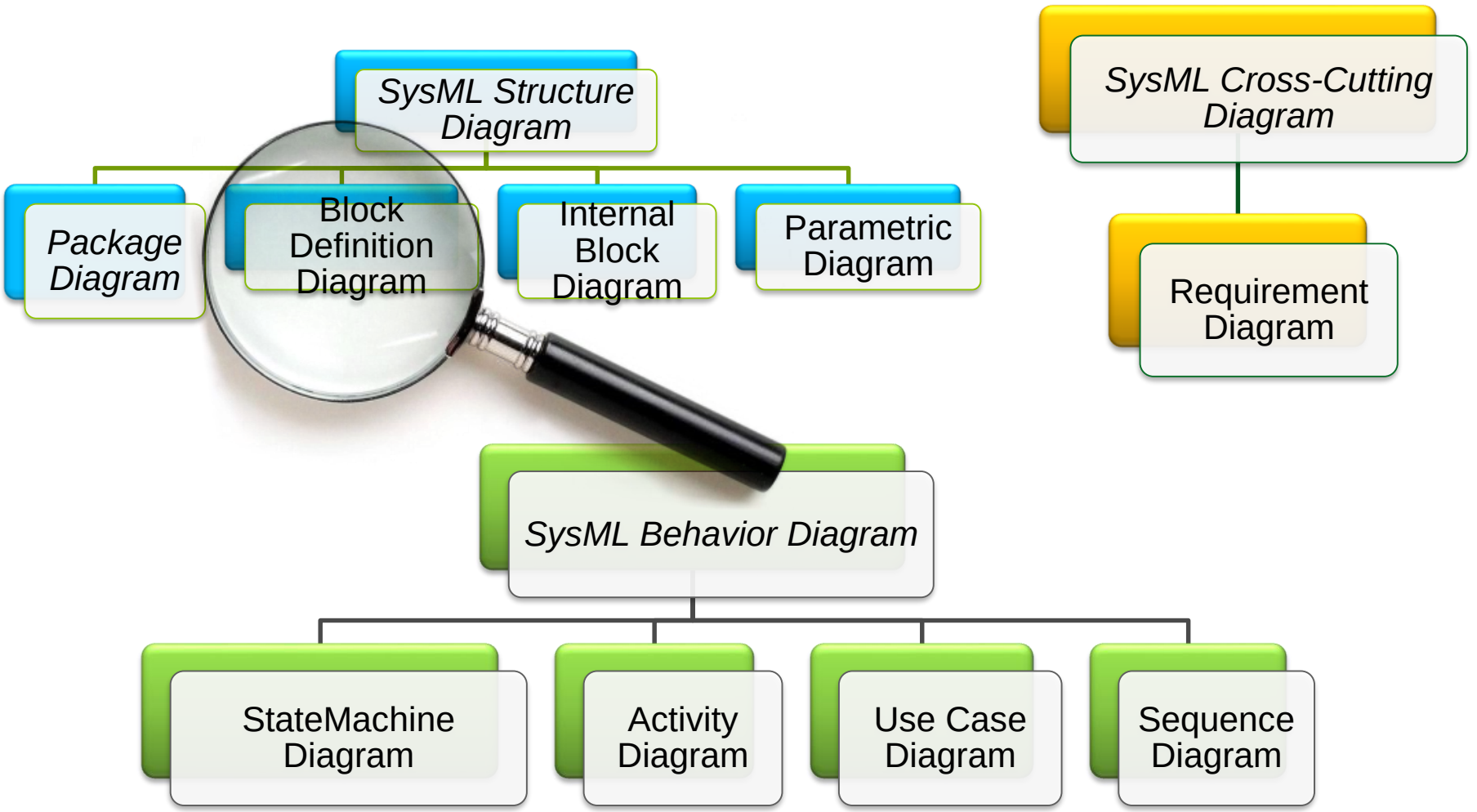
DE LA RECHERCHE À L'INDUSTRIE

STRUCTURAL ARCHITECTURE MODELING WITH SYSML

Software and system engineering department (DILS)
Laboratory of model driven engineering for embedded systems (LISE)

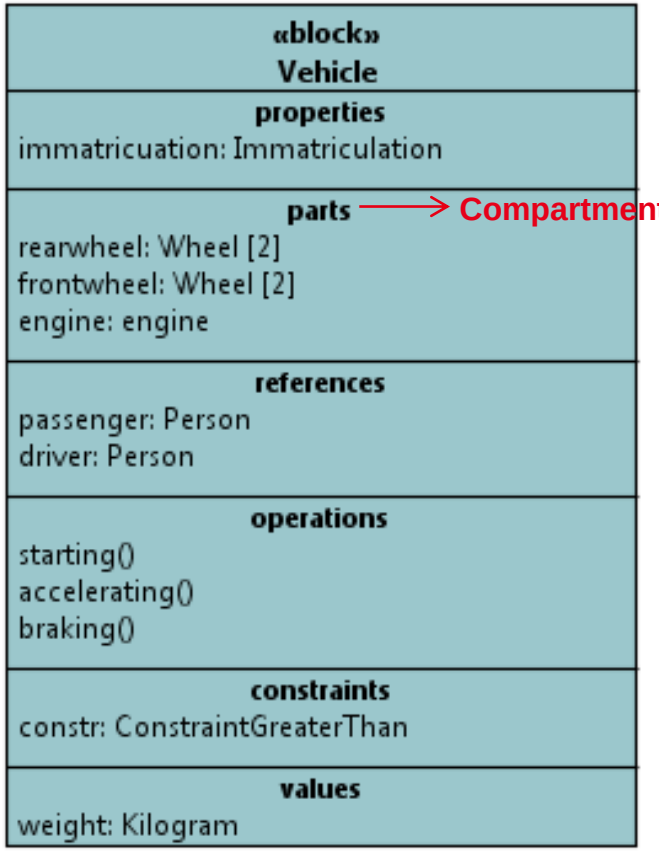
www.cea.fr

BLOCK DEFINITION DIAGRAM



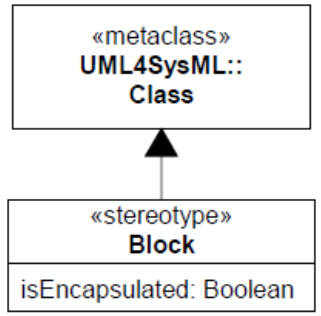
Block Definition Diagram (BDD)

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



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 compartments (see figure). The only obligatory compartment is the compartment for the block name.

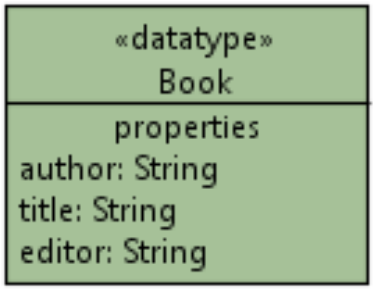


PrimitiveType

- No properties / no operations

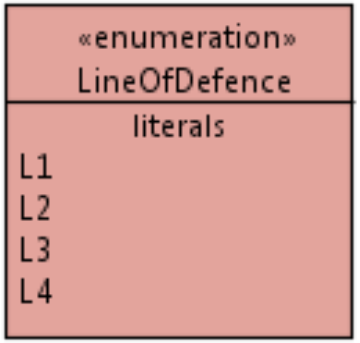
DataType

- Structured type
- Properties
- May have operations



Enumeration

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

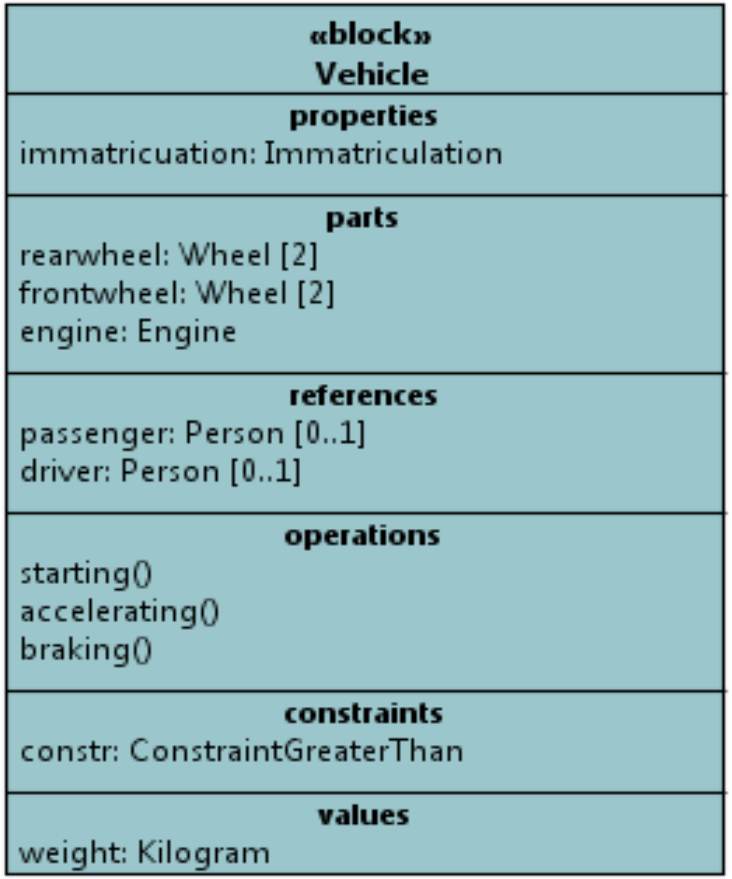


Block extends Class so it has…

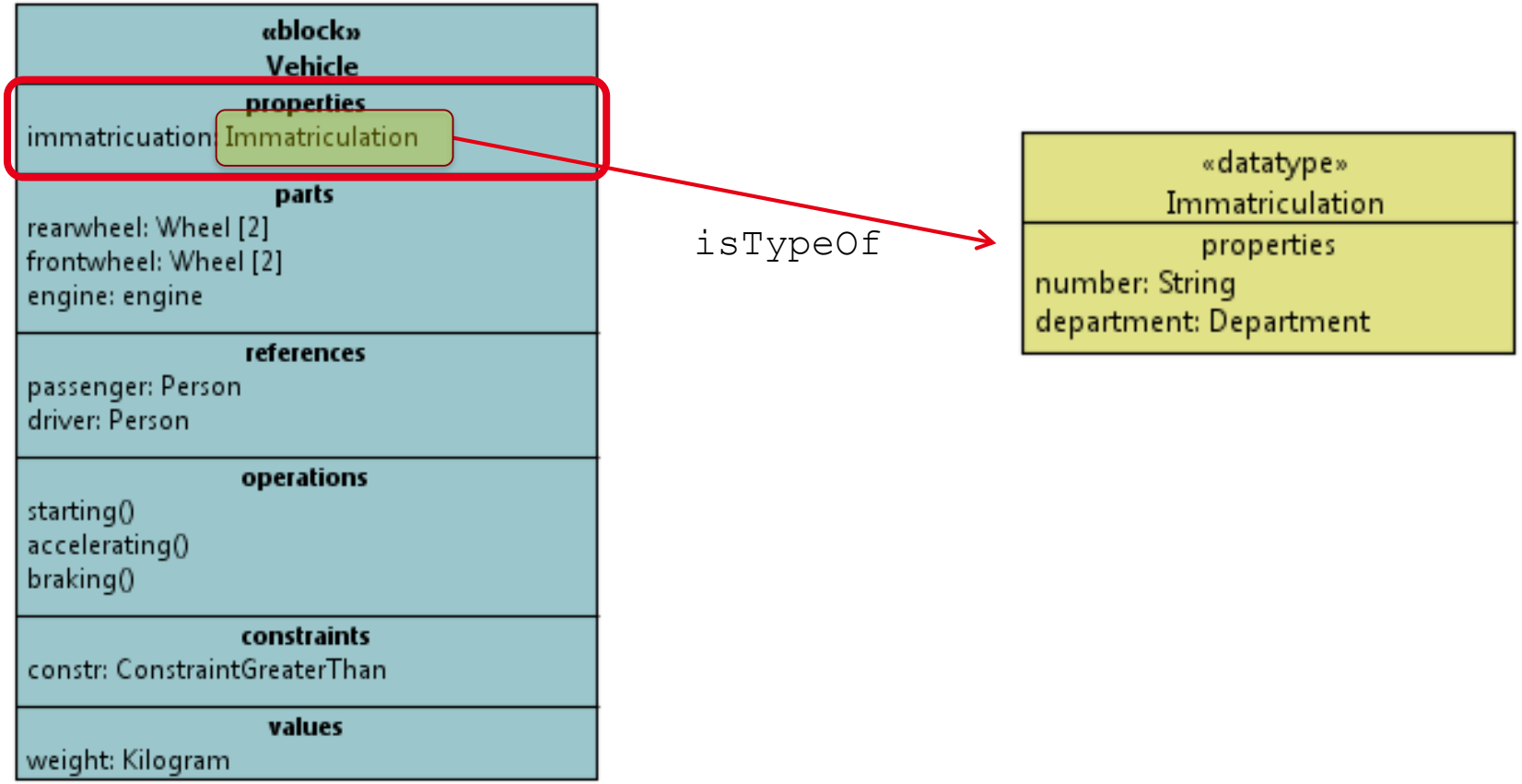
- Properties
- Operations

Block specializes properties

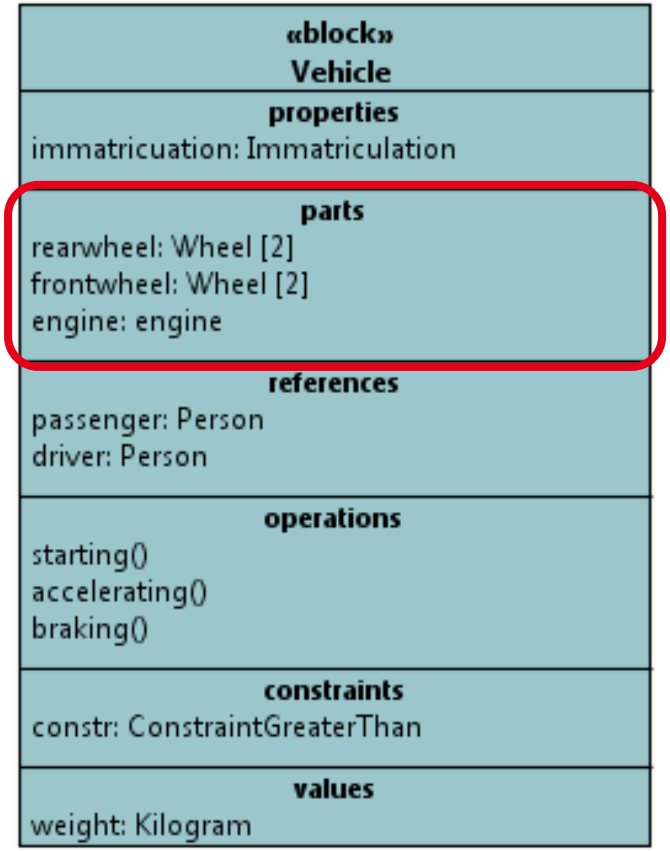
- Part properties
- Reference properties
- Constraint properties
- Values properties



Simple properties: always typed properties



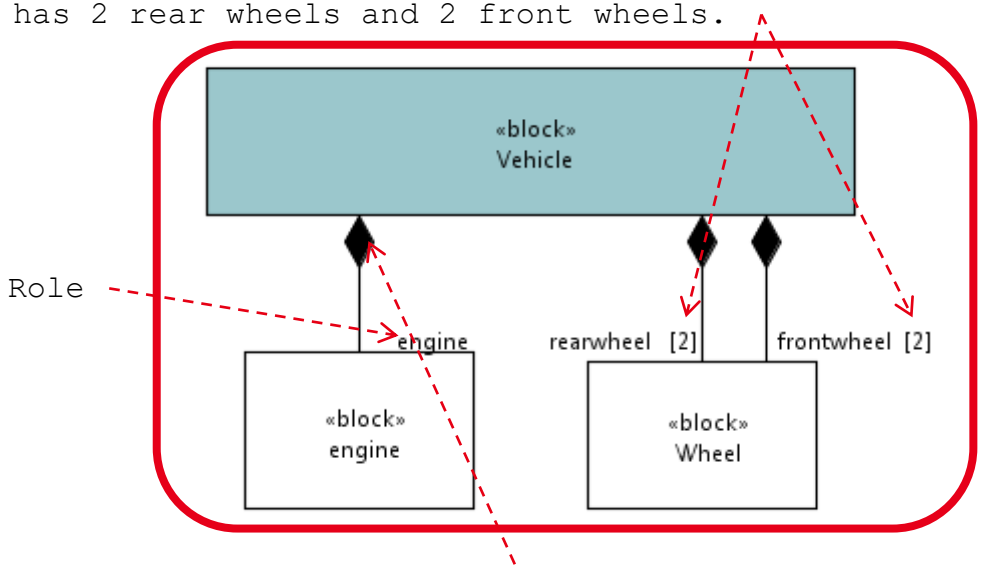
Part properties: describe composition



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

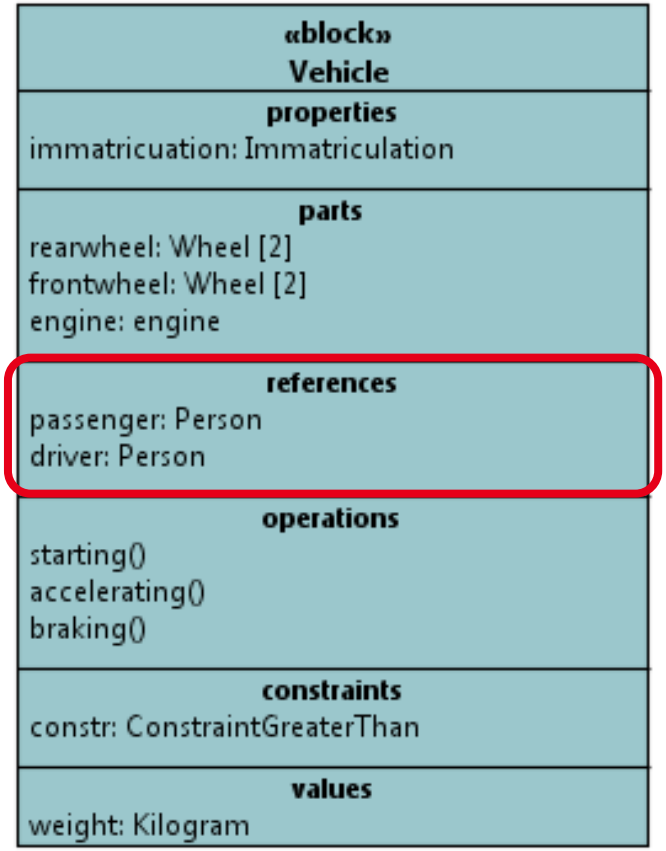
- Specifies a multiplicity
- Specifies a role

Multipilicity 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.

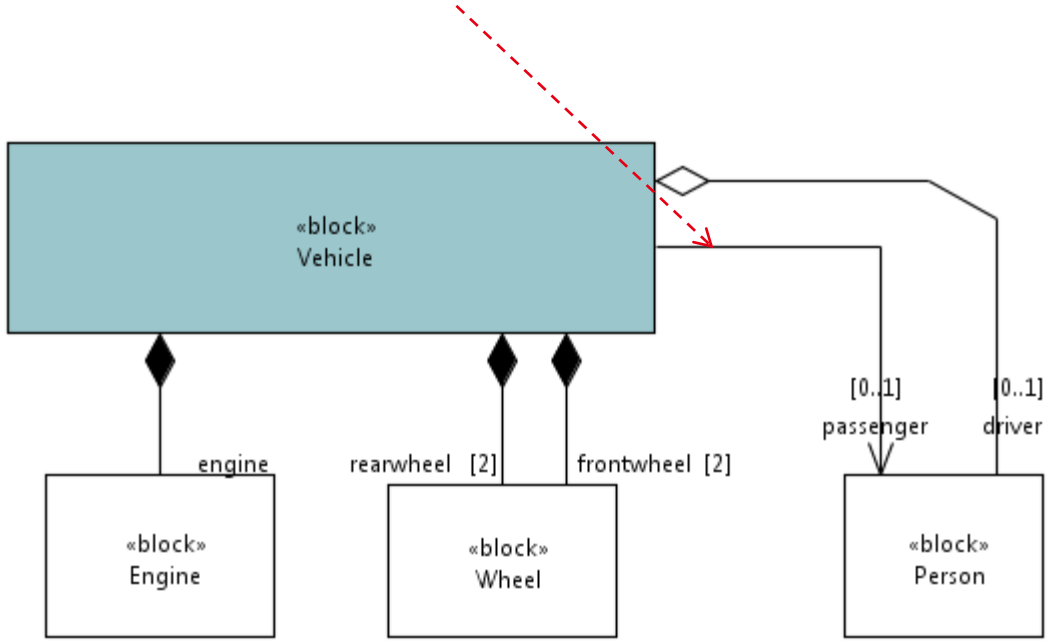


Filled diamond: if an instance of Vehicle is destroyed, the contained instances of Engine is also destroyed.

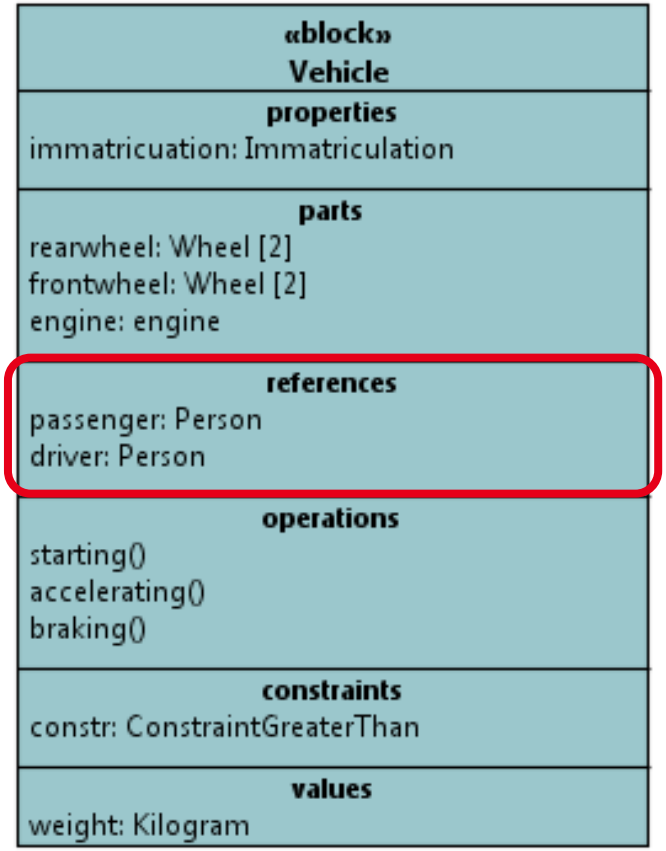
Part properties: describe associations and simple aggregations



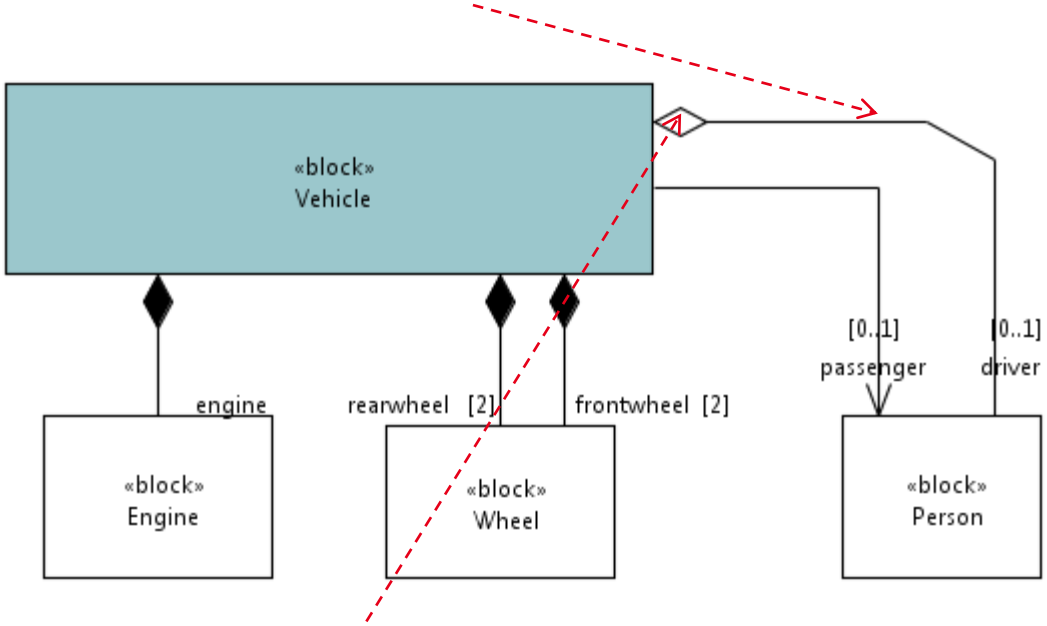
Association: simple reference (no containement relationship between blocks)



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

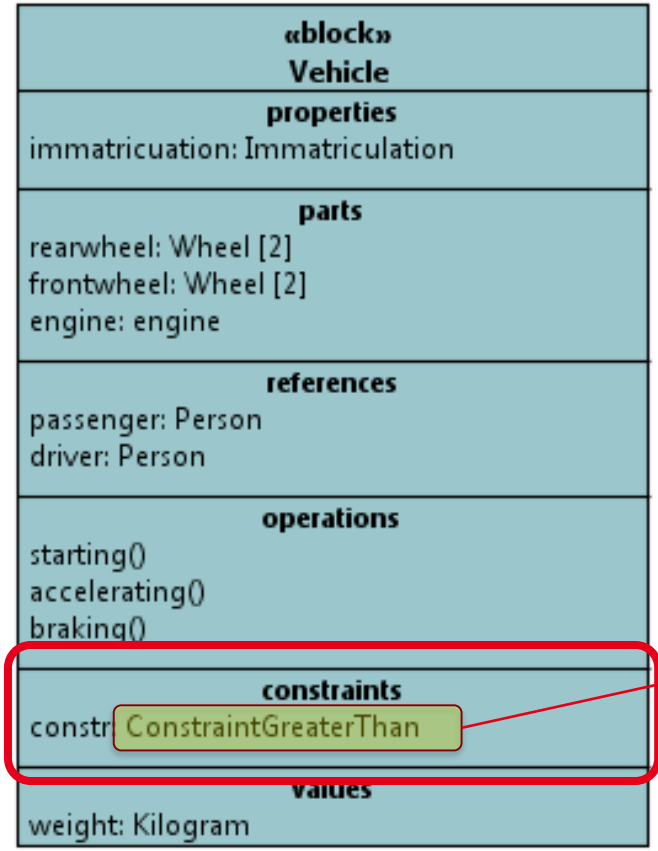


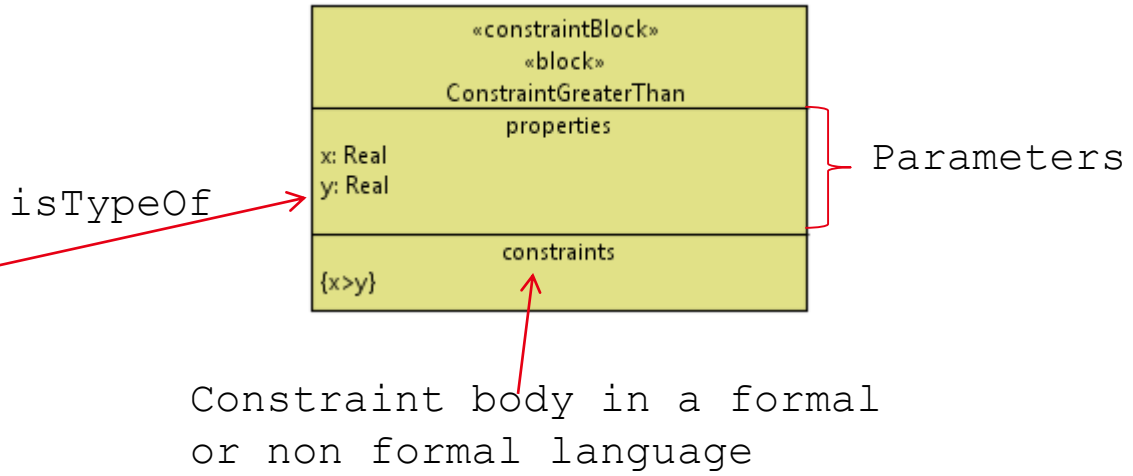
Empty diamond: if an instance of Vehicle is destroyed, the contained instances of Person are not destroyed.

(Hopefully ☺)

Constraint properties: typed by constraint block

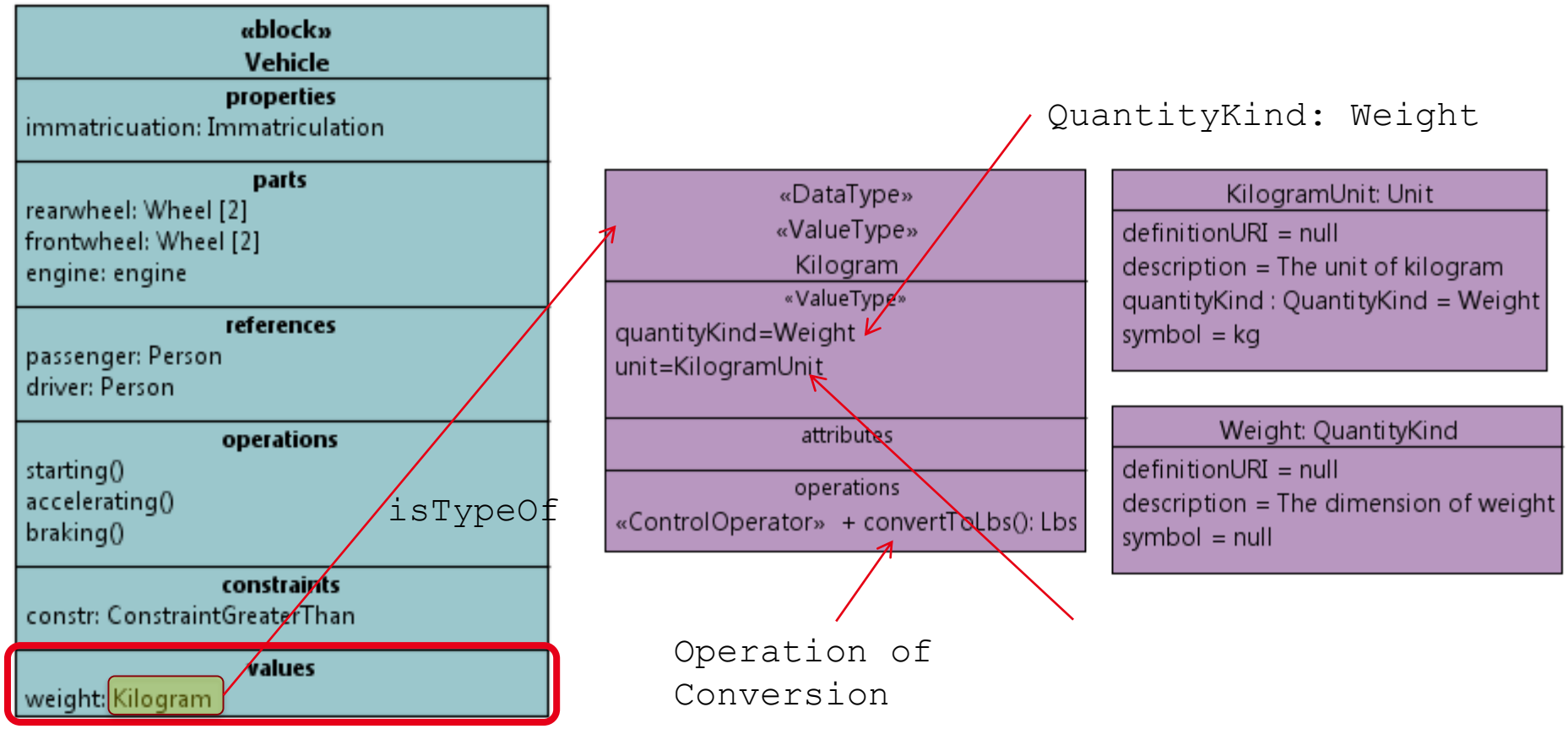
Constraint block: can be used to specify a network of constraints that represent mathematical expressions which constrain the physical properties of a system
They define generic forms of constraints that can be used in multiple contexts.





Constraint properties: typed by a ValueType

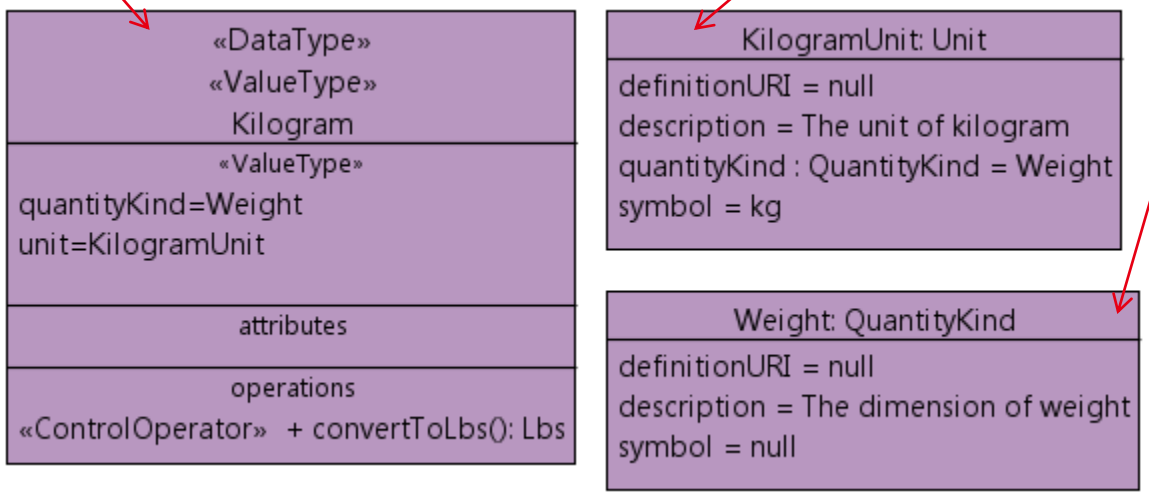
ValueType: describes the type of values; may have an associated Unit and Dimension



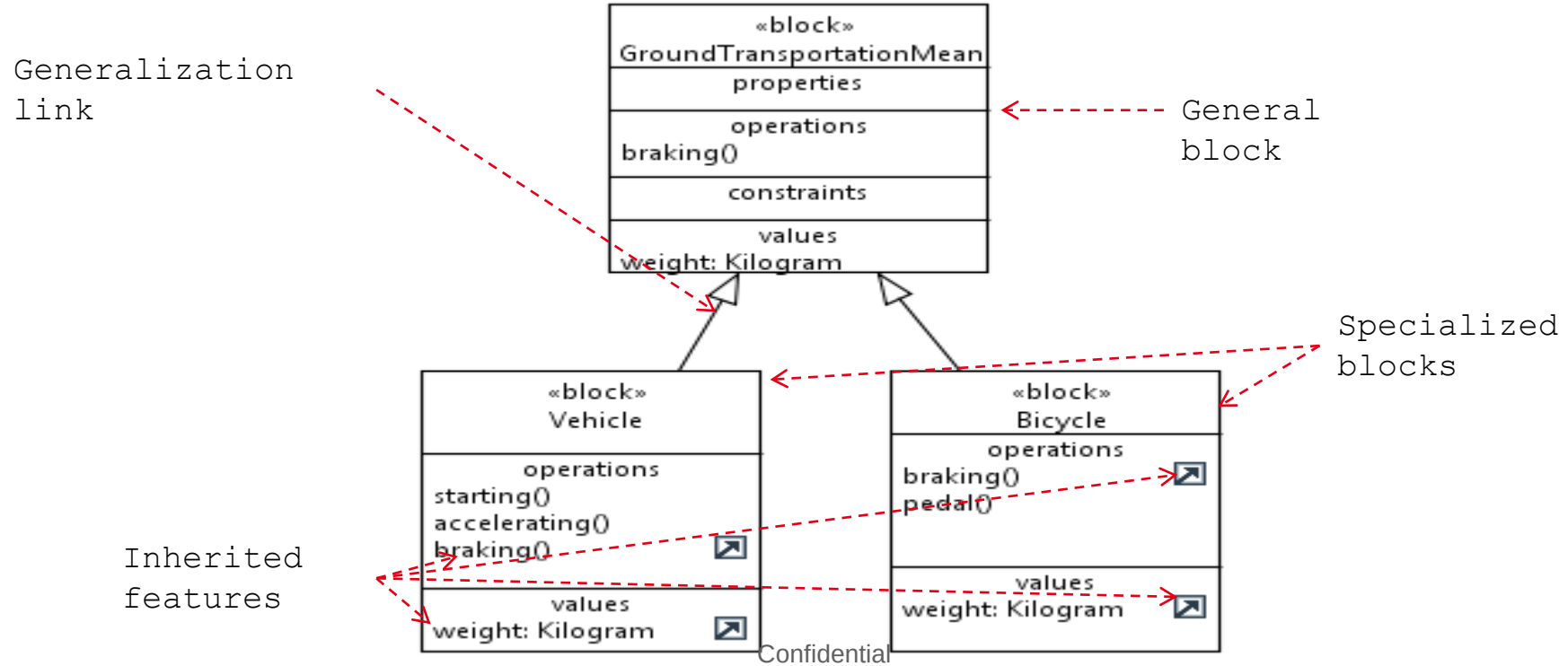
- **QuantityKind:** Kind of quantity that may be stated by means of defined units. For example, the quantity kind ‘length” may be measured by units of meters, kilometers, or feet.
- **Unit:** Quantity in terms of which the magnitudes of other quantities that have the same dimension can be stated.

<<ValueType>> applied on a DataType

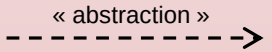
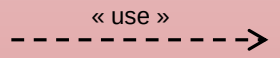
InstanceSpecifications of blocks “Unit” and “QuantityKind” defined in SysML model libraries.



- Like a Class a Block may be a specialization of other Block(s)
- Specialized Blocks inherit features (properties, operations) of general Blocks and adds its own features
- Each instance of the specific block is also an instance of the general block
- Mean 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.	

Your turn

- Design the architecture of the system.
- The objective is to identify the different parts of the systems and to describe how they are related to each other.

Do it in Papyrus

- Create a package “Architecture” at the root of the model and create a BDD in this package.
- The root of the architecture is “RoverSystem” and it contains two parts: Rover and Remote.
- Further refine this architecture.