CDM-Telematics Wiki Documentation

Contents

1 CDM-Telematics compiled wiki pages

Self-contained version with local images

1.1 Version

This document was generated on 2025-11-07 16:09:50 UTC

Contents
Operational entities ## Purpose

Contents

This diagram contains the main classes used for describing operations. Their scope is not necessarily restricted to the TAF TSI scope: for instance, containers (assets) and their “freight load”

Contents

Presentation

Contents

The left part (colored classes) shows the DUL (DOLCE+DnS Ultralite) concepts underpinning the model. The rest represents the taxonomy that is specific to the CDM-TAF ontology

Contents

Next to the right of the DUL classes are the classes describing our domain (operations). Their names should sound familiar.

Contents

Then, further right, are the superclasses (grouping our domain classes). They will mostly be ignored by users. Details are provided in subsequent pages.

Contents

About "DOLCE + DnS Ultralite" (DUL)

Contents

DUL is an “upper ontology” defining very general concepts (physical object, role, description…). It is suitable for describing domains mixing assets, processes, and all sorts of

Contents

End users can safely ignore this upper ontology. It is however useful for those who wish to understand, maintain, or extend the CDM-TAF ontology, since it contributes to sepa-

Contents

Comments on the diagram

GRAPHOL conventions

Contents

GRAPHOL is able to represent all OWL2 elements and relationships graphically. The graphs may evoke E/R or UML diagrams, but there are significant differences resulting from the

Contents

Classes and subclasses Square boxes represent classes (sets of individuals).

Contents
The graph A -> B means “A is included in B”. When A and B are classes (which is the case here), this means “A is a subclass of B”.

Contents

Keep in mind that ontology classes are sets, rather than types (as would generally be the case in object-oriented programming). All set operations, such as intersection, union, or complement,

Contents

Disjoint unions

A black, flattened hexagon means “disjoint union of” whatever classes are attached to the hexagon by dashed lines.

Contents

For instance, Train, Wagon, Intermodal Transport Unit, Track, and Facility, are bundled into an anonymous “disjoint union”. This implies that e.g. a Train cannot at the same time be a Wagon, etc.

Contents

This anonymous, disjoint union is a subclass of “Operational artifact” (did you notice the direction of the arrow?). This means that “operational artifact” contains (or includes) the mu-

Contents

Original page: 01---Operational-entities.md

2 Entity details

2.1 Purpose

Operational entities may all share time-independent properties (attributes), which are described here.

2.2 Diagram

For the time being, a user-defined "name" is proposed. It is not unique (one entity may have several names; different entities can have the same name as the "name" is a vernacular, not an identifier.

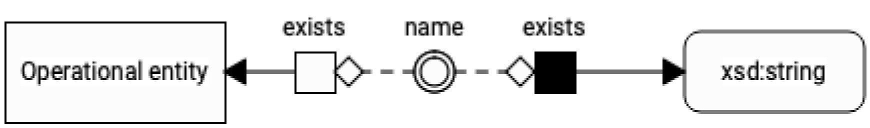


Figure 1: Operational entity details

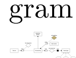
2.3 Comments

The universally unique identifier is expected to be the entity IRI. IRIs are the foundations of the Semantic Web.

Note: one reason for introducing the "name" property is that annotation properties (such as rdfs:label) that would play this role are left out of OWL to JSON transformers (CIM or PIM to PSM).

Original page: 01a---Entity-details.md

Train run

Train servicing ## Purpose ## Diagram  ## Comments Original page: [02a---Train-servicing.md]

---

3 Operational Location

3.1 Purpose

Defining primary and subsidiary location codes. Locating fixed assets (tracks, facilities) at the places designated by these codes.

3.2 Diagram

3.3 Comments

3.3.1 Property "at operational location"

This property only applies to (= has range) spatially fixed assets such as station tracks, facilities (yards, depots, stations, terminals). There is no time information attached to it.

Please note that a yard *has* a location; a yard *is not* a location. This confusion often occurs because a yard happens to be a fixed thing. The confusion never occurs with moving things. In the present

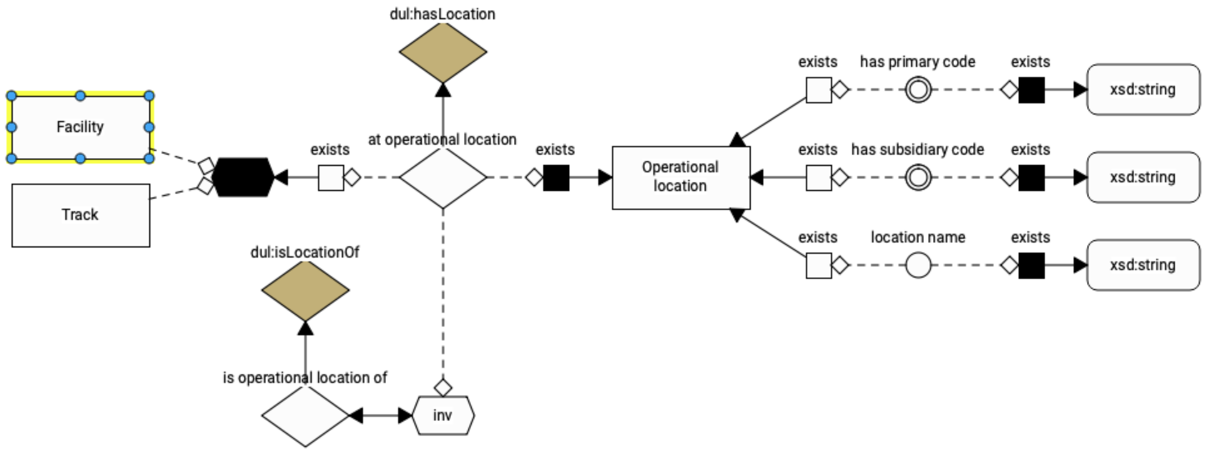


Figure 2: operational location

ontology, fixed and moving things are treated homogeneously, so fixed things are not assimilated to their location.

3.3.2 So where are moving things located at time t ?

The answer lies in the dul:Situation class and its subclasses (generally called "... status" in the present ontology). dul:Situation is the class where time-dependent information is asserted.

Original page: 03---Operational-Location.md

Train
Wagon ## Purpose ## Diagram  ## Comments

---

Train

Original page: 05---Wagon.md

---

4 Intermodal Transport Unit

4.1 Purpose

4.2 Diagram

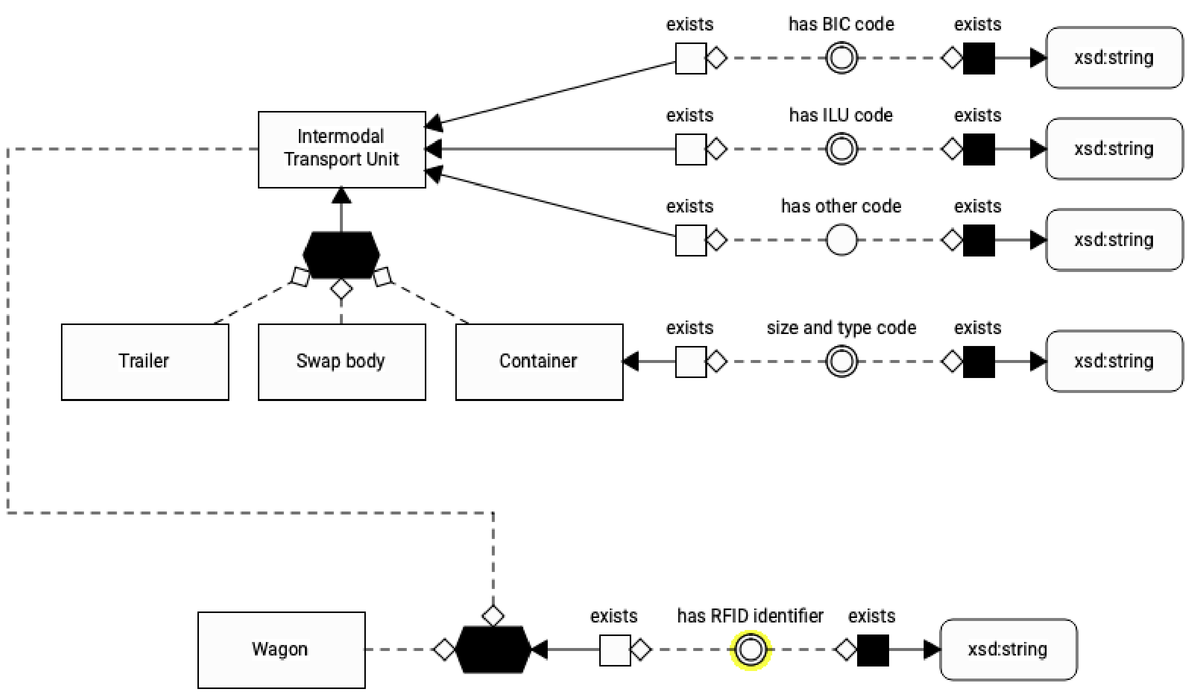


Figure 3: ITU

4.3 Comments

Original page: 06---Intermodal-Transport-Unit.md

Cargo

Track ## Purpose ## Diagram

Comments Original page: 07---Track.md

---

5 Facility

5.1 Purpose

5.2 Diagram

5.3 Comments

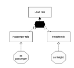
Original page: 08---Facility.md

Traction role

Load Role ## Purpose

Traction role

Diagram



Comments Original page: 10---Load-Role.md

---

6 Operational roles

6.1 Purpose

Companies may play different operational roles at different times. Examples of operational roles are: Train operator, Cargo carrier, Lead RU, Lead carrier (the latter is encountered in the TAF TSI).

6.2 Diagram

Operational roles are currently listed as members of class OperationalRole. In the future, these may be replaced by a SKOS concept scheme provided by ERA.

The answer to "who plays the role" is outside this ontology. Such companies are by definition subclasses of dul:Organization and regorg:RegisteredOrganization.

dul:Organization tells that its members are able to play a role. regorg:Organization provides lots of useful attributes (legal name, jurisdiction, registration authority, etc.). The W3C REGORG ontology is based on the W3C ORG ontology and is recommended for use by the EC (see this ontology collection item and this announcement page from the Interoperable Europe Portal.

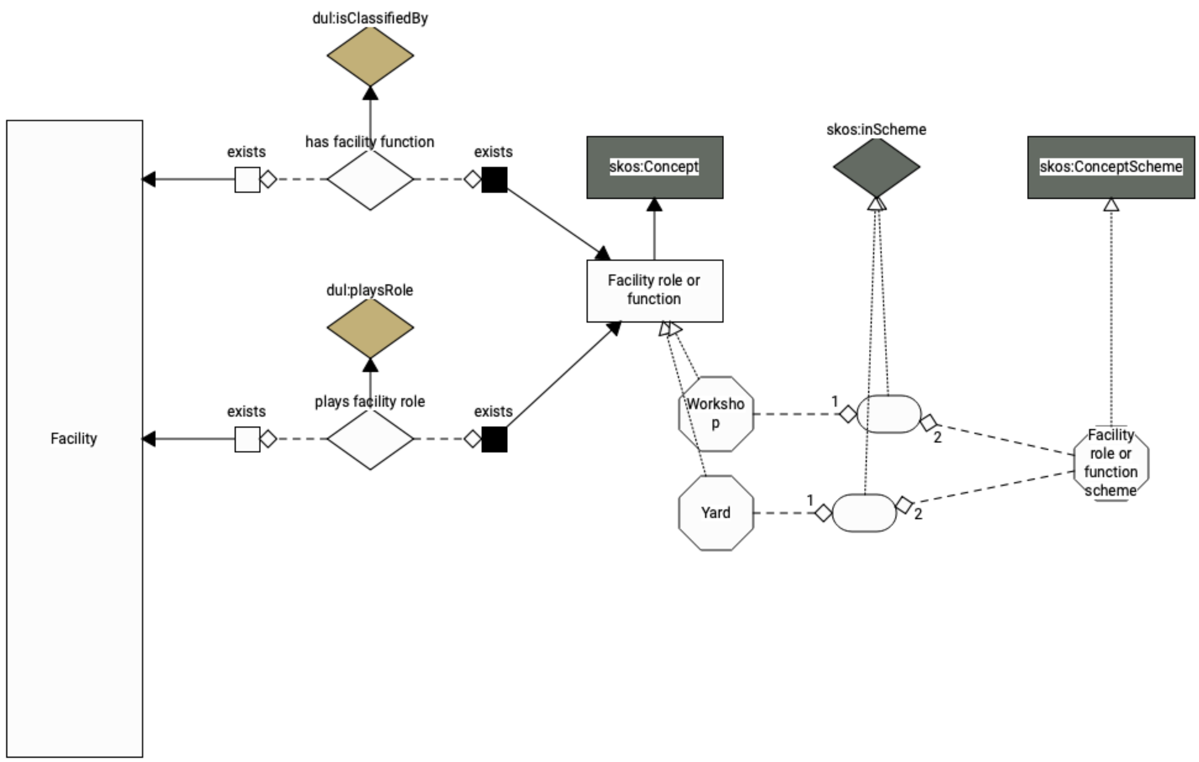


Figure 4: Facility

6.3 Comments

6.3.1 No dedicated class for organizations - just a blank node

We simply use a "blank node" (a class without a name) that is the intersection ("and") of dul:Organization and rerorg:RegisteredOrganization.

What the diagram expresses (and what OWL2 says) is that anything that plays an operational role in the context of telematics is, by definition, both a dul:Organization (hence not a person) and a rerorg:Organization (with some or all of the foreseen properties provided by RERORG).

Original page: 11---Operational-roles.md

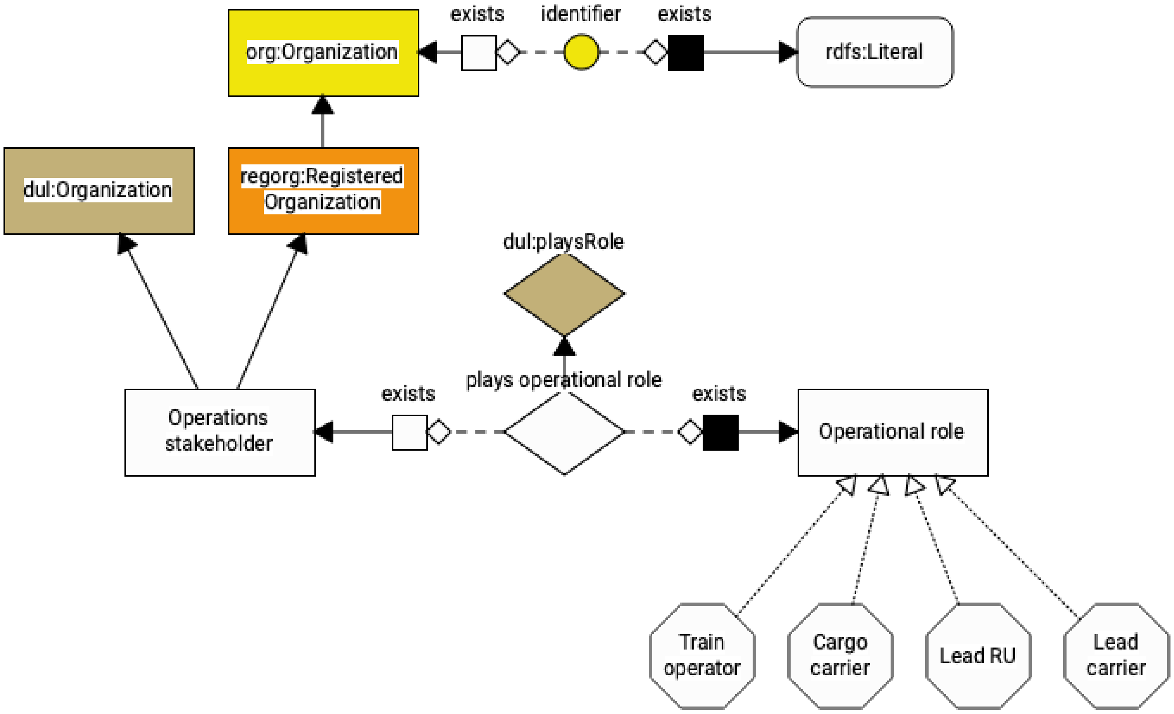
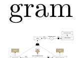


Figure 5: operational roles

Versioned description

Journey ## Purpose ## Diagram



Versioned description

Comments

Original page: 12a---Journey.md

7 Journey Schedule

7.1 Purpose

Represent the train journey (planned or executed) and possibly the train path in one uniform way.

The Journey may be composed of a sequence of journey sections, each having an origin and a destination (operational locations). Obviously, the destination of section N is expected to also be the origin of section N+1, and times must be increasing (except when midnight is passed).

7.2 Diagram

7.3 Comments

7.3.1 Nested Lists

Since any Journey section is a Journey schedule, it can be in turn broken down. This is convenient, as it allows to insert pass-through locations at a later stage when deemed convenient.

7.3.2 Not a speed profile

Locations documented by Journey Schedule are, for the time being, only Operational Locations. The Journey Schedule is not suitable for representing a speed profile with temporary speed restrictions for instance.

7.3.3 Data consistency

It is a user responsibility to check the order and consistency of journey sections. The ontology will preserve the sequence (using a List ontology, because OWL2 has no primitive concepts for order),

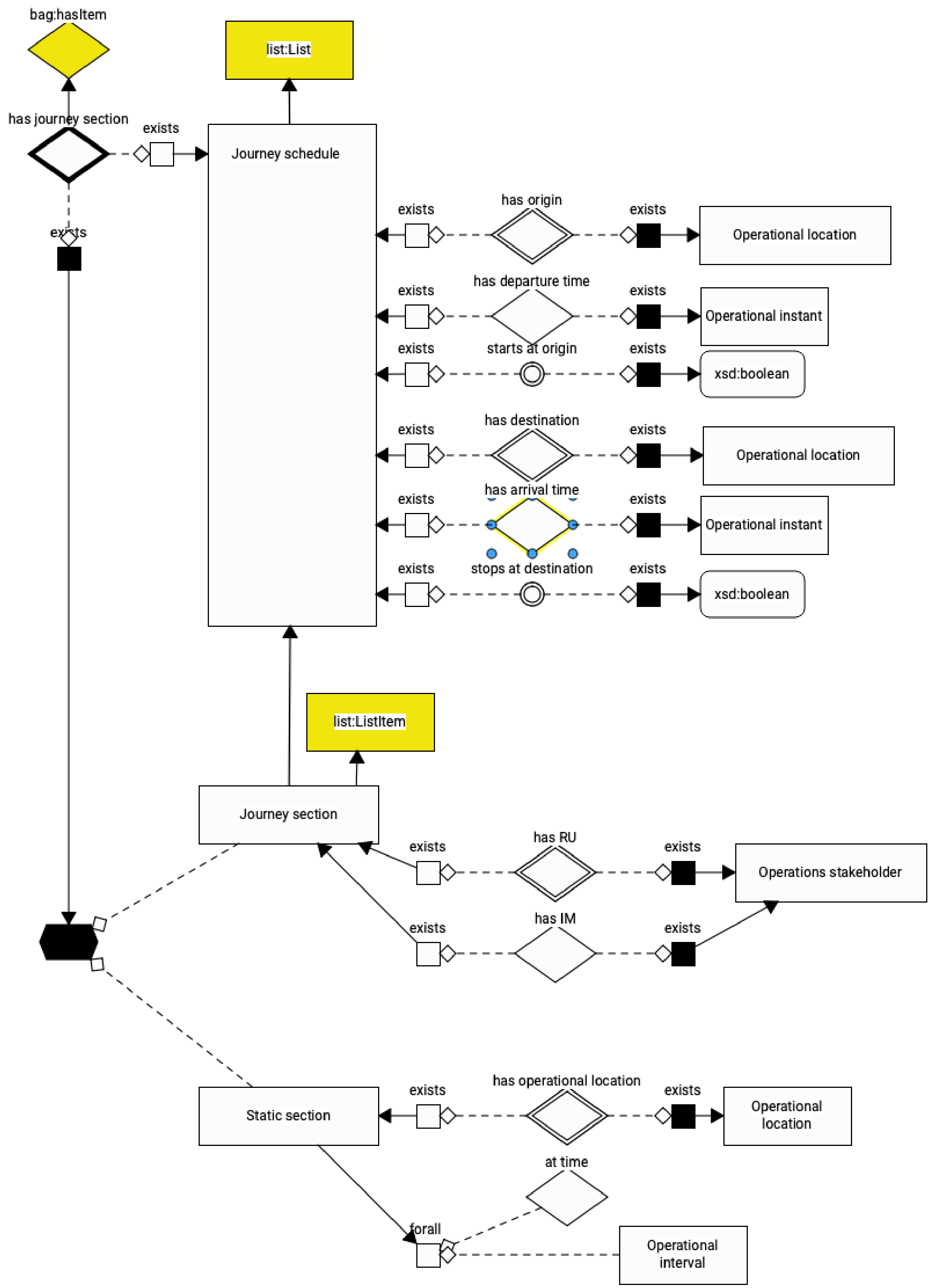


Figure 6: Journey Schedule

but whether the sequence *makes sense* must be checked by other means, such as SWRL rules, or queries (possibly embedded in SHACL), etc.

7.3.4 List ontology

The List ontology used here is different from the one used in IfcOwl (the ontology version of Industry Foundation Classes). Later on, one of the two ontologies may be eliminated in favor of the other.

Original page: 12b---Journey-Schedule.md

Journey Schedule properties
Operational State ## Purpose

Journey Schedule properties
Describe time-dependent attributes (properties) that are relevant to operations: train run state (“where is my train?”), train state (“what is the train composition?”), load state (“on which wagon is my con-

Journey Schedule properties

Diagram "Operational state" is the topmost class, on which the time-dependency hinges. Time can be expressed as an instant or as an interval.

Journey Schedule properties
Time instant suggests that the state is actually a state change, valid until the next, or a spot measurement (“is my train at rest?”).

Journey Schedule properties
Time interval suggests that the state extends over the interval, or maybe the state change.

Comments

Restriction on property dul:hasTimeInterval

Journey Schedule properties
In our context we use the predefined property dul:hasTimeInterval but restrict its range to time:Interval (improper time interval in the W3C Time ontology) when it applies to an Opera-

Journey Schedule properties

Ambiguity to be lifted The ambiguity state / state change should be removed at a later stage. ### Proper vs. degenerate intervals

Journey Schedule properties
All time-related classes are borrowed from the W3C Time ontology.

Journey Schedule properties

Time intervals can be bounded, or open-ended, or degenerate (start instant = end instant). Again, it would be useful to clarify when to use an instant and when to use a de-

Journey Schedule properties

Original page: 13---Operational-State.md

---

8 Train run state

8.1 Purpose

8.2 Diagram

8.3 Comments

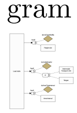
Original page: 13a---Train-run-state.md

Train state

Load State

Purpose

Diagram



Comments

Train state
Original page: 13c---Load-State.md

9 Message

9.1 Purpose

9.2 Diagram

9.3 Comments

Original page: 14---Message.md

Image
RID codes ## Purpose

Image
Add the information about dangerous goods and the resulting hazard classes, as per RID.

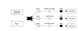
Image
Information is defined in chapter 3.2 of the RID, Table A, columns 1 and 3a, respectively. ## Diagram

Image
The information relates to the dangerous substance, but shall be affixed on the ITU or the wagon, as per chapter 5 of RID.

Image

Accordingly, two properties are provided, the domain of which is the union of Wagon and ITU. This union is a disjoint one, since classes Wagon and ITU were declared disjoint (no wagon is an ITU, no ITU is a

Image



Comments

Original page: 20---RID-codes.md

10 Time

10.1 Purpose

Propose “time entities” to be consumed by time-dependent entities (such as “train speed” or “custom clearance status”).

The time entities are derived from the W3C time ontology and aligned with the DUL TimeInterval concept.

10.2 Diagram

An operational time is either an instant or an interval: these are disjoint classes, as the black flattened hexagon indicates in the diagram. The distinction between instant and interval rests on semantics, not on timestamp values: an interval has a beginning and an end (whether or not the values are known), while an instant has a beginning (instant) and an end (instant). These may happen to coincide, i.e. have equal timestamp values.

The GRAPHOL diagram expresses that each interval is expected to have exactly one beginning and one end, by means of OWL universal restrictions (“forAll”), and that the beginning or end are of type “Operational instant”.

Note: if the data provide two beginnings for an interval, the logical consequence is that the beginnings B1 and B2 are the same individual (:B1 owl:sameAs :B2) and any OWL reasoner will infer that and inform the user accordingly. Then the user software may want to compare the respective timestamp values (OWL2 ignores them, so use SPARQL or code). If timestamps of B1 and B2 fail the test for equality (say, difference is more than one millisecond? which is context-dependent), then the user has detected an inconsistency and should consider resolving it. In a world of imperfect data, wrong data should not break the database. Ontologies, being robust against such data errors, are a

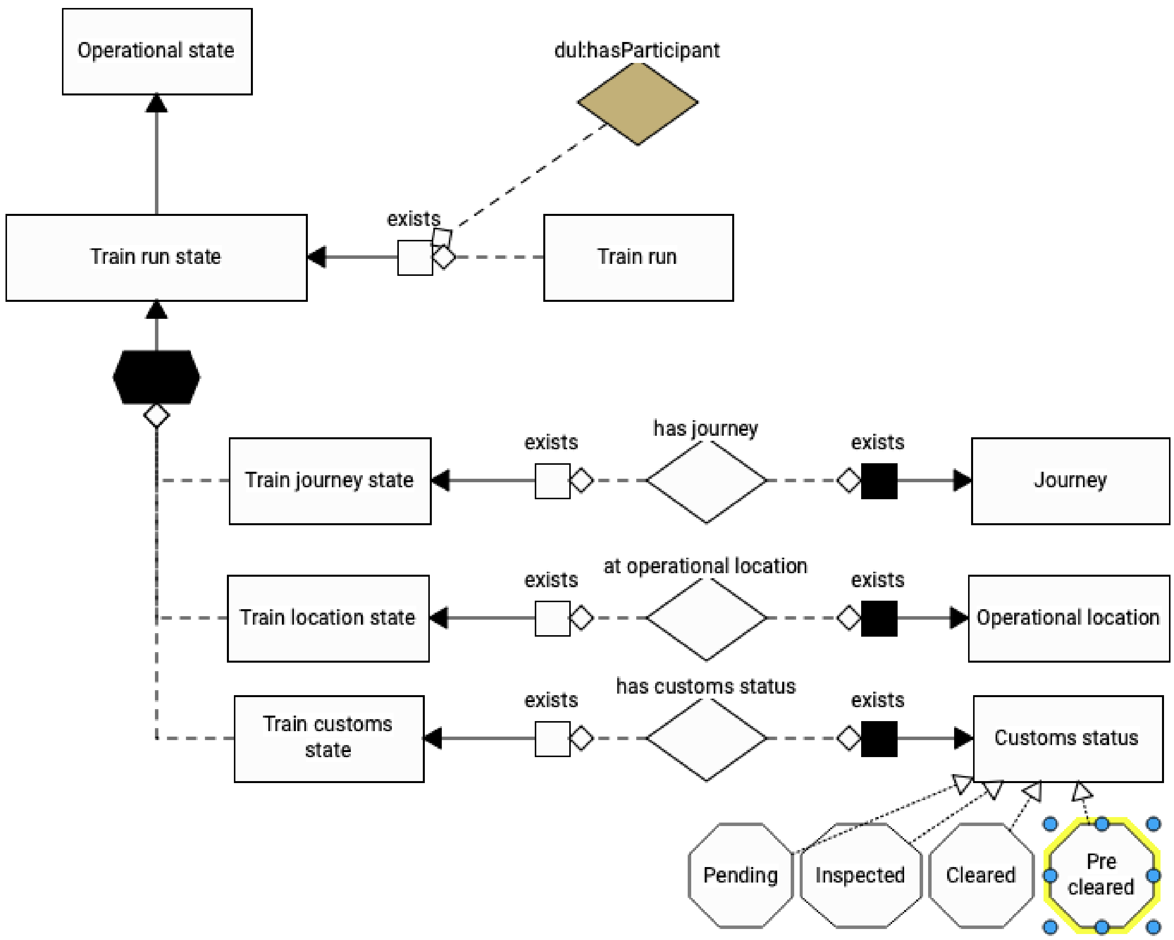


Figure 7: Train run state

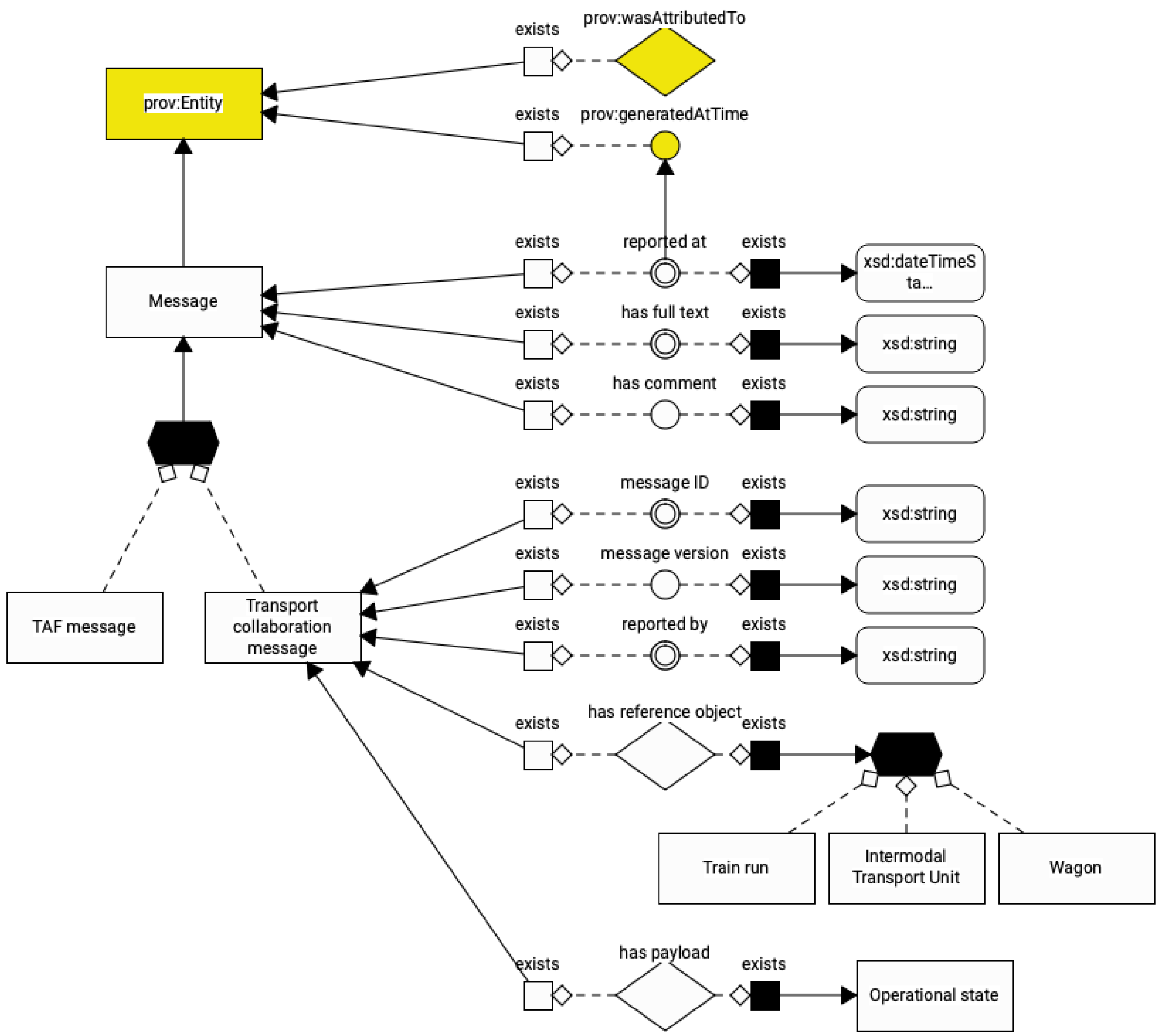


Figure 8: Message

suitable tool to represent them, but the onus of error detection is partly on the user: OWL2 has no numeric operators.

Operational times (instants or intervals) may have a "date and time of issue". This non-mandatory information is of interest in the case of repeated forecasting or revision exercises. The bulk of exchanged times, in an operational environment, is composed forecasts or revisions, so this "time property of time" is everything but ludicrous.

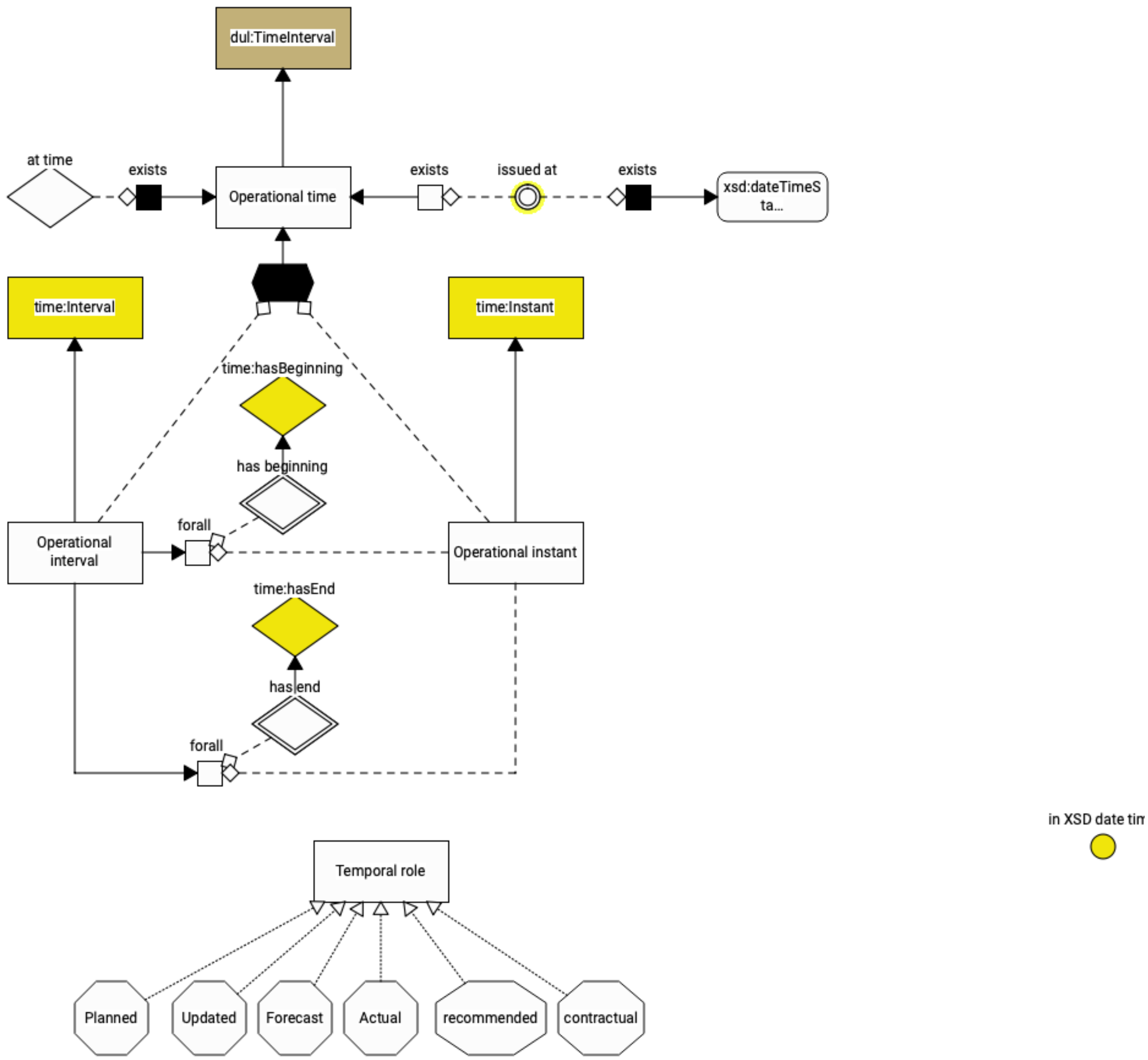


Figure 9: Time

The bottom part (Temporal role class and individuals) add meaning to time-dependent situations, i.e. whether they relate to a planned, revised, forecast, actual, etc. situation (the DUL term) or state (our used term).

10.3 Comments

10.3.1 Real Time applications

The time representation is conceptually compatible with the UML MARTE profile, so it can serve as a basis for a demanding real-time application. The main missing concept is that of Clock: we currently assume all clocks to be synchronized, so we ignore them altogether (there is no need for a Clock class).

We do not distinguish intervals from "proper intervals" (length > 0) in the sense of the W3C Time ontology: as no assumptions are made regarding the timestamps, if may well happen that the beginning and the end of an interval are distinct (in actual time), but have the same timestamp value. Calling it a "proper interval" would then be a semantic error.

10.3.2 Against open-ended intervals

Leaving out the beginning or end of an interval *may* be interpreted as the interval being "open-ended". We would like to sternly warn against such convention, since it breaks the monotonicity of time reasoning. Example: if a fire extinguisher has a usability interval but the end is missing from the available data, it shall not be understood as "usable forever", as this conclusion would immediately be falsified by the provision, at some stage, of the missing information.

In the context of railway applications, nothing is actually open-ended in time, and most data may be made available with some delay, so the "open ended" interpretation of any interval with missing data would be a gross mistake.

Original page: 90---Time.md

Varia
Dependencies

Varia
General and upper ontologies ### W3C Time ontology ### SOSA/SSN ### QUDT ### DOLCE+DnS Ultralite ### REGORG, ORG ## Semantic RSM ontologies

Varia

Rolling stock ontologies

Consist

Typology

ERA Concept Schemes

Original page: 97---Dependencies.md

11 References

...

Original page: 99---References.md