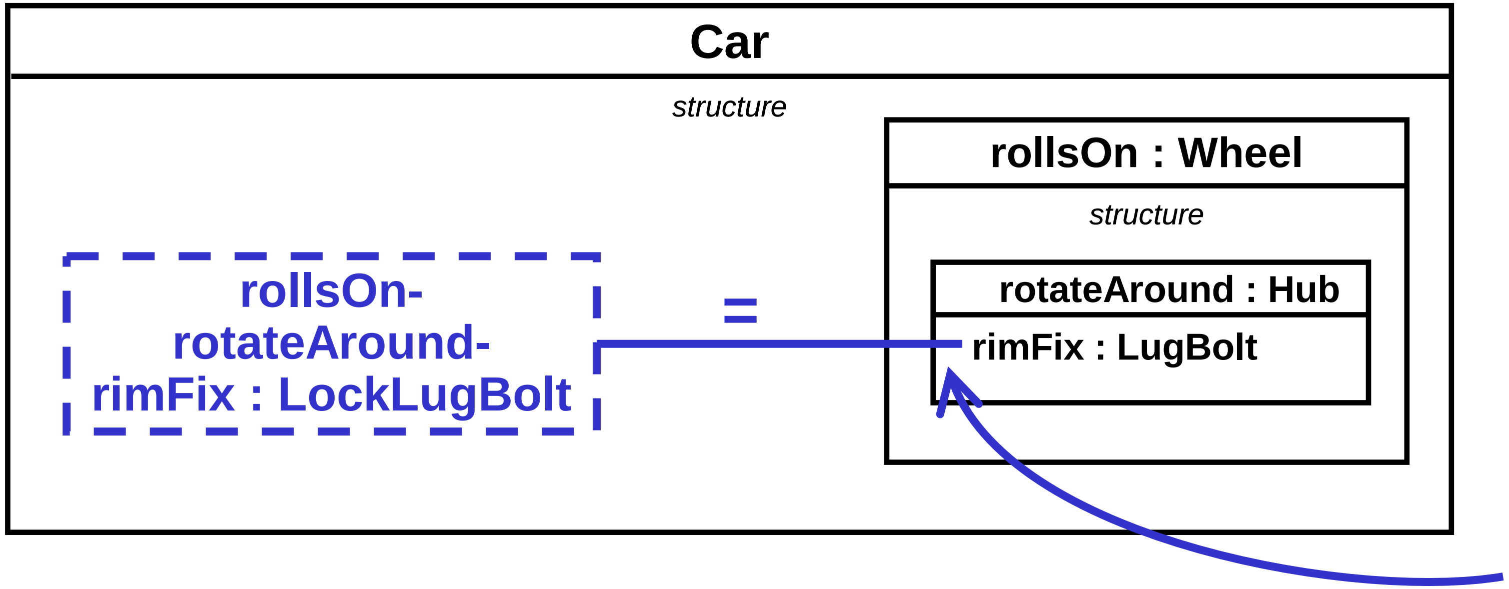


Binding means end property values are the same.

Restrictions on one apply to the other.

- **Bind new top-level property to nested one.**
  - **Restrict top-level property**
- **Pro: No new classes needed.**
- **Cons:**
  - **Restrictions on nested elements are at top-level.**
  - **Multiplicity restrictions count over all nested values.**

SysML 1.x Property Paths, Multiplicity

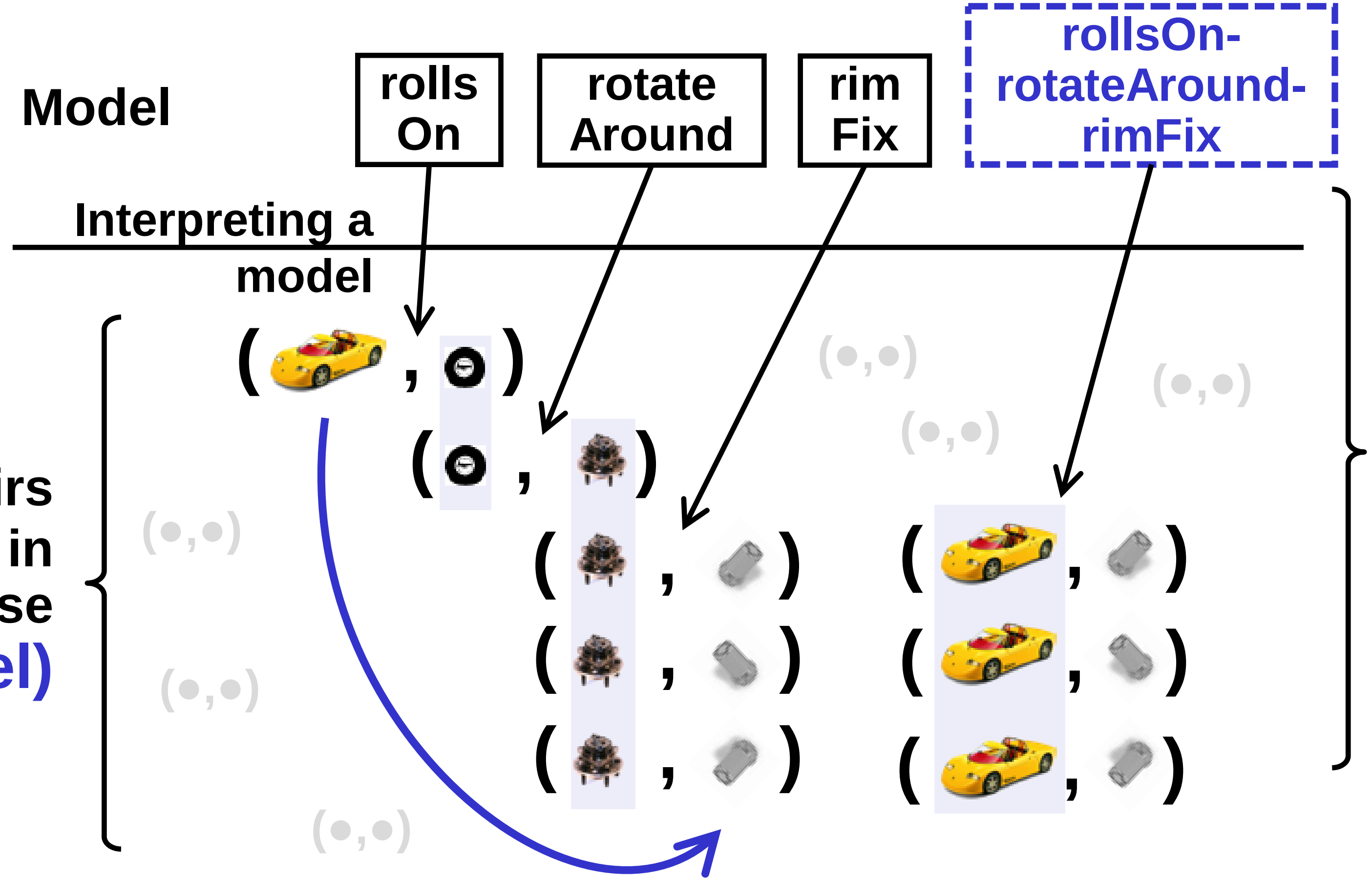


Nested connector end has property path:
(rollsOn, rotateAround, rimFix)

- **Bound values are found by “navigation” from each car.**
  - **Right end would be all lugbolts of hubs on all wheels.**
- **Don’t want multiplicity on bound reference to count all LBs.**
  - **Just the ones on each wheel.**

SysML 1.4 Variant WG Archive http://www.omg.org/members/sysml-rtf-wiki/doku.php?id=rtf4:groups:variant:variants_modeling

SysML 1.x PropertyPaths, Interpretation



Model

rolls On

rotate Around

rim Fix

rollsOn-rotateAround-rimFix

Properties

Interpreting a model

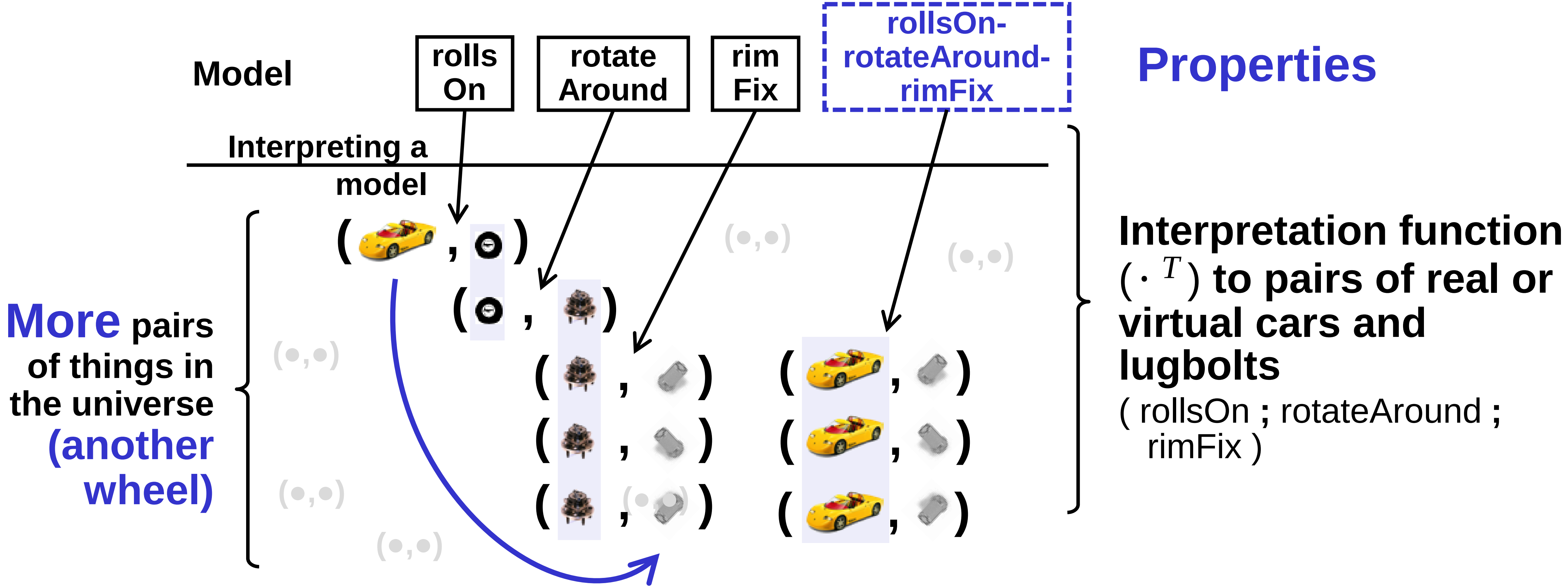
Some pairs of things in the universe **(one wheel)**

Interpretation function $(\cdot^{T})$ to pairs of real or virtual cars and lugbolts

(rollsOn ; rotateAround ; rimFix)

- **Bound reference links cars to their lugbolts**
  - It can **restrict type** of lugbolt.

SysML 1.x PropertyPaths, Interpretation



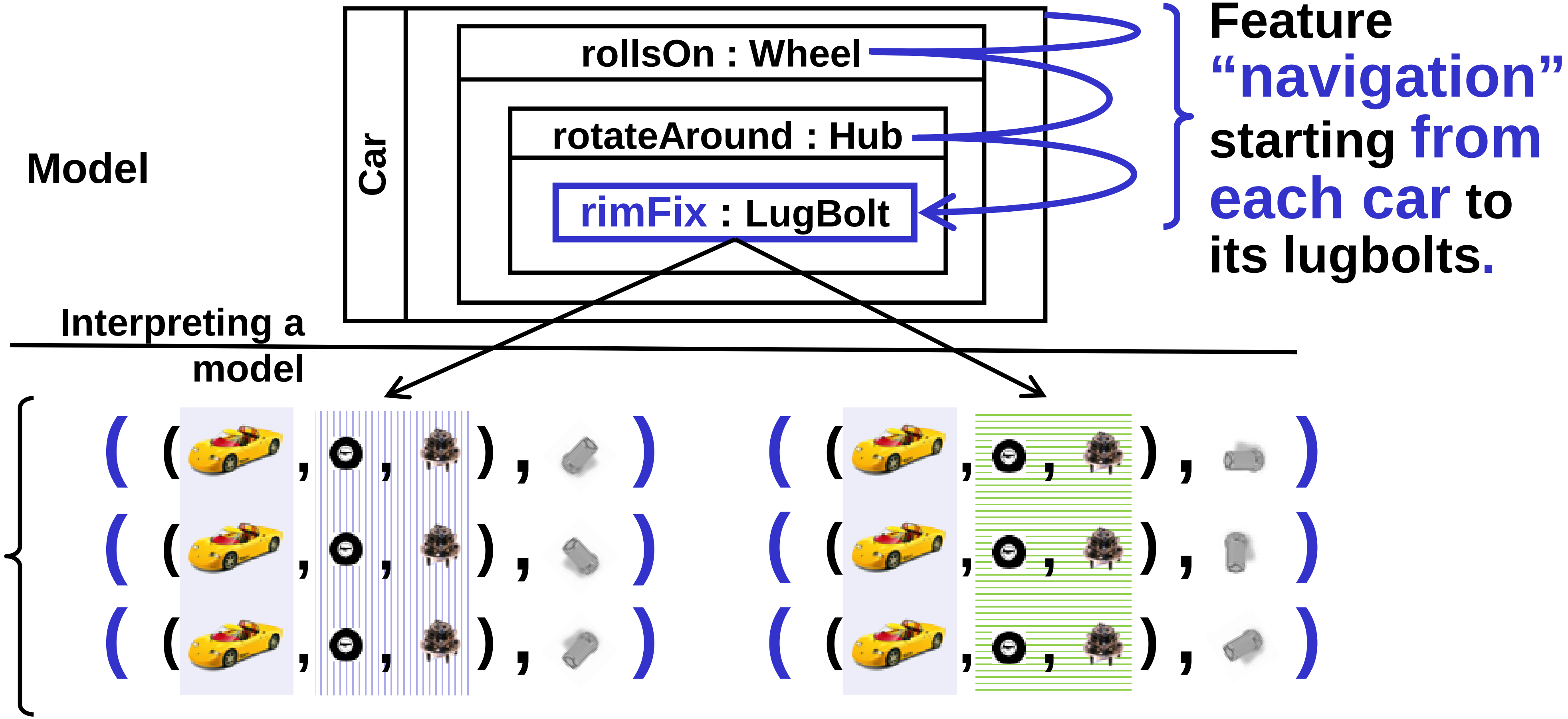
More pairs of things in the universe **(another wheel)**

Interpretation function $(\cdot \, ^T)$ **to pairs of real or virtual cars and lugbolts**

(rollsOn ; rotateAround ; rimFix)

- Bound reference links cars to **all** their lugbolts
  - Restrictions apply **to all hubs of all wheels**.
  - Maybe OK for type, but probably **not for multiplicity**.

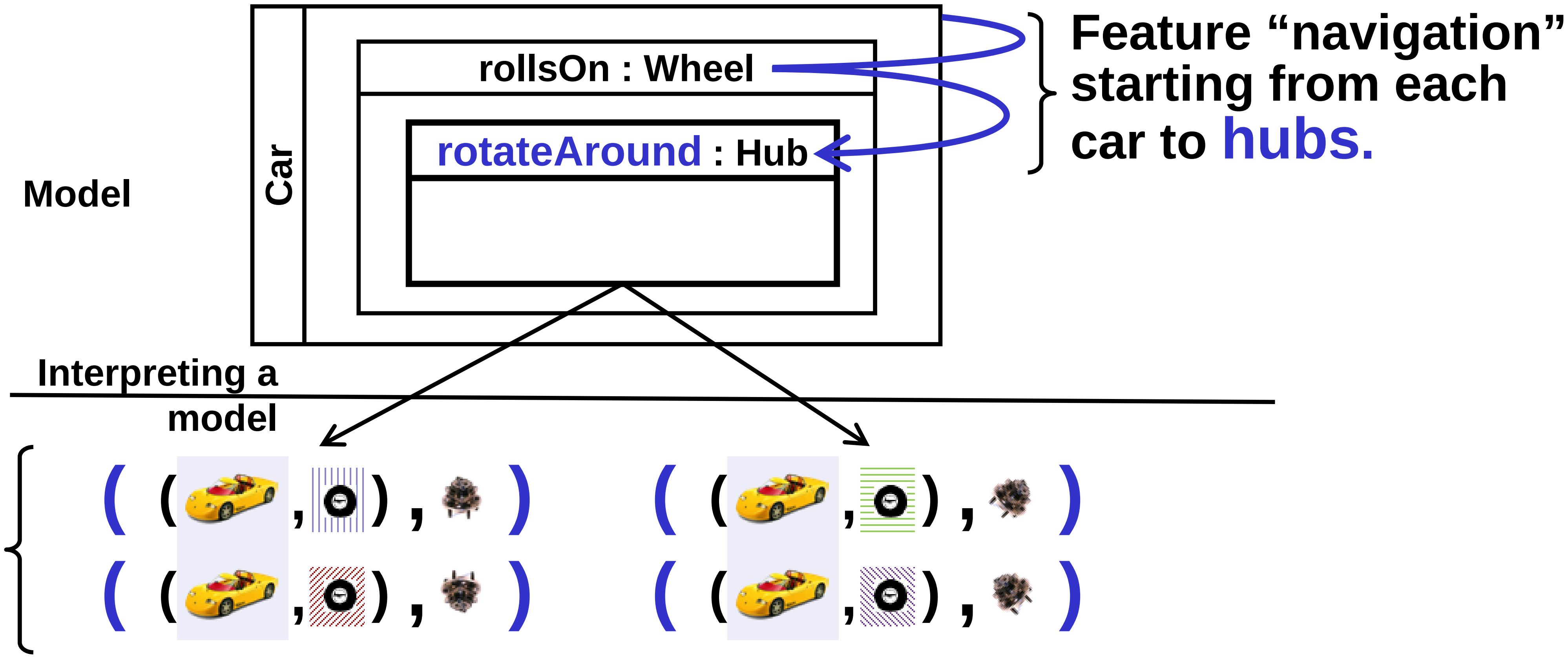
“Nested” Features, Interpretation



Model
Car
rollsOn : Wheel
rotateAround : Hub
rimFix : LugBolt
Feature “navigation” starting from each car to its lugbolts.
Interpreting a model
Pairing things with sequences of things in the universe (two wheels)

- Lugbolts paired with sequences of “navigations” to each.
  - Restrictions apply to each hub separately.
  - Works for types and multiplicity.

Less “Nested” Features, Interpretation

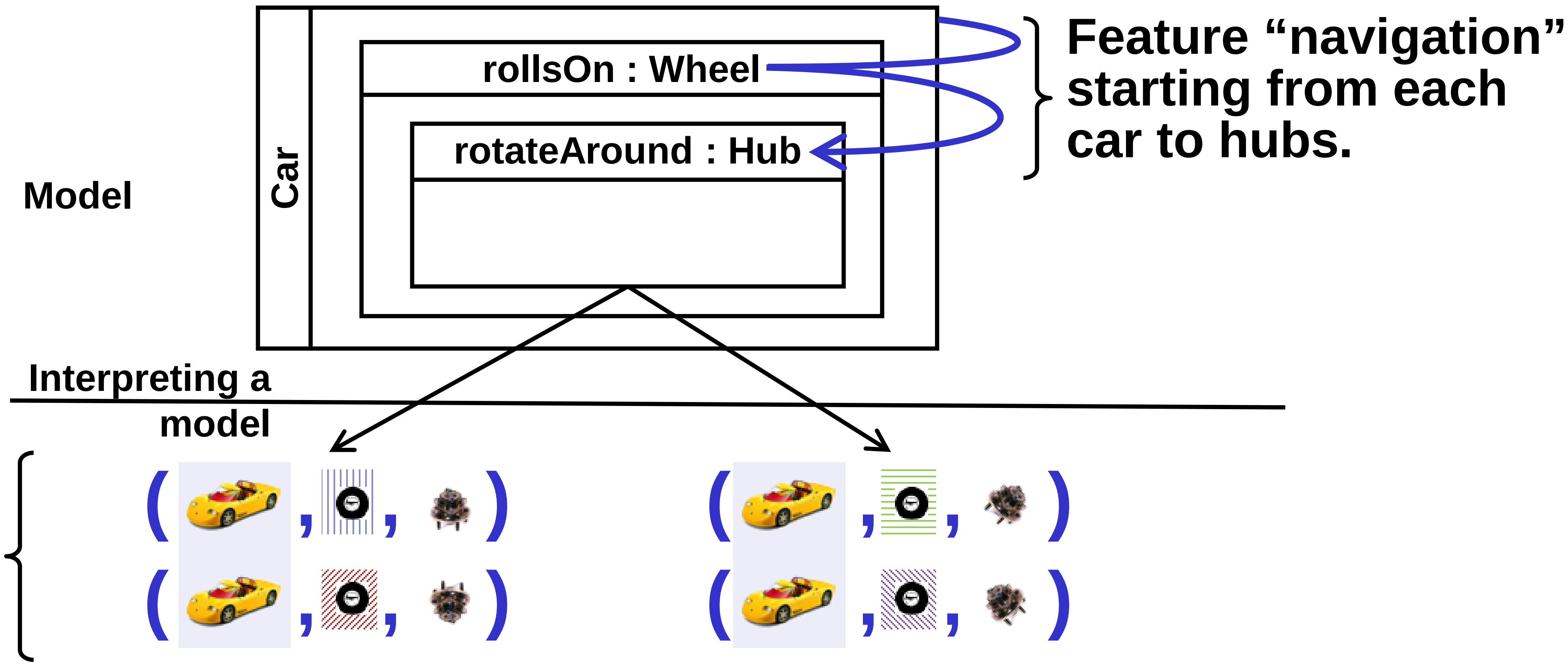


Feature “navigation” starting from each car to **hubs.**

Pairing things with pairs of things in the universe (two wheels)

- § **Hubs paired with sequences “navigating” to each.**

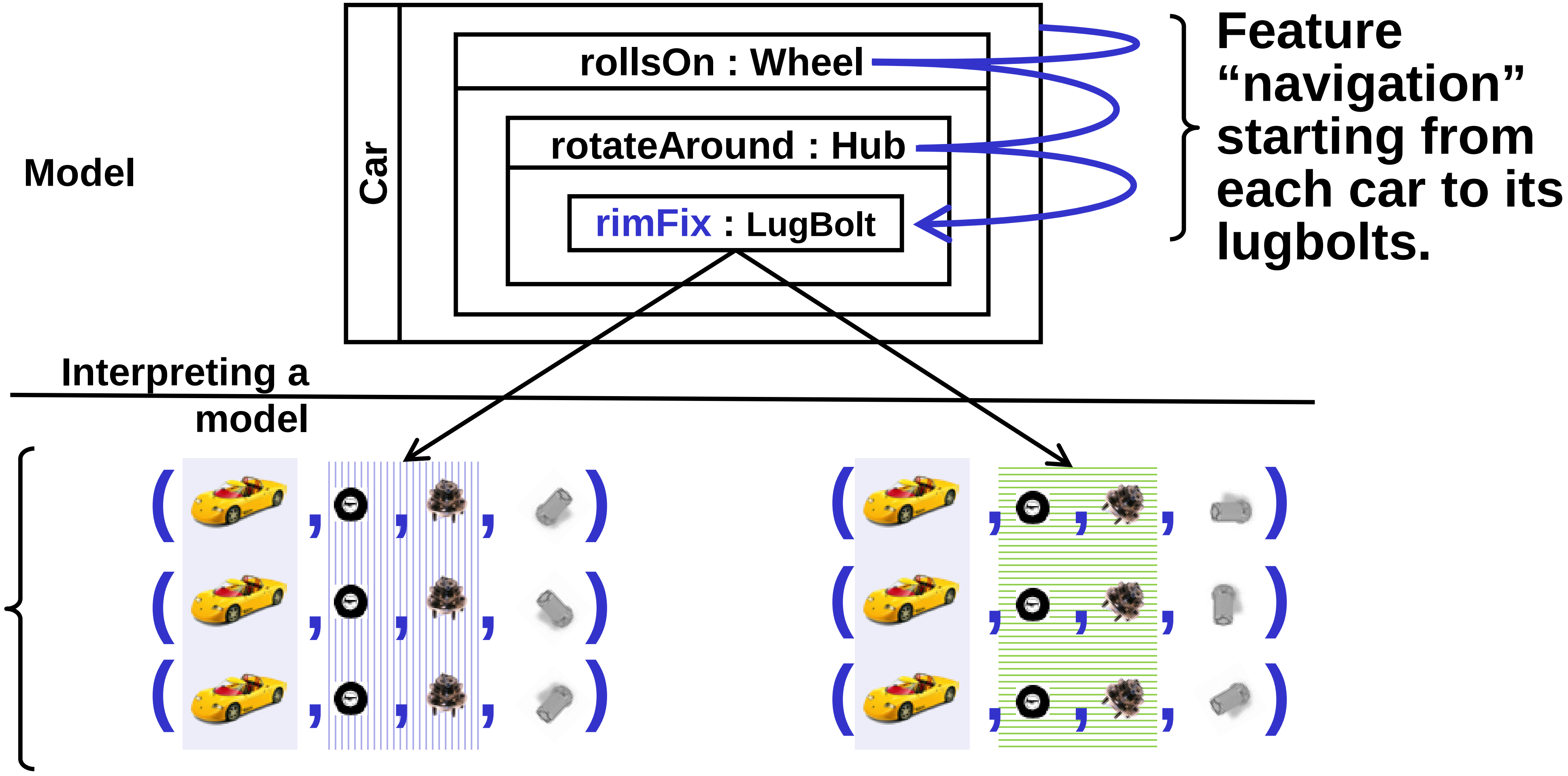
SysML 2 Less “Nested” Features, Interpretation



Model
Car
rollsOn : Wheel
rotateAround : Hub
Feature “navigation” starting from each car to hubs.
Interpreting a model
Sequences of things in the universe (four wheels)
(, ,)
(, ,)
(, ,)
(, ,)

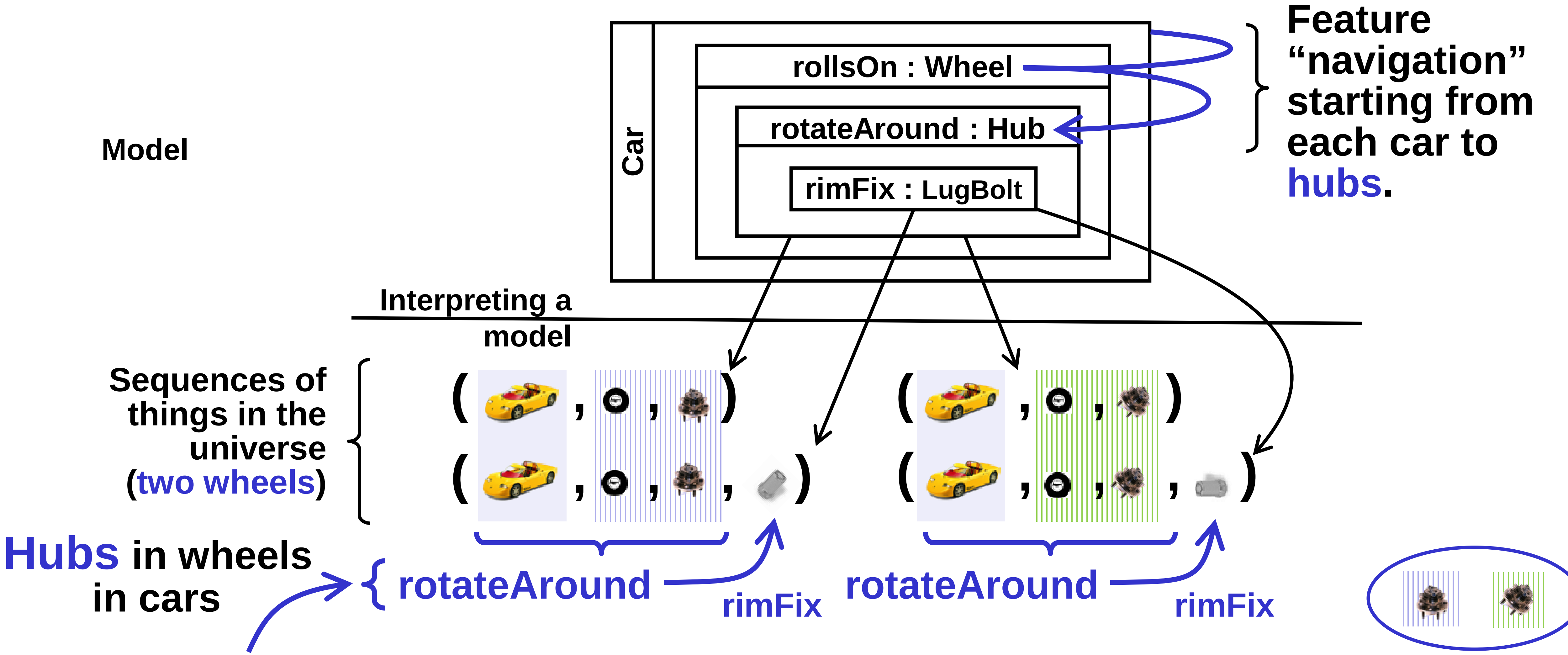
- Hubs at end of sequences “navigating” to them.
  - No nested pairs.

SysML 2 “Nested” Features, Interpretation


Model
Car
rollsOn : Wheel
rotateAround : Hub
rimFix : LugBolt
Feature “navigation” starting from each car to its lugbolts.
Interpreting a model
Sequences of things in the universe (two wheels)

§ Lugbolts at end of sequences “navigating” to them.

SysML 2 Features as “Classifiers” ?



Model

Car

rollsOn : Wheel

rotateAround : Hub

rimFix : LugBolt

Feature “navigation” starting from each car to hubs.

Interpreting a model

Sequences of things in the universe (two wheels)

Hubs in wheels in cars

{ rotateAround

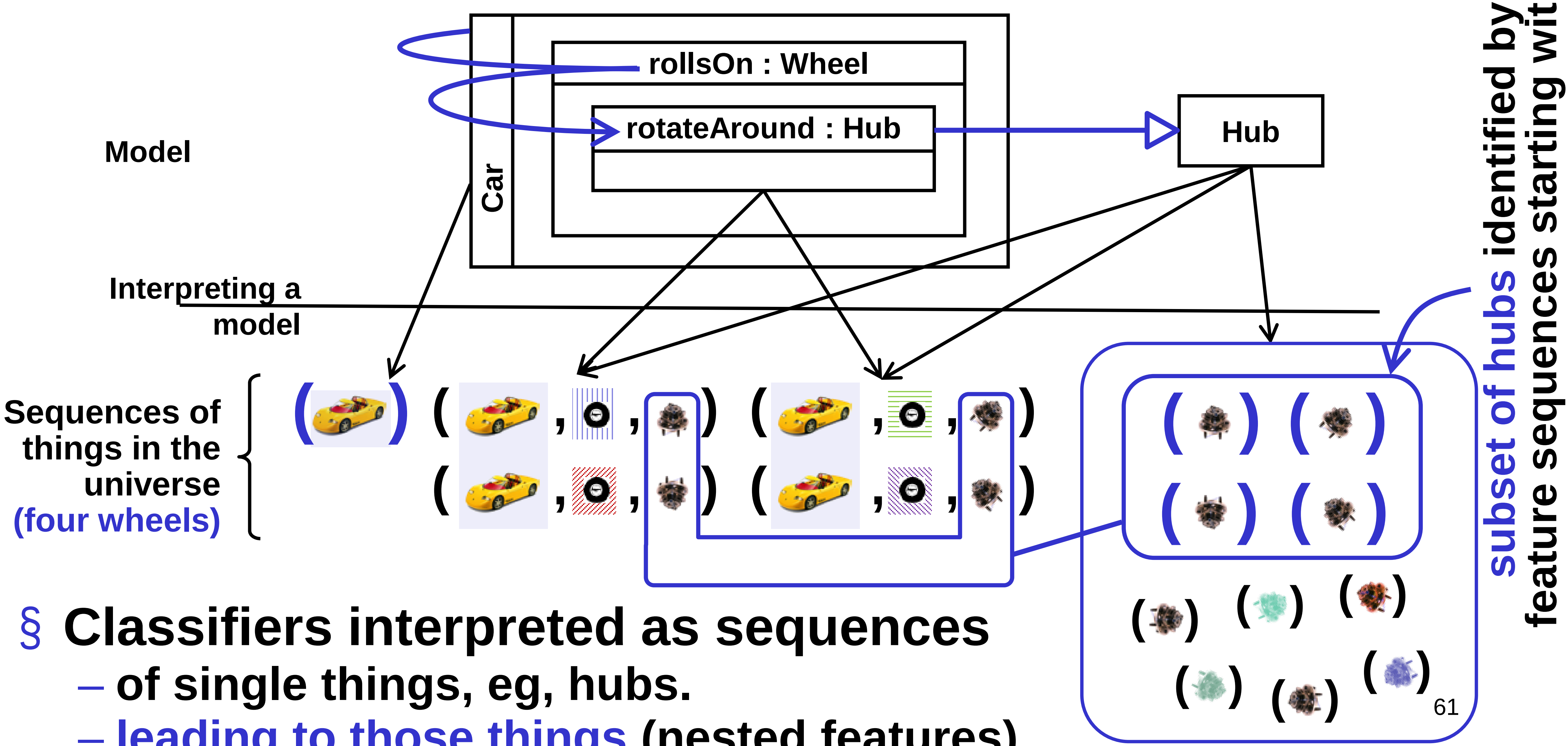
rimFix

rotateAround

rimFix

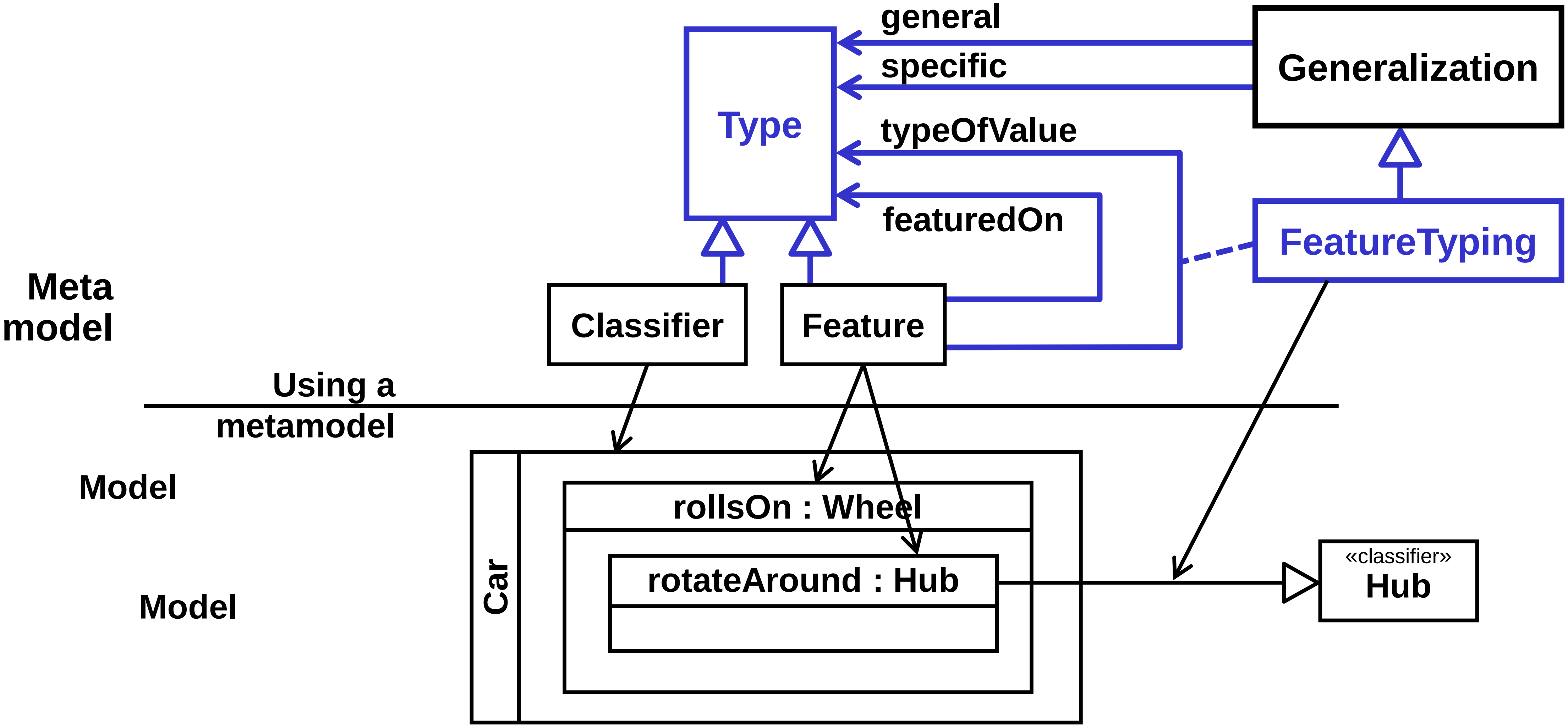
- Nested rotateAround sequences identify a subset of hubs
  - … without additional classes.

SysML 2 Features as “Classifiers” ?


Model
Car
rollsOn : Wheel
rotateAround : Hub
Hub
Interpreting a model
Sequences of things in the universe
(four wheels)
subset of hubs identified by feature sequences starting with

- **Classifiers interpreted as sequences**
  - **of single things, eg, hubs.**
  - **leading to those things (nested features)**

SysML 2 Features, Classifiers as Types



- Metamodel : Feature , Classifier are disjoint
- Model : Features, Classifiers are not.

SysML 2 Classifier, Feature Math

7.3.3.4 Semantics

Classifier ~~Type~~ Semantics

The interpretation of the Classifiers in a model shall satisfy the following rules:

1. If the interpretation of a Classifier includes a sequence, it must also include the 1-tail of that sequence.

$$\boxed{\forall c \in V_C,\ s_1 \in S \quad s_1 \in (c)^T \Rightarrow (\forall s_2 \in S \ \ \mathit{tail}(s_2,\ s_1) \wedge \mathit{length}(s_2) = 1 \Rightarrow s_2 \in (c)^T)}$$

7.3.4.4 Semantics

Feature Semantics

The interpretation of the Features in a model shall satisfy the following rule:

1. The interpretations of features must have length greater than one.

$$\boxed{\forall s \in S,\ f \in V_F \quad s \in (f)^T \Rightarrow \mathit{length}(s) > 1}$$

Overview

- Motivation / Problem
  - Modeling Languages and Analysis
  - Interpreting Models (Semantics)
- Solution
  - Standardizing Semantics
  - Logical Classification
  - Semantics, Without Math
  - SysML 2 Semantics
- **Summary**

Summary

- Language designers and analysis tool builders.
  - Expectations for system construction / operation ...
  - … coordinated through a standards specifications.
- Interpreting models
  - Real or virtual systems built/operating according to model ...
  - … checked against the model and language semantics.
  - Conformance (checking) = classification (yes/no).
- Specifying semantics
  - Classifying (pairs of) things in a hypothetical universe.

Summary, SysML 2

- **Semantic framework, motivation**
  - Classifying **sequences** of things in a hypothetical universe ...
  - … to model subsets of things reached by feature "**navigation**" …
  - … **without** additional classes. Facilitates variation modeling.
- **Features and Classifiers**
  - **Features** interpreted as sequences **longer than one**.
  - **Classifiers** interpreted as sequences of **exactly one thing +** …
  - … all feature sequences **ending in those things**.
  - Enables features to be "classifers" for other ("**nested**") features.
  - Kinds of feature values (typing) = **Generalization**

Other Information

- OWL 2 Direct Semantics
  - https://www.w3.org/TR/owl2-direct-semantics/
- Introduction to Reasoning
  - Section 3.1 in Bock, et al, “Evaulating Reasoning Systems,” NISTIR 7310 https://www.nist.gov/publications/evaluating-reasoning-systems
- SysML 1.4 Variant WG Archive
  - http://www.omg.org/members/sysml-rtf-wiki/doku.php?id=rtf4:groups:variant:variants_modeling
  - Scroll down for literature and presentations.
  - Discussion deck: http://tinyurl.com/ybxlc2wy
    - Bound references on slides 12-44.

https://www.omg.org/members/sysml-rtf-wiki/lib/exe/fetch.php?id=rtf4%3Agroups%3Avariant%3Avariants_modeling&cache=cache&media=rtf4:groups:variant:variant-wg-130619a.ppt