# CDM-RST Wiki Documentation

## Contents



# 1 CDM-RST compiled wiki pages

*Self-contained version with local images*

## 1.1 Version

This document was generated on 2025-11-20 14:32:36 UTC

---

# 2 Rolling stock consist

## 2.1 Purpose

Represent rolling stock vehicles and formations, their parthood relationships, and the order and orientation of vehicles in a formation.

## 2.2 Vehicles and formations

### 2.2.1 Purpose

Simple distinction between a vehicle and a formation. A formation consists of vehicles or other formations. A formation that cannot be broken down further is called a "vehicle". The old debate is whether the breakdown can, or cannot be performed in operations. "It depends", not necessarily on the rolling stock, so the rolling stock ontology is neutral about it.

The main statement, under the ontology, is: if a piece of rolling stock is either a vehicle (exclusive or) a formation. If it is a vehicle, it cannot be broken down into other pieces of rolling stock.

A TGV trainset or ICE3 for instance have trailers, but these will not be handled separately in operations (ICE case) and cannot even rest on rails (TGV trailers rest on 0 to 2 bogies), but are still considered "vehicles" by many, and for sure they cannot be further broken down into smaller pieces of rolling stock - only parts (carbodies, bogies, HVAC units, seats…).

### 2.2.2 Diagram

The diagram illustrates a composite pattern (known from UML) which expresses that a formation may contain vehicles or other formations.

The flattened black hexagon is the GRAPHOL symbol for disjoint unions: hence the set of all rolling stock contains two disjoint subsets, vehicles and formations. A vehicle cannot be a formation, and conversely. A formation X can however contain a single vehicle Y, but X and Y remain distinct objects.

Property "part of formation" is functional (and it inverse "includes Rolling stock" is inverse functional): this means that a piece of rolling stock cannot be part of two formations. For reference, an OWL "functional property" has at most one object.

### 2.2.3 Comments

**2.2.3.1 More details about the representation of properties in GRAPHOL** The GRAPHOL diagram (and the OWL2 ontology) also expresses that a formation includes *at least* one piece of rolling stock. The double arrow (equivalence) between class Formation and the small white box means exactly that. In detail:

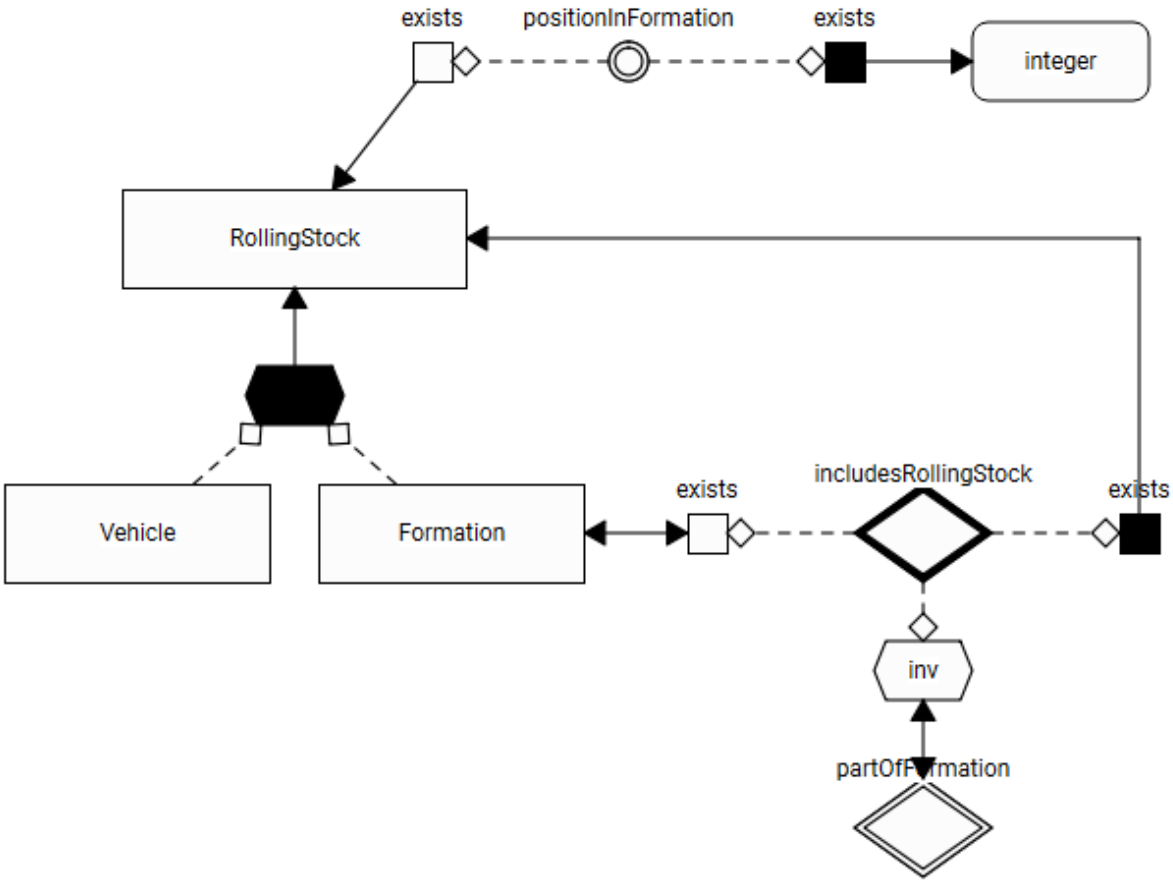


Figure 1: GRAPHOL diagram, formations

- property "includes Rolling stock" has an extension, which is the set of pairs (a formation, a rolling stock) for which the property holds.
- the white box represents the set of formations for which the property holds
- the black box represents the set of rolling stock for which the property holds
- the double arrow means mutual inclusion, i.e. equivalence, so the extension of formation is equivalent to the set of formations that includes some (existential quantifier = at least one) rolling stock.
- this double arrow entails "every formation includes at least one rolling stock".
- please note that the RDFS domain semantics are different: `cons:includesRollingStock rdfs:domain cons:Formation` means:
  - "a formation *may include* rolling stock"
  - the corresponding GRAPHOL representation would show a single arrow pointing from the white box into the Formation box (Formation is a super-set of the things that include rolling stock).

**2.2.3.2 About handling of constraints** We generally use OWL2 constraints when they also carry some semantics. Here, having a container (formation) without contents (vehicles or other formations) is pretty much meaningless.

Constraints only dealing with data correctness are better represented as SHACL shapes. These can be checked, but do not interact with the semantics of the data representation.

In some cases, we move to SHACL such constraints that are semantically relevant but do not match the self-imposed limitations on OWL2 usage. For instance, an ETCS Balise Group consists of at most 7 balises, but the qualified cardinality restriction "at most 7" cannot be expressed in the OWL2 RL profile; the OWL2 DL profile at least is required.

## 2.3 Extremities and sides

### 2.3.1 Purpose

All rolling stock is oriented on the drawing board. Extremities are usually designated "extremity 1", "extremity 2", and this distinction helps differentiating driving cabs on a locomotive, the placement of axles or loads, etc. independently from other information (e.g. running direction). Rolling stock orientation is conventional and may rest on standards: see for instance EN 13775-1.

Once the extremities are defined, the sides can be defined too. We consider any rolling stock to have an elongated box shape with six outer surfaces (sides): front, rear, left, right, top, bottom. Again, the sides refer to the intrinsic orientation of the rolling stock, not to its travel direction.

### 2.3.2 Diagram

The diagram illustrates two properties: hasExtremity ("a rolling stock has two extremities") and facesExtremity (on coupling, an extremity of a rolling stock faces the extremity of another rolling stock).

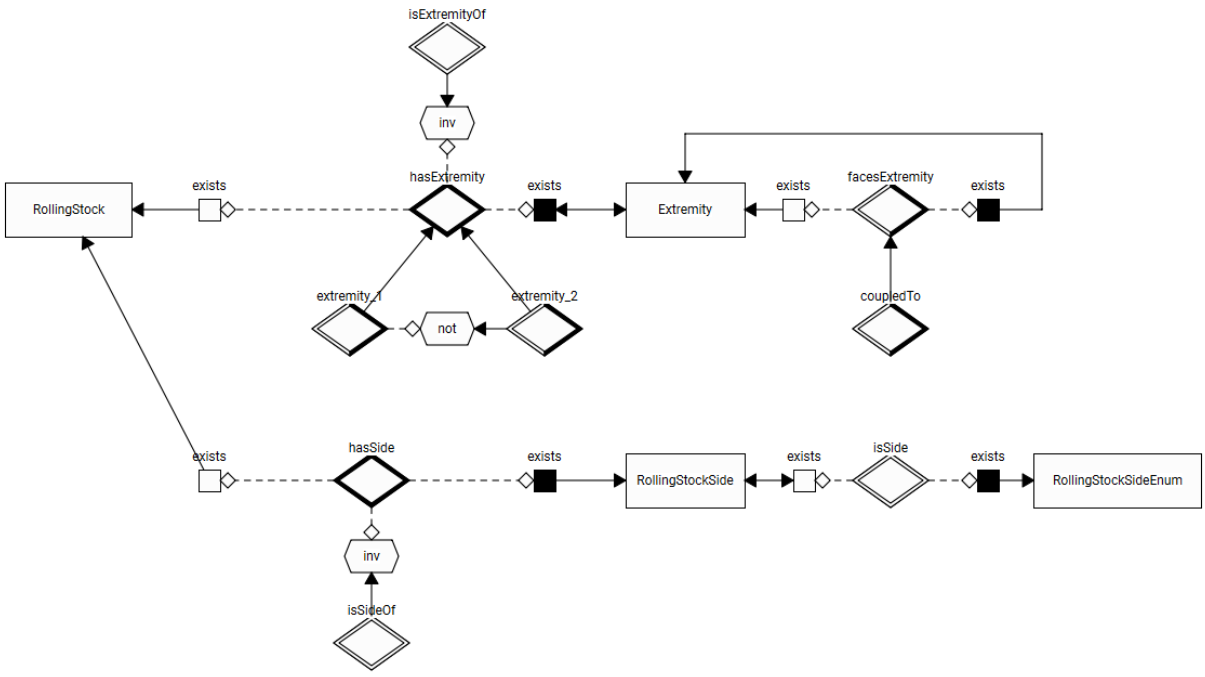


Figure 2: diagram: extremities

An extremity belongs to one wagon (it cannot be shared with another one), so "hasExtremity" is inverse functional.

Extremities of two adjacent wagons may face each other: this is expressed by "faces extremity", a symmetric property (if X1 faces Y2, then Y2 faces X1).

Once extremities are facing each other, one may consider coupling them; if they are coupled, they are obviously facing each other. This is exactly the meaning of "coupled to" being a subproperty of "faces extremity".

The sides are "objects" in their own right, and characterized by "Left", "Right", etc. in a ternary relationship", e.g. "This particular side is a side of (*isSideOf property*) that particular rolling stock, and more precisely is it (isSide property) a Left (a skos concept) side".

### 2.3.3 Comments

**2.3.3.1 About cardinalities, subproperties, and disjoint properties** Property hasExtremity should have a cardinality of exactly 2 (not available in OWL2 RL) and this cardinality is of little use anyway, since we need to identify extremity 1 and extremity 2 that will orient the rolling

stock. These extremities are singled out by sub-properties of hasExtremity. These sub-properties are disjoint: by essence, extremity 2 of a wagon cannot be the same as extremity 1 of the same wagon. This is expressed by the "not" flattened hexagon in GRAPHOL, that tells that the set of (rolling stock, extremity) pairs satisfying "extremity_2" does not overlap the set of pairs satisfying "extremity_1". In set theory, Y does not overlap X iff Y is a subset of the complement of X (as the complement of X is the largest set not overlapping X).

**2.3.3.2 About the representation of ternary relations** In the same diagram, two different ways were chosen. This apparent inconsistency deserves some explanations.

In the case of sides:

- a rolling stock has six sides but the diagram would allow any number. One may consider a SHACL shape to forbid more than 6 sides, which would reveal a wrong representation of the rolling stock. If there is no interest in rolling stock sides, the cardinality could be zero, no harm done: this is why there should be no minimum cardinality in that case.
- what a side "is" (a left side, etc.) is however indispensable for a "side" to make sense, so property "is side" must have exactly one value. This qualified cardinality is semantically relevant, and is expressed here in the usual OWL2 RL-compliant way: a functional property (hence max cardinality = 1) and an existential restriction (class RollingStockSide must have at least one value on property "is side").
- however, the diagram allows two or more sides to be defined as e.g. "left side". Again, this would be a mistaken use, to be caught by either SHACL shapes or SWRL rules, because OWL2 does not allow to express restrictions involving two distinct properties.

In the case of extremities:

- extremities have no "attributes" or subclasses that would allow to distinguish them. Instead, there are two properties that are disjoint (mutually exclusive), namely extremity_1 and extremity_2. What is expressed very *concisely* here is that a rolling stock has two distinct extremities 1 and 2; they cannot point to the same individual.
- here, any mistake would be caught by nay OWL2 reasoner.
- **we did not use that strong semantic guardrail in the case of sides** although we could have used it. A matter of taste: introducing 6 subproperties and 15 pairwise disjoints look rather confusing to the human reader.

Designers may be ;ore or less concerned about conciseness according to the intended usage of the ontologies. Here (aiming at TRL 8), the argument for conciseness might still be heard. And in case of higher TRLs, precautions will rather be taken with additional SHACL shapes, most likely.

## 2.4 Formations as ordered sets

### 2.4.1 Purpose

Sequences, vectors, arrays are not part of OWL2 syntax elements. To express order, one must use some List ontology expressed in OWL2. Here, we use the same List ontology also used by IfcOwl. It rests on the "linked list" paradigm, with each element of the list pointing to the next.

The list is invariably closed by an instance of EmptyList, a special class provided by the List ontology. This ensures that the list has actually come to an end.

*Note: given the underlying "Open World Assumption", we do not assume that data are complete, we only accept* ***positive information telling that it is complete****. Here, the fact that a list element does not point to any next element has no meaning. After all, the next element could simply be unknown and get discovered later. But if the list element points to an "empty list", this positively tells that the list has come to an end.*

### 2.4.2 Diagram

The first diagram showed that a formation includes some (= at least one) rolling stock. The diagram below expresses, on the left side, that the included rolling stock can be the "head rolling stock", in which case all other information can be pulled from the head rolling stock that must indicate which successor ("has next rolling stock") it has, until the closing element of the list is reached. The closing element is an individual of class EmptyList, imported from the List ontology; it cannot be confused with a real piece of rolling stock.

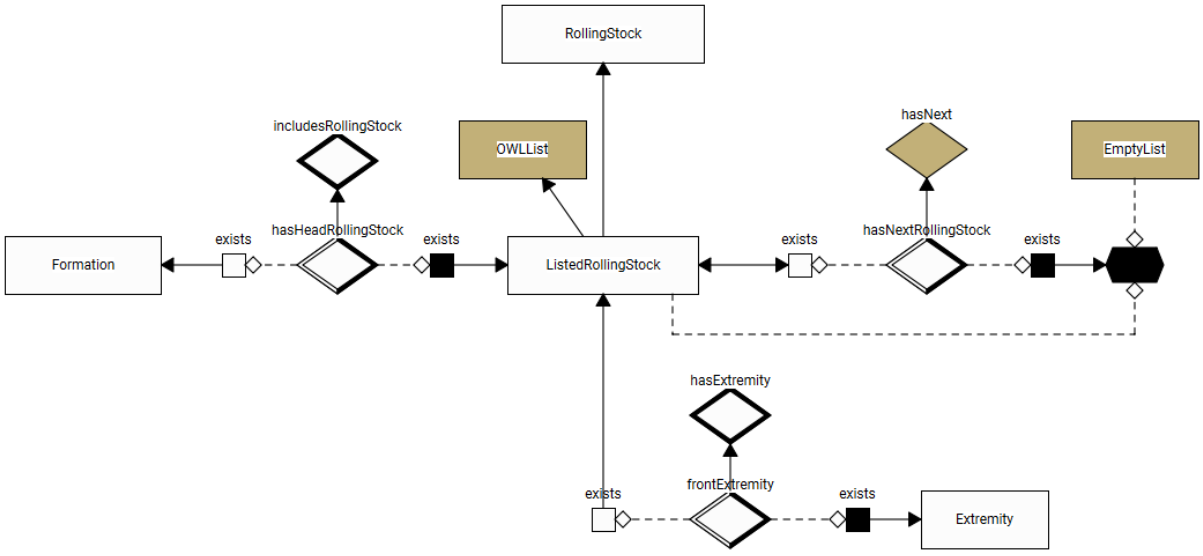


Figure 3: diagram: rolling stock consist

*Reading the GRAPHOL diagram from "Listed rolling stock" towards the right: every listed rolling*

*stock must (double arrow between "listed rolling stock" and the white square) have as next rolling stock at most one (functional property, signalled by double edge) rolling stock xor (disjoint union, signalled by black flattened hexagon) an empty list.*

The bottom of the diagram adds property "front extremity" that designates which extremity is at the front of the rolling stock in the formation.

To know what is at the front of the formation, you can use the property chain (has head rolling stock)$o$(front extremity).

#### 2.4.3 Comments

Rolling stock orientation is secondary to rolling stock composition: you can express the sequence of rolling stock while ignoring orientation. You can also provide partial orientation information (e.g. expressing which cab of the locomotive is in the front, and nothing else). If you provide full orientation information, the relative positions of extremities (properties "faces extremity" and "coupled to") can be derived.

If, on the other hand, your use case is only about the relative position of two vehicles, these do not need to be part of a formation and you can use the extremities in the second diagram.

Original page: 01---Rolling-stock-consist.md

---

## 3 Example: ghost train consist

### 3.1 Purpose

Present the basic mechanism of linked lists in the rst-cons ontology.

### 3.2 Sample file ex1

The "Ghost train" is a formation composed of a locomotive and two rakes of three wagons each.

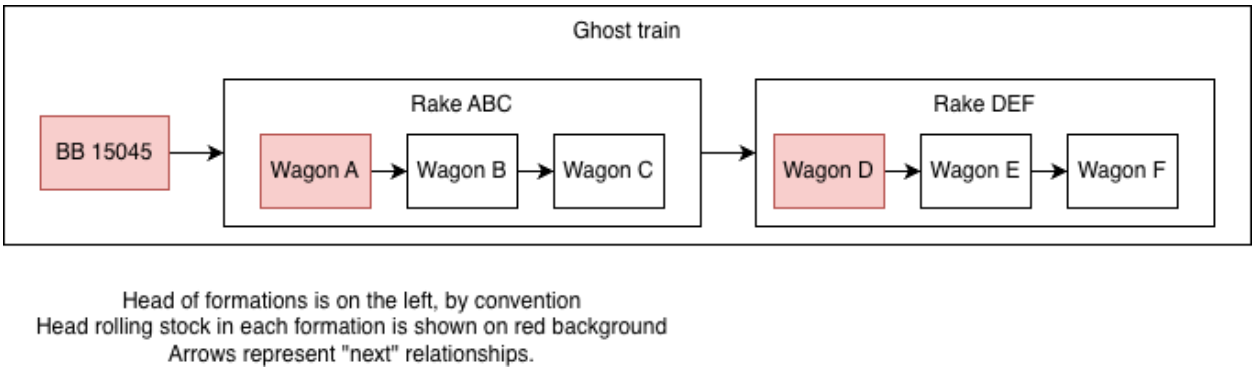


Figure 4: illustration: ghost train

The corresponding RDF Rurtle is ex1.ttl and can be found under ontology/data.

Original page: 02---Example:-ghost-train-consist.md