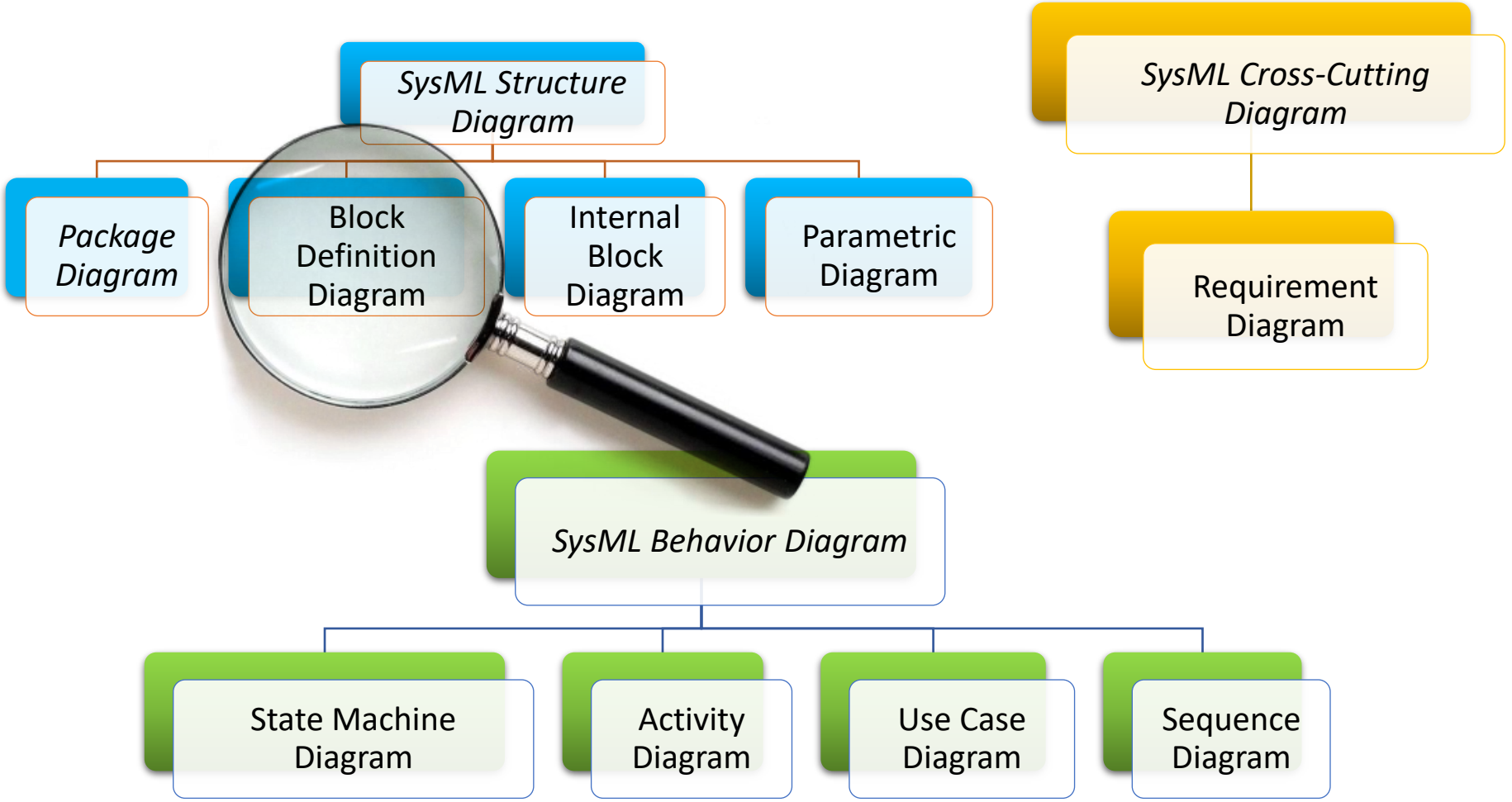
SysML v1 Blocks

Ansgar Radermacher / Asma Smaoui
ansgar.radermacher@cea.fr

- **Acknowledgments** – contains material from our CEA colleagues Shuai Li, Jérémie Tatibouët, François Terrier, Sébastien Gérard

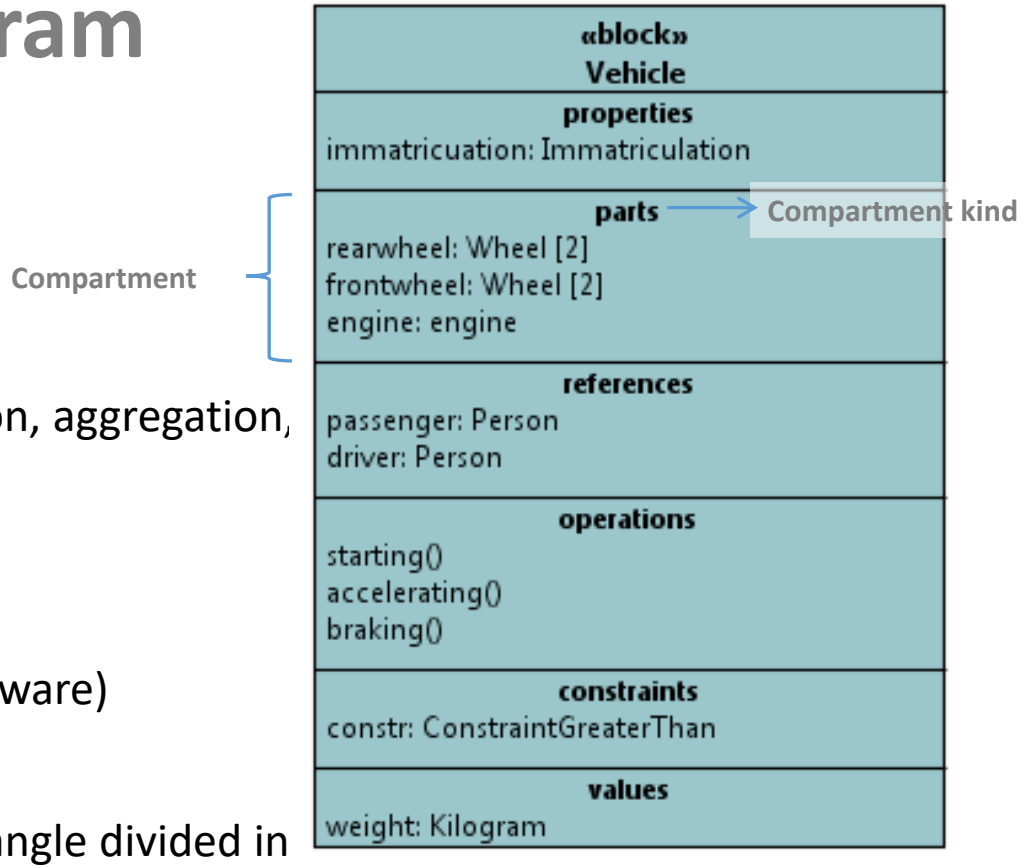
Block Definition Diagram



Block Definition Diagram

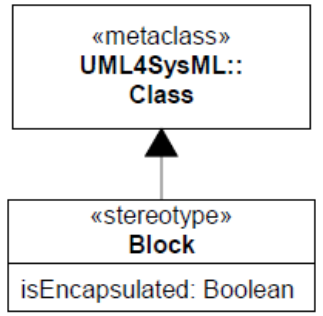
Block Definition Diagram (BDD)

- Represent blocks…
- …their properties and their relationships (decomposition, aggregation,



Block

- Extension of the UML meta-class Class.
- Basic entity, no restriction on its nature (hardware, software)
- Definition of a type, reusable in multiple contexts
- Notation: inside a BDD a block is represented by a rectangle divided in only obligatory compartment is the compartment for the block name.



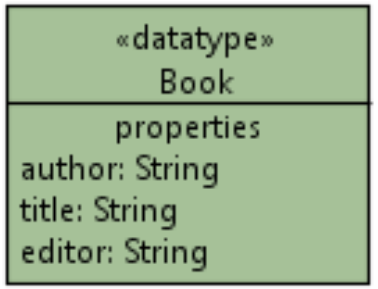
Basic Types

PrimitiveType

- No properties / no operations

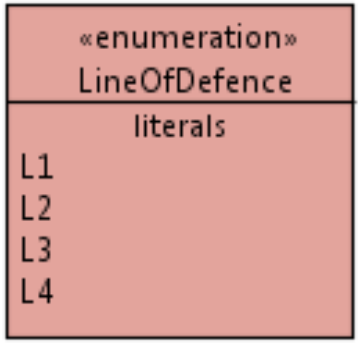
DataType

- Structured type
- Properties
- May have operations



Enumeration

- Finite number of possible values (literals)
- No properties / no operations



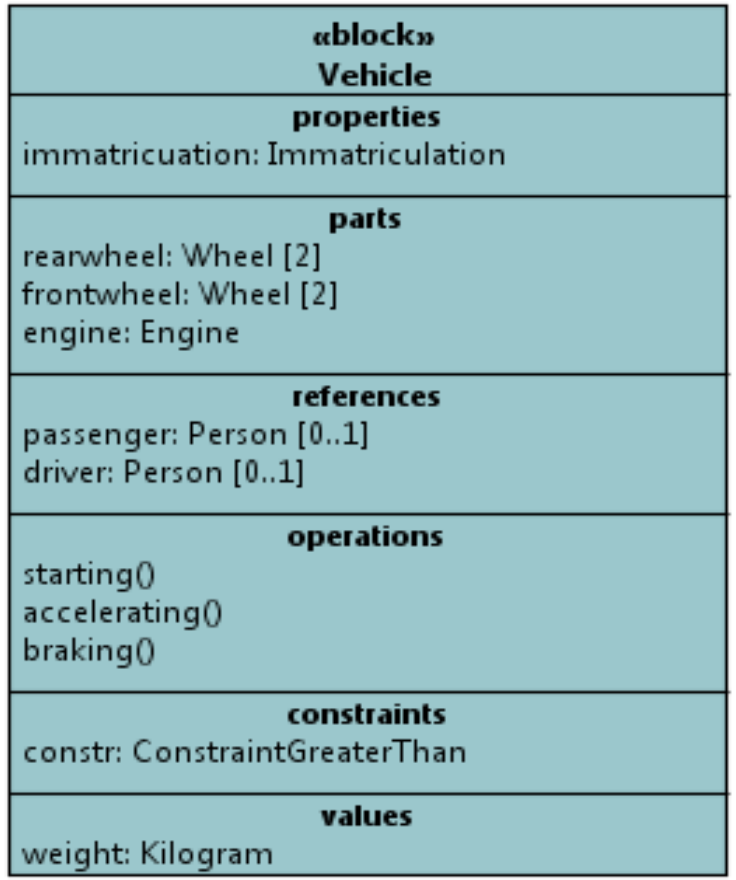
Block FEATURES

Block extends Class so it has…

- Properties
- Operations

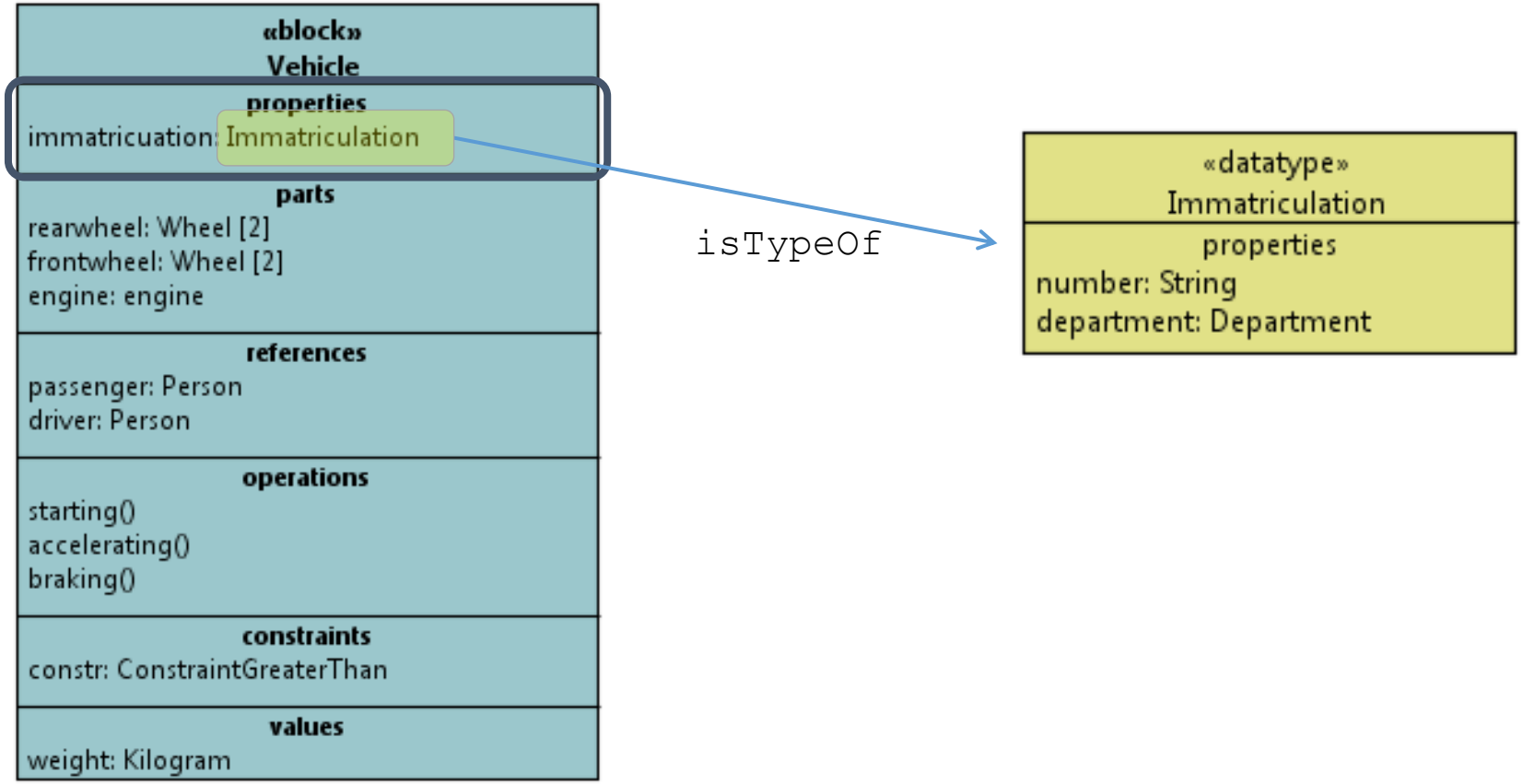
Block specializes properties

- Part properties
- Reference properties
- Constraint properties
- Values properties



Block FEATURES

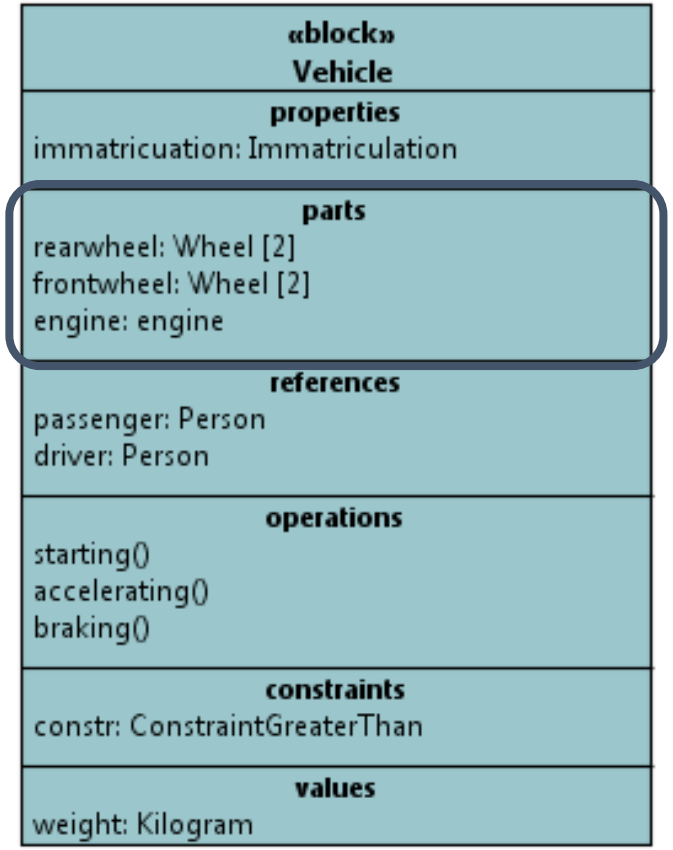
- **Simple properties: always typed properties**



Block FEATURES

As in standard UML

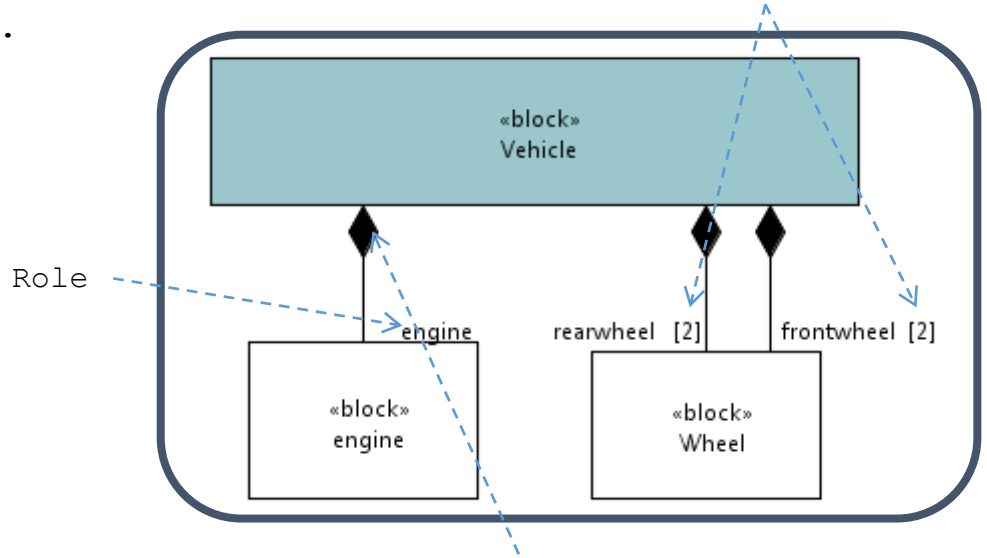
- **Part properties describe composition**



Composition: used to model decomposition of blocks (containment relationship)

- Specifies a multiplicity
- Specifies a role

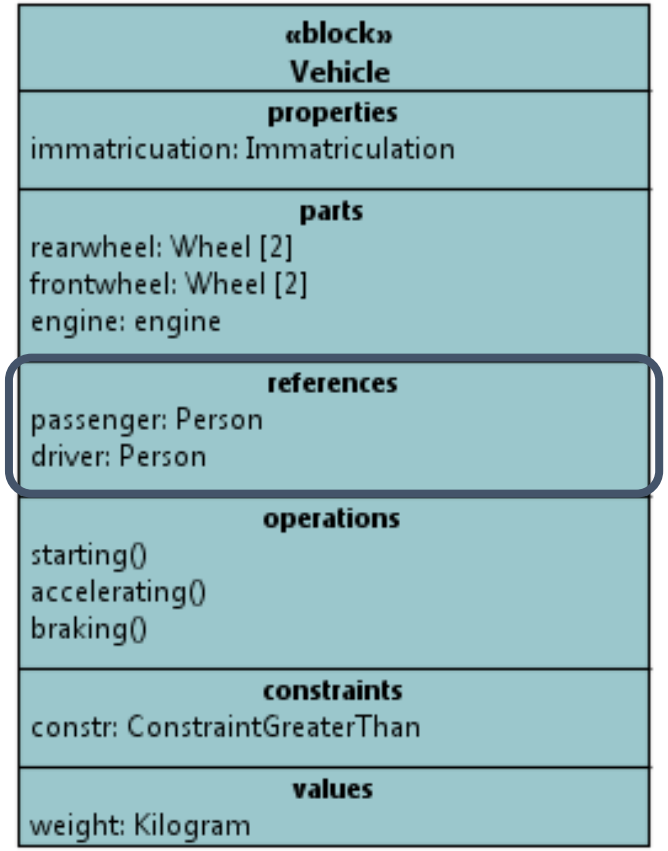
Multiplicity specifies, in the form of an interval, the number of instances of a block that can be contained in an instance of another block. Here, Vehicle has 2 rear wheels and 2 front wheels.



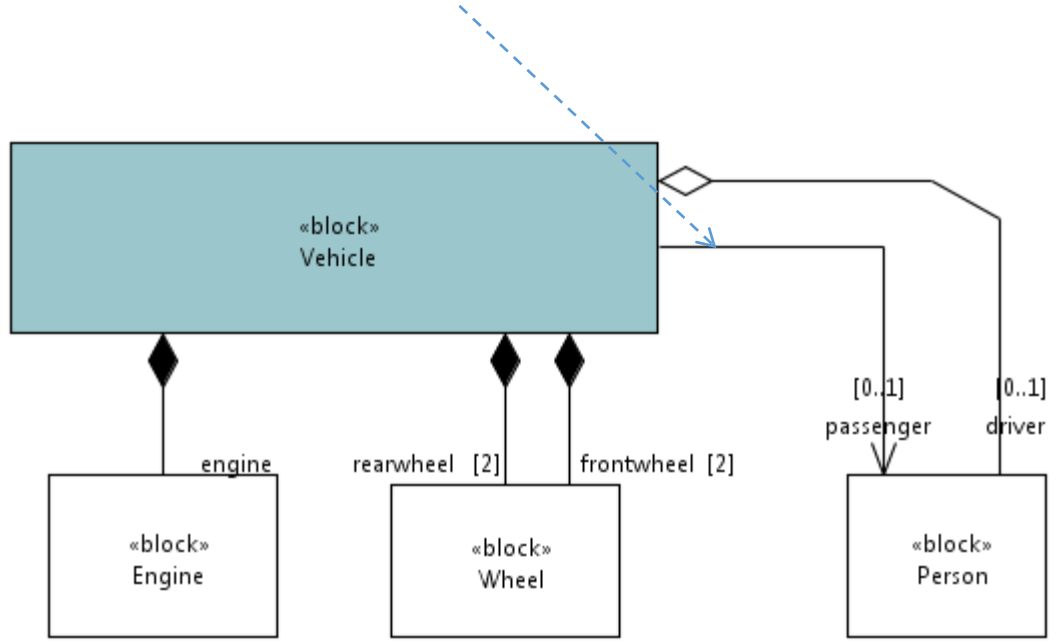
Filled diamond: if an instance of Vehicle is destroyed, the contained instances of Engine is also destroyed.

Block FEATURES

As in standard UML

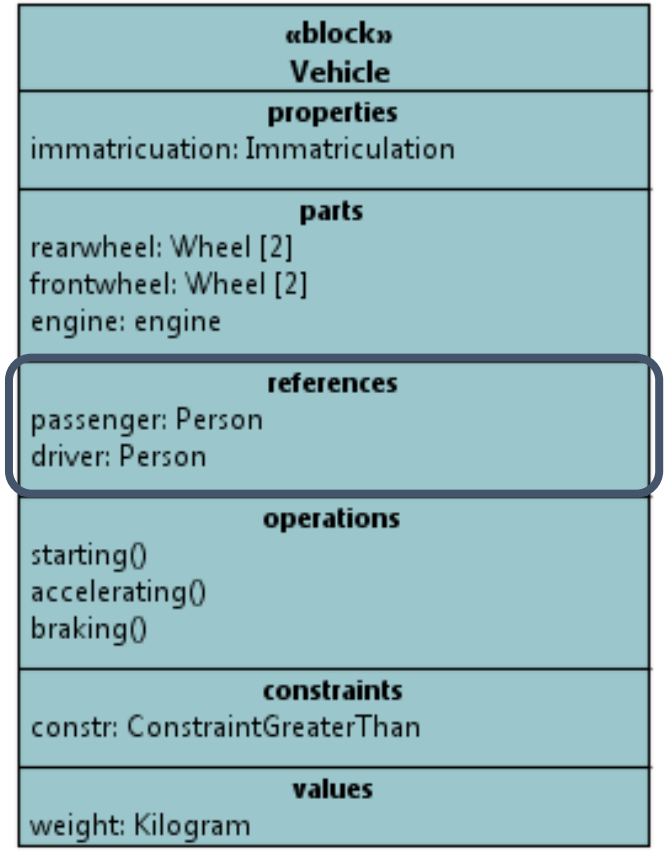


Association: simple reference (no containment relationship between blocks)



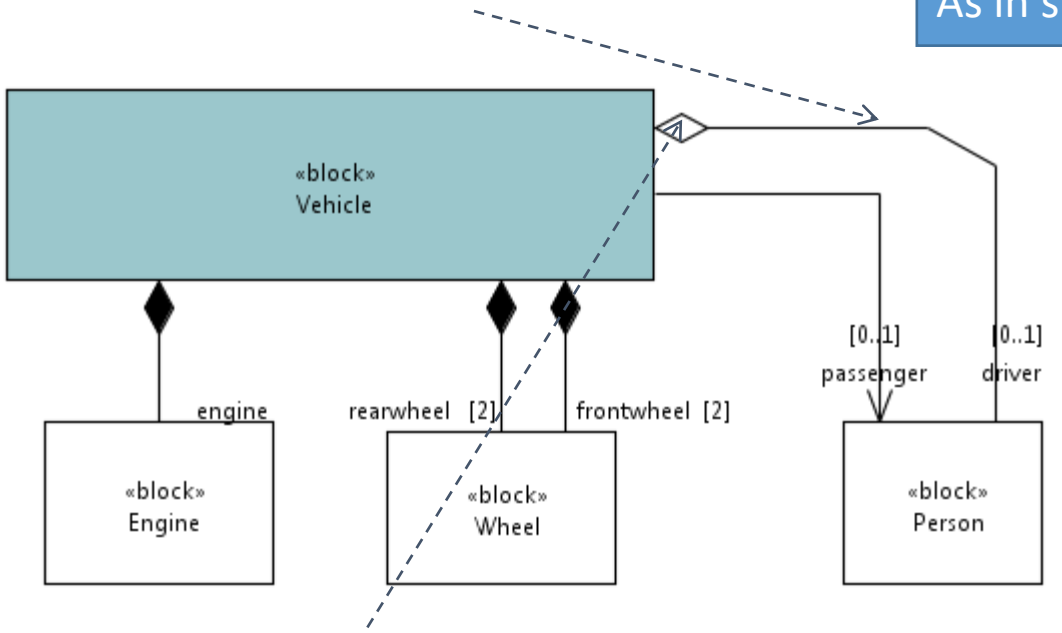
Block FEATURES

Part properties: describe associations and simple aggregations



Aggregation: kind of virtual "composition" (notion of containment), i.e. the referenced block is shared with other blocks

As in standard UML

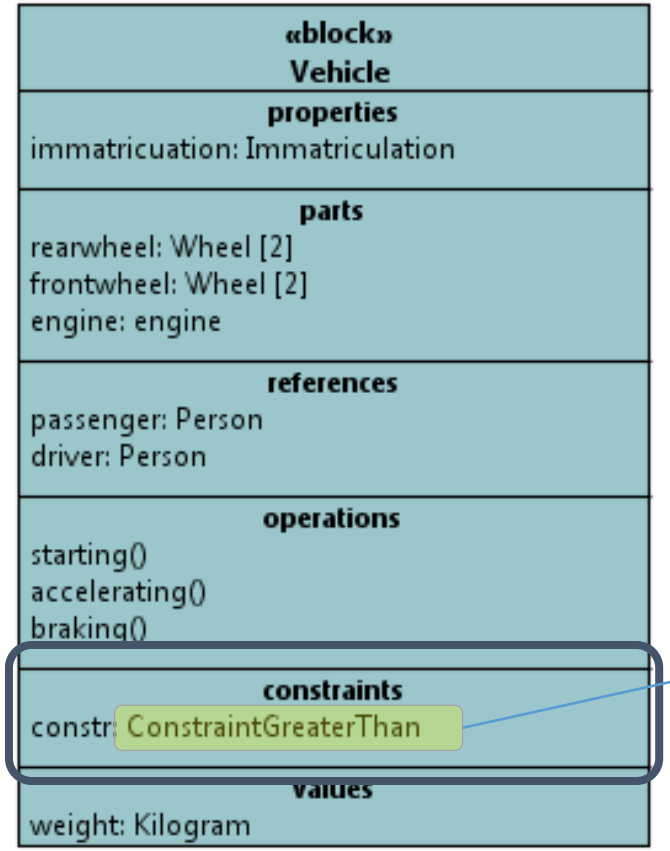


Empty diamond: if an instance of Vehicle is destroyed, the contained instances of Person are not destroyed.

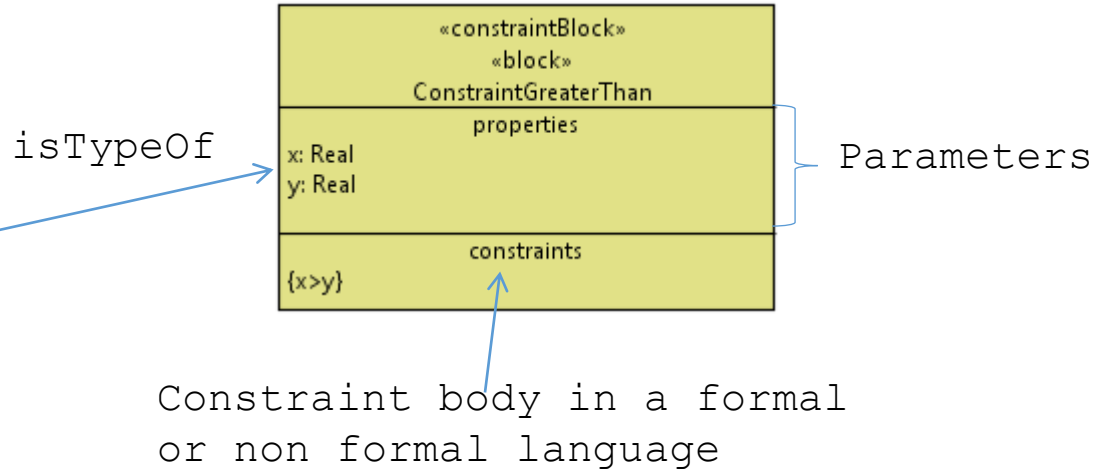
(Hopefully ☺)

Block FEATURES

Constraint properties: typed by constraint block



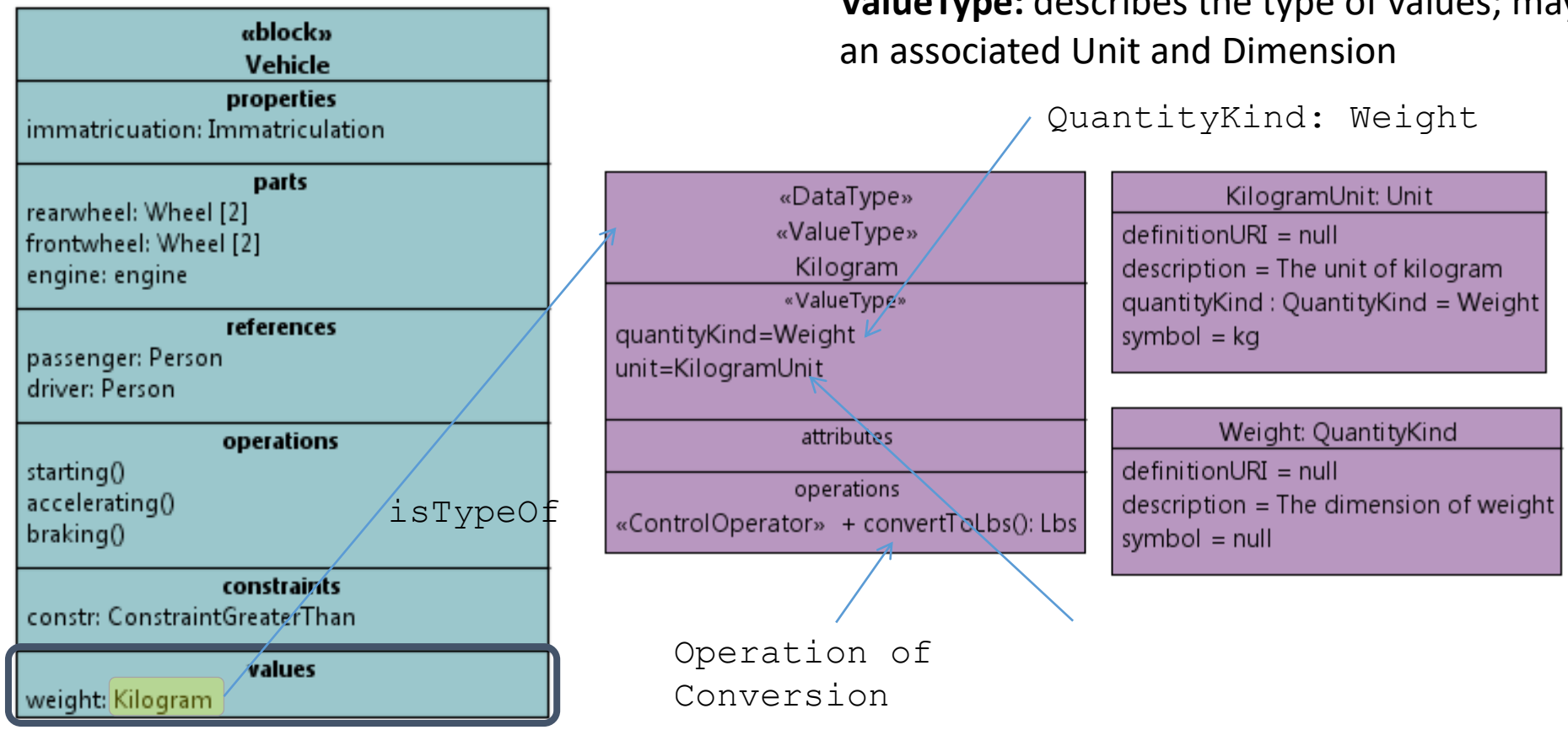
- **Constraint block**
  - Specify network of constraints that represent mathematical expressions
  - Constrain physical properties of a system
  - Define generic forms of constraints that can be used in multiple contexts.



Block FEATURES

- **Constraint properties: typed by a ValueType**

ValueType: describes the type of values; may have an associated Unit and Dimension

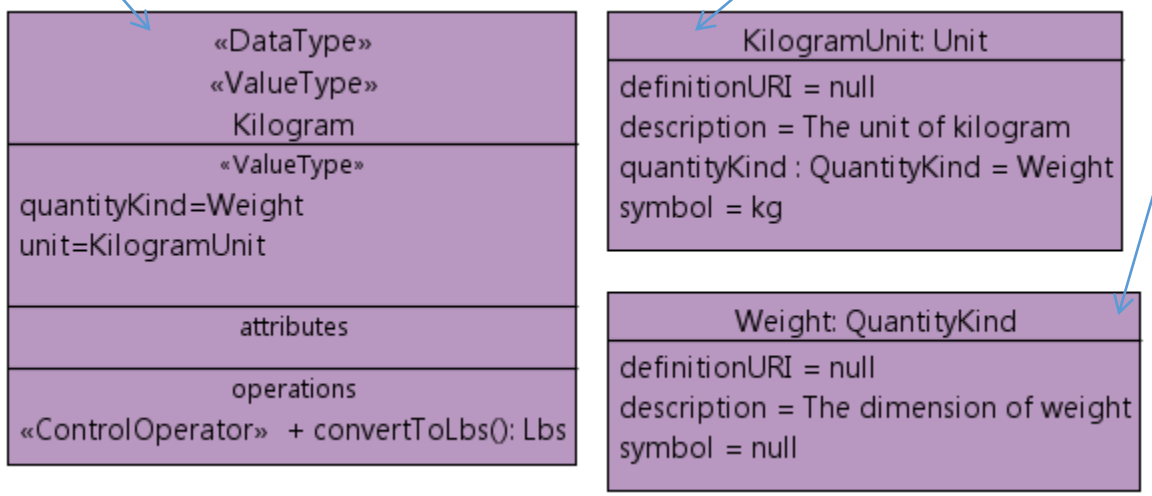


Dimensions and Units

- QuantityKind: Kind of quantity that may be stated by means of defined units. For example, the quantity kind ‘length” may be measured by units of meters, kilometers, or feet.
- Unit: Quantity in terms of which the magnitudes of other quantities that have the same dimension can be stated.

<<ValueType>> applied on a DataType

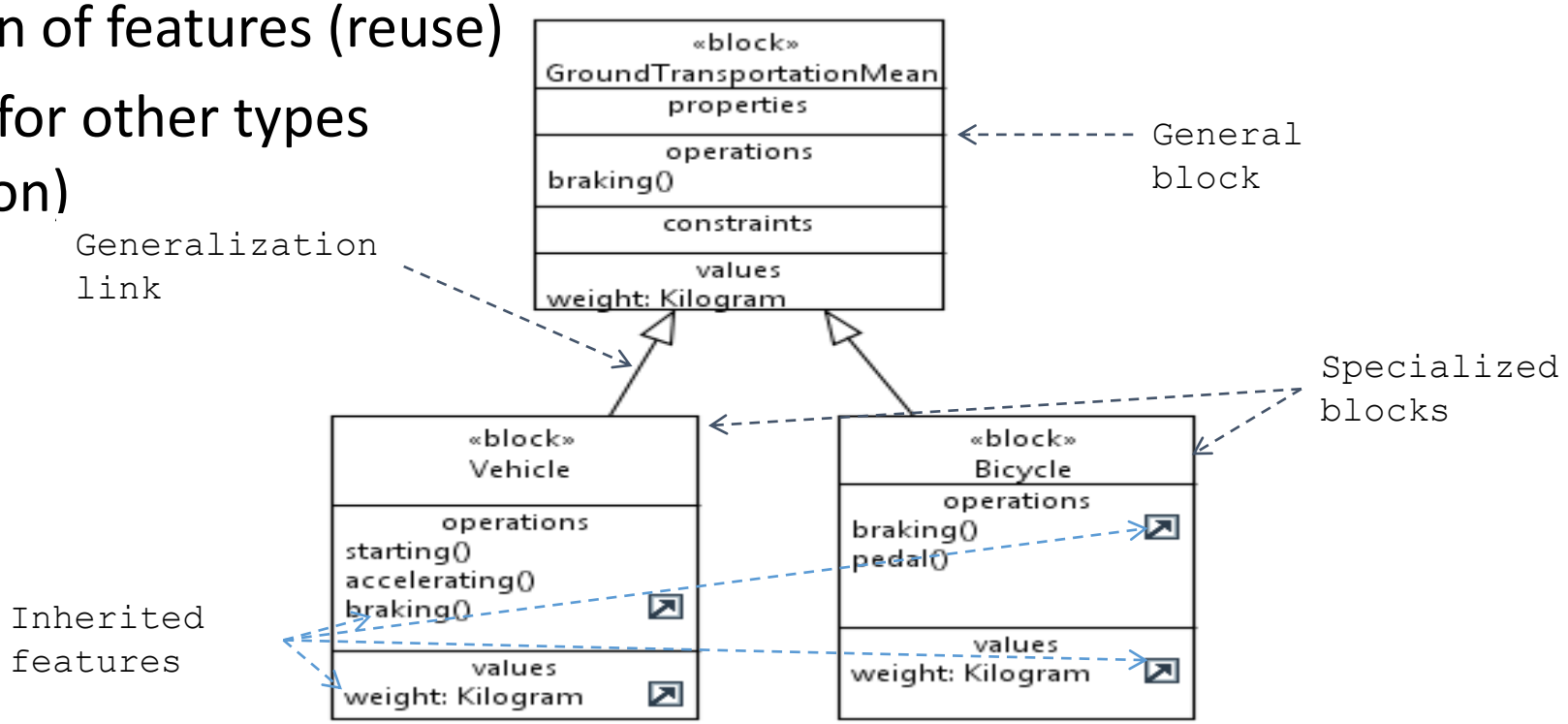
InstanceSpecifications of blocks “Unit” and “QuantityKind” defined in SysML model libraries.



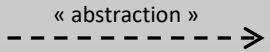
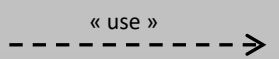
Generalization

As in standard UML

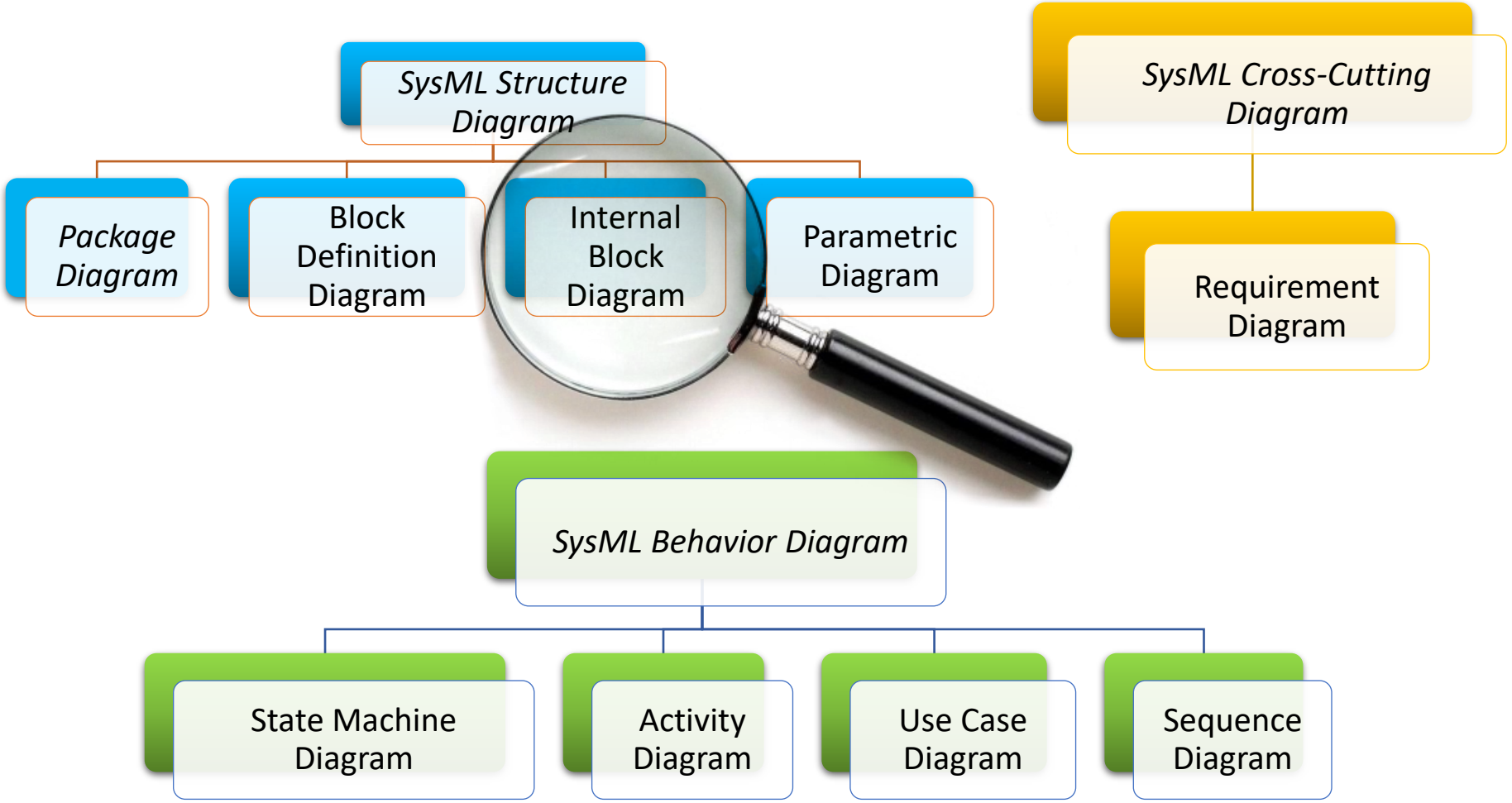
- Like a Class a Block may be a specialization of other Block(s)
- Specialized Blocks inherit features (properties, operations) and add their own
- Each instance of the specific block is also an instance of the general block
- Means for factorization of features (reuse)
- Reminder: works also for other types (except for Enumeration)



Other Relationships

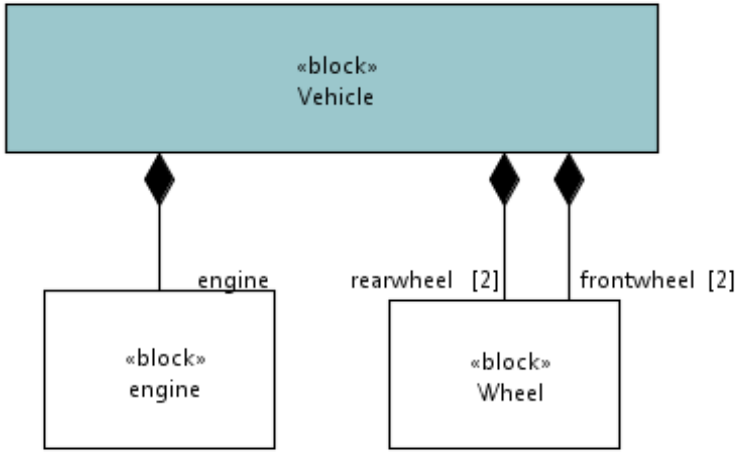
Name	Description	Notation
Dependency	Relationship meaning that a single or a set of model elements requires other model elements for their specification/implementation.	
Abstraction	Relationship that relates two elements or sets of elements that represent the same concept at different levels of abstraction.	
Realization	Specialization of abstraction relationship between two sets of model elements, one representing a specification (the supplier) and the other representing an implementation of the latter (the client).	
Usage	Relationship in which one element requires another element (or set of elements) for its full implementation or operation.	

Block Definition Diagram

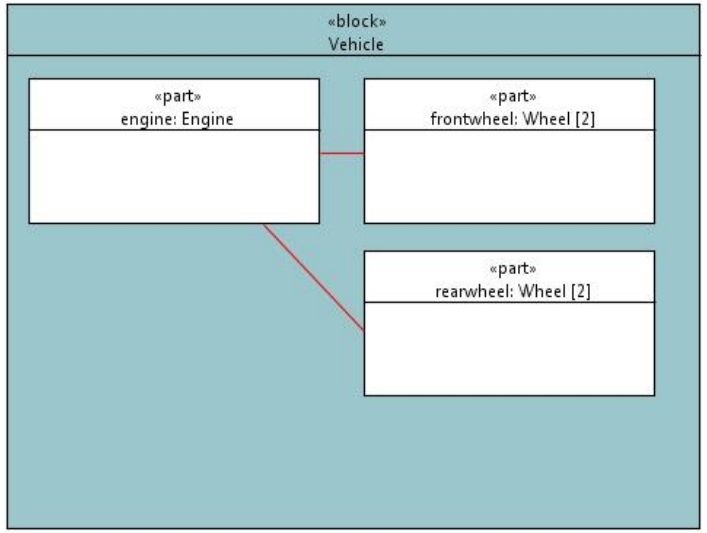


Internal Block Diagram (IBD)

- Describe internal structure (architecture) of a block
- Quite similar to composite structures in standard UML
- BDD: defined block properties (part, references, etc)



BDD ⟷ IBD



(1) different view, in which properties are represented as squares inside the Block

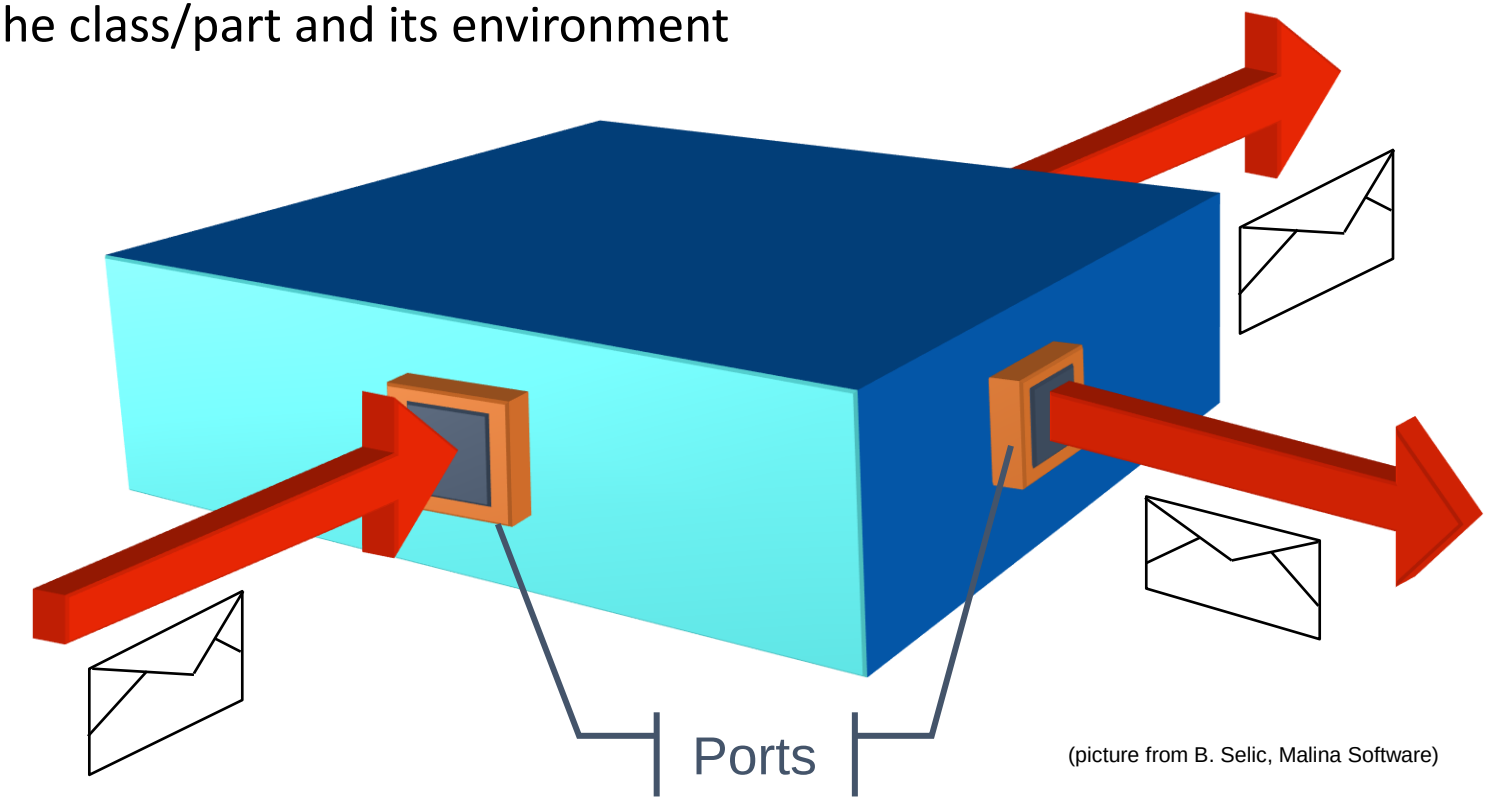
(2) add information about logical or physical wires (connectors) connecting these properties

IDB Ports

- *Services* are provided and required through standard UML Ports.
- Ports specify services the owning block provides (offers) to others and services that the owning block expects (requires) from others
- *Flows* are produced and consumed through SysML Flow Ports.
- A flow port specifies the input and output items that may flow between a block and its environment.
- Flow ports are interaction points through which **data, material, or energy can enter or leave the owning block**.
- The specification of what can flow is achieved by typing the flow port with a specification of things that flow.

Ports

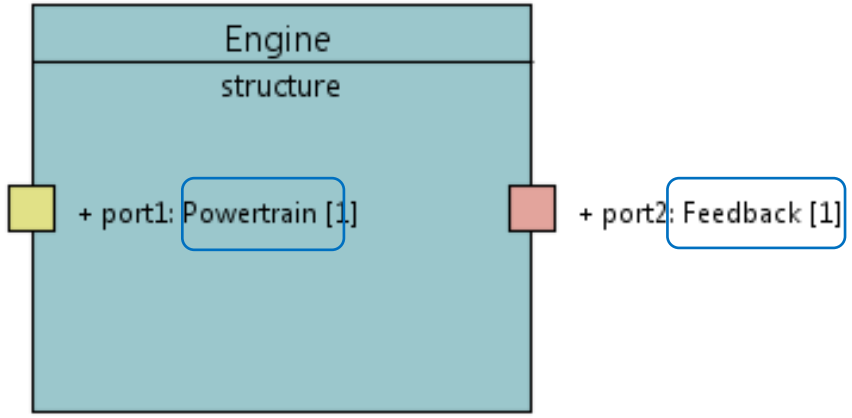
- Ports are interaction points
  - Between the class/part and its internal structure
  - Between the class/part and its environment

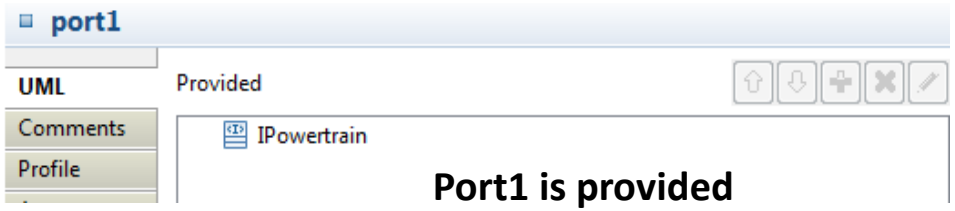


(picture from B. Selic, Malina Software)

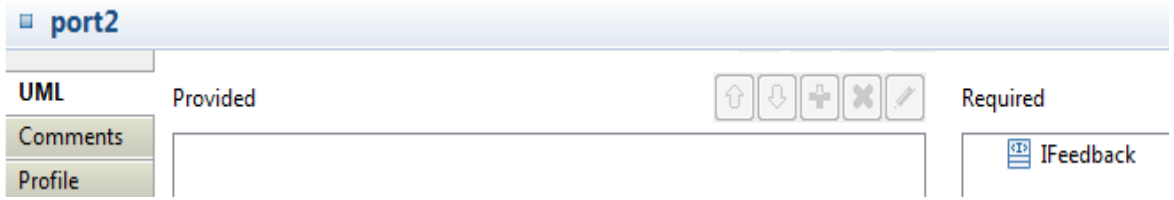
Required/Provided Services (Standard Ports)

The Engine block has two ports typed by Powertrain and Feedback.



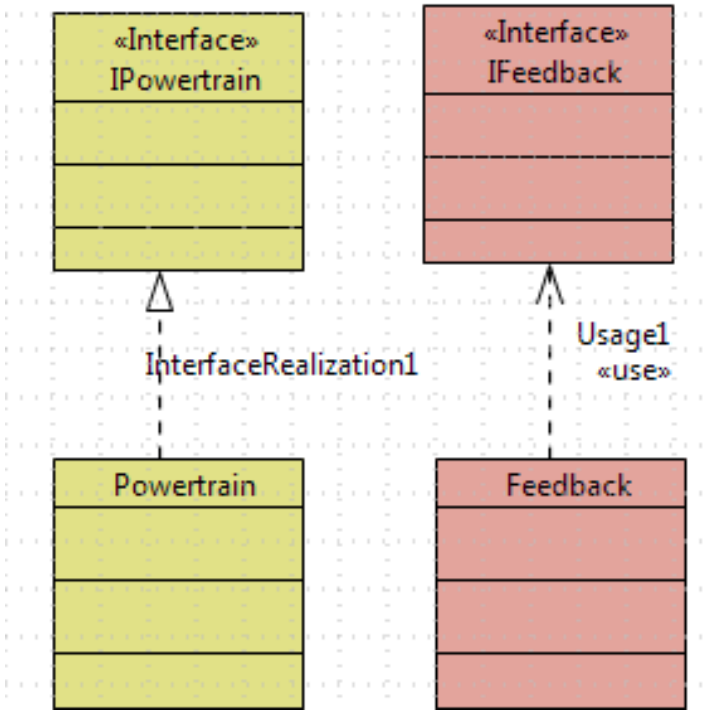


Port1 is provided



Port2 is required

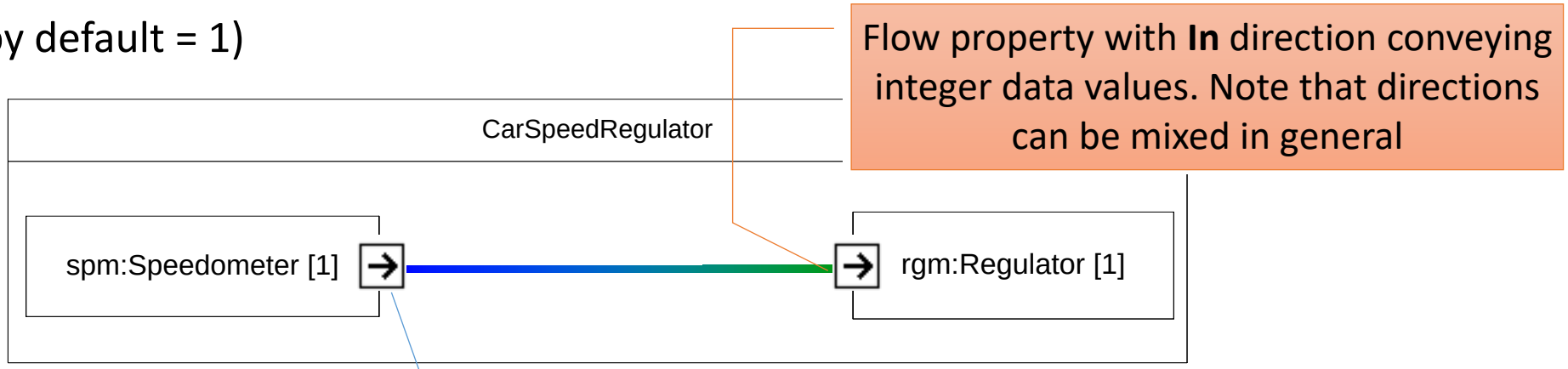
Powertrain realizes the IPowertrain interface
Feedback uses the IFeedback interface



As in standard UML

Full Ports

- Part of the system, typed with a block (port itself can interact)
- The block has set of **flow-properties** (stereotyped attribute)
- Each flow property has a
  - Name and type (since being a UML attribute)
  - Direction = {IN, OUT, INOUT}
  - Multiplicity (by default = 1)



full ports

Full Port vs. Proxy Port

- Act as **proxy** for owning blocks or its internal parts
- Always typed with an **interface block**
- Interface block exposes features of owning block

Connector

- **Connector** specifies links that enables communication between two or more ports or parts.
- **Connected pPorts must have a compliant definition**: opposite direction on each side (or inout on both sides)
- **Connectors are owned by the block (Fb)**. The connector **can cross** a part boundary if the encapsulating block (F3) is not a black box (isEncapsulated attribute of Block equal to false)