Railway systems' ontologies: A literature review and an alignment proposal

David Camarazo, Ana Roxin, Mohammed Lalou

July 2024

Railway systems' ontologies: A literature review and an alignment proposal

David CAMARAZO [a,b], Ana ROXIN [a] and Mohammed LALOU [a]
[a] *Laboratoire d'Informatique de Bourgogne (LIB EA 7534), University of Burgundy, Dijon, France*
[b] *Railenium Technological and Research Institute Railenium, Valenciennes, France*

Abstract. Railway systems are complex systems whose design and construction require many cooperating actors. Multiple models have been produced to reduce semantic heterogeneity and enable cooperation across many companies and persons. However, each model created has its point of view regarding the railway domain and uses its vocabulary. Therefore, it is necessary to investigate ways to align those models and to ease cooperation between actors. This paper first identifies the main ontologies proposed in the last two decades. Then, we study and compare those models and analyse the difficulties when aligning them. The results regarding our attempts to align identified railway ontologies are thus discussed.

Keywords. railway systems, railway ontologies, alignment framework, semantic heterogeneity, semantic interoperability

Introduction

A railway is defined in [17] as an aggregation of three components: rolling stock, infrastructure, and environment. Rolling stock refers to any vehicle that operates on rails. Infrastructure refers to four systems: power supply, telecommunication, signalling, and rail. It comprises elements that allow rolling stock to work correctly and transport people from point A to point B. Finally, environment refers to anything that existed before the development of the railway system (e.g., mountains, tunnels, bridges, rivers, etc.).

According to the definition above, a railway system is a system of systems, as seen from the definition in [5]. Furthermore, regarding security requirements, a railway system is considered crucial [8]. Therefore, it is essential to consider security constraints in the early phase of designing railway systems.

Multiple projects have been initiated to develop semantic models to assist the complex conception of such a system (we detail existing models in section 4). Those models aim to give a common vocabulary with an unequivocal interpretation for every actor implied in the design of railway systems. Since no model has taken hold, model heterogeneity and unclear interpretations of railway terminology exist (e.g. does "train" refer to a disjoint union of "freight train" and "passenger train" or can there be other kind of train ?). Aligning the different models is essential to ease cooperation across companies working on railway systems.

In this paper, we present three contributions. First, we evaluate the different railway ontologies that have been proposed. Second, we evaluate available tools to align ontologies automatically. Third, we propose a set of alignments between railway ontology models.

The rest of the paper is organised as follows: Section 1 details our motivation for the presented work, section 2 presents the fundamental notions about ontology and ontology alignment, railway domain ontologies, as well as an evaluation of the presented ones, is covered in section 3. Section 4 shows our alignment results with different tools. Section 5 evaluates the obtained alignments. Section 6 discusses our conclusion from our experiments. The last section concludes the paper and provides several future research directions.

1. Motivation

Railway systems' conception involves a complete modelling process [7].

The modelling process has four phases: requirement analysis, conceptualisation, architecture design and implementation. It aims to move from an abstract informal representation of the system to a more formal and concrete representation. To do so, different models are generated at each phase of the modelling process, depending on the point of view studied at the relevant phase. As shown in [15], many points of view about a system are represented during a modelling process. As it is difficult to find a person with all the skills required to model each system's point of view, we need different actors with different skills and vocabularies in each phase of the modelling process. This implies many actors have to cooperate during the modelling process.

According to feedback from Railenium engineers, it is challenging to reach a common understanding among the various actors involved [7]. Indeed, each actor has its specific vocabulary, tools, and languages, so implementing a coherent interpretation of these elements and contexts to control and verify the overall process is not trivial. Our goal is to conceive an automated approach for verifying the modelling process. This approach uses railway domain knowledge to check the consistency between models produced during the railway modelling process. As explained in our previous paper [6], we need an explicit and formal model of railway knowledge, i.e. an ontology. For that, we first analyse all existing ontologies in the field, each with its scope, point of view, and language. Then, to build a complete domain ontology, we need to establish equivalencies between different ontologies' concepts. We considered manual and automatic alignment approaches to match the meaning of related elements and tried to reach semantic interoperability among them. This paper evaluates railway ontologies, discusses alignment methods, presents the established alignments and evaluates them through reasoning.

Given the large number of classes and properties in the models, we look for an automated solution that motivates the need for an automated alignment method between ontologies of the railway domain. Still, those alignments require defining semantic relations among various ontologies' elements (classes and properties). This paper presents our defined alignments.

2. Background definitions - Ontologies and alignments

We define an ontology as a "formal, explicit specification of a shared conceptualisation" according to the definition [18]. From [11], conceptualisation refers to "an intentional semantic structure which encodes the implicit rules constraining the structure of a piece of reality". We say an ontology is formal, meaning it is written with a syntax that defines a domain's semantics and prevents ambiguous interpretation. Also, an ontology is explicit as it states explicitly the different relations and classes of a shared conceptualisation. Furthermore, it explains implicit knowledge shared in a general agreement.

An ontology is built using logical formula [3] with a vocabulary divided into two sets: classes **C** and properties **P**. An ontology has a terminological scheme, a TBox, and a set of assertions called ABox. The TBox specifies a hierarchy of classes, properties, and restrictions over elements from **C** and **P**. The ABox contains the ontology instances with their specific values according to the axioms specified in TBox.

We follow the definition provided by [20] regarding ontology alignments. An ontology alignment is the output of an ontology matching process. An ontology matching can be seen as a function $matching(O_1, O_2, A', p, r)$. Where O_1 and O_2 are the ontologies we are trying to align, A' is a set of already existing alignments, p is a set of parameters (e.g. threshold, metric, filter) for ontology matching, and r is a set of external resources that can be required for ontology matching (e.g. dictionaries or other ontologies). An ontology matching process produces an ontology alignment, i.e. a set of relations (e.g., equivalence, specification, generalisation, parthood) between classes or properties from O_1 and O_2. The matching process relies on various methods and resources; it can use existing alignment A' to compute new ones or a dictionary to compute equivalences between synonyms.

3. Ontologies for railway systems' modelling - Identification and evaluation

As mentioned in section 1, the modelling process requires many actors with different points of view over the domain they are working on. This heterogeneity leads to different interpretations regarding the models produced and the terms used during the modelling process. This heterogeneity of interpretations can cause errors in the prototype produced. To detect those errors in the modelling process, we propose to build an ontology that represents railway domain knowledge to check the consistency between railway domain knowledge and the models produced as intermediate results during the modelling process. Model is seen here as an abstraction of the system and can have different formalism level, going from unstructured text to Unified Modeling Language (UML) https://www.omg.org/spec/UML/2.5.1/PDF models. The following subsection presents the ontologies proposed to specify railway domain knowledge. In this section, we describe the different ontologies considered and evaluate them using quantitative and qualitative indicators.

3.1. Presentation of the models

The following paragraphs present the existing ontologies for specifying railway domain knowledge. We have selected ontologies listed during the Linx4Rail European project,

as, to the best of our knowledge, it is the only European project that has addressed the issue of building a global railway domain ontology. Given our scope to align existing railway ontologies, we aim to build upon the ontologies already considered for alignment in the context of the Linx4Rail project. Our ontology selection is further completed with ontologies found through keyword searches on Google Scholar and Linked Open Vocabularies https://lov.linkeddata.es/dataset/lov/about/. We used the keywords "railway domain ontology" and "railway ontology". Our efforts allowed us to find, in addition to the ontologies proposed in Linx4Rail, the ontology RaCoOn [19].

ERA. It is an ontology created by the European Union Agency for Railways and models knowledge about the legal sector and railway-related use cases. ERA is about vehicles and three infrastructure modules: infrastructure sub-system, alimentation sub-system, signalling and rail command. Compared with the other ontologies considered, this one is the smallest regarding classes and properties. It holds only tens of classes, while the other ontologies have hundreds of classes. It also contains around one hundred and fifty properties, unlike other ontologies considered in this study that can have more than three thousand properties (like TransModel). It is worth noting that the ontologies containing thousands of properties also contain redundant properties. Moreover, while other ontologies have primarily object properties, ERA contains mainly hundreds of data properties, while the other ontologies contain between 0 and 21 data properties. Additionally, ERA proposes an important hierarchy between classes, and it is freely available in Turtle format here[1].

IFC Rail. It is a subpart of the IFC (Industry Foundation Classes) model, which is an open schema developed and promoted by bSI (buildingSmart International). The IFC schema and its various subparts are also provided as an ISO standard, *i.e.* ISO 16739-1:2018. The model is presented in UML and considers the geometrical aspects of the railway domain, power supply, signalling, telecommunication, and rails. IFC Rail is an important model, one of the most complete we found. It covers all of the aspects of the infrastructure. We consider the ontology implemented during the OntoRail project; it can be downloaded here[2]. The associated ontology contains thousands of classes and properties with many redundancies. Moreover, the hierarchy between classes is more horizontal than the hierarchy we have found in the other ontologies.

Rail System Model. It is the rebranded version of the Rail Topology Model (RTM), which has become the Rail System Model (RSM). More complete than RTM, RSM models infrastructure and rolling stock elements, represented in UML format. RSM contains elements from every part of the infrastructure (power supply, signalling, telecommunication, rail). The associated ontology is reasonably large for humans, containing around one hundred classes and properties. However, it does not cover essential parts of infrastructure and has redundant properties. We note that RSM proposes a well-organised hierarchy between classes. While the model is promising, the implemented ontology has a narrow scope that needs more relevant classes in signalling, telecommunication, and power supply. It is available at OntoRail portal[2].

[1] https://data-interop.era.europa.eu/era-vocabulary/
[2] https://app.ontorail.org/download

Eulynx. It is a model created by the EuLynx consortium. It reuses pieces of RSM. Eulynx models signalling classes and traffic control are written using different graphical (UML-like) languages. The implemented version of the ontology is relatively large, with more than one thousand classes and more than two thousand properties. It covers an essential part of infrastructure but still lacks more specific classes about rail classes. The proposed hierarchy is rich, giving relations between classes and has a depth of six levels. The implemented ontology is available at OntoRail portal[2].

TransModel. It is a model sponsored by the European Comity of Normalisation and has been approved as a European standard. A European team has developed a model describing trains, geographic infrastructure, topologic information and some infrastructure information (e.g. accessibility, vehicle equipment, line equipment). The model can be visualised in UML. The implemented ontology associated with TransModel is the largest regarding the number of classes and properties. Then again, the ontology contains a lot of redundant properties. Additionally, some classes do not represent railway infrastructure concepts. The underlying hierarchy is considerable, presenting rich taxonomy for the various classes. The implemented ontology is available at OntoRail portal[2].

X2Rail-4. It is a project for studying various railway themes. The model is grounded in European Railway Traffic Management System (ERTMS) standards. X2Rail-4 manages railway traffic and captures knowledge about signalling and autonomous train control. On the one hand, the number of classes is reasonable (around one hundred). On the other, we note many redundant properties. The model covers many important infrastructure classes but lacks classes related to telecommunications. We also note that the hierarchy presented by the model is ”weak” and does not establish many relations between classes. There are only 52 rdfs:subClassOf relations for 551 classes. Moreover, the model does not include other properties or relations to add relevant semantics to the classes. The implemented ontology is available at OntoRail portal[3].

RailML. It is a RailML.org project that developed the markup language RailML in 2002. It specifies knowledge about the railway domain reusing the RailTop Model (RTM). It proposes a set of XML schemas to model the control system, infrastructure, rolling stock and traffic timetable. The models have been created from talks with various partners from different European countries (the list of partners is available here[4]). RailML aims to propose a standard interface to exchange railway data. We have examined the current version, the version 3.2. The model proposes a complete overview of the railway domain but it is not implemented as an ontology at the time of writing. We noted that works are in progress to align ontologies [13], but the results of those works are not yet published.

RaCoOn. It is an ontology developed by the University of Birmingham following the Neon method [10]. Its creation has been guided by three use cases that illustrate its scope: maintenance application, signalling conception, and infrastructure visualisation. The vocabulary uses elements from RailML. The ontology size is reasonable for humans based on the number of classes (hundreds). There are fewer properties than in the other ontologies, but it is important to note that there are almost no redundant properties. The hierarchy is rich, reaching six levels and introducing important class relations. However,

[3]https://app.ontorail.org/dow load

[4]https://www.railml.org/en/introduction/partners.html

the scope of the ontology lacks classes about telecommunications and power supply. The ontology is available at GitHub[5].

OntoRail. It is a federation of railway ontologies for different projects (ERA, Eulynx, IFC Rail, RSM, TransModel, X2Rail-4). It uses Simple Knowledge Organisation System (SKOS) vocabulary to express class relationships. At the time of writing, Ontorail is an ongoing project. It aims to propose alignment between ontologies, federate railway models, and archive different railway ontologies. Since OntoRail is a federation of ontologies and not a simple ontology, we will go into further detail about its structure in the next section, which will assess the various ontologies and evaluate them. It is possible to download the vocabularies and the relations between them here[3].

3.2. Evaluation of the related ontologies

After introducing the main ontologies, it is time to evaluate those implemented and available for free download.

First, considering OntoRail as a federation of railway ontologies, it is a direct answer to our problem as an attempt to align various railway vocabularies. However, it presents severe limitations to achieve our goal of creating an automated method to verify the modelling process. The SKOS relations do not formally specify semantic relations between terms. For the relation ”related_match”, classes linked with this relation do not inherit properties from their equivalent classes, contrary to owl:equivalentClass that creates a formal equivalency. Therefore, a reasoner cannot use alignments in OntoRail to infer knowledge or check consistency. Moreover, in the case of OntoRail, we can not directly map SKOS relations with formal relations. In fact, in OntoRail, ”related_match” serves as a way to describe an equivalence relation (e.g. era262:Signal skos:related_match rsm12:Signal) or a synonym relation (e.g rsm12beta:VehicleStop skos:related_match fcr:Bumper). Finally, OntoRail does not specify a term to refer to a class, and there is no hierarchy between terms that could help actors in the railway domain to agree on the vocabulary that must be used.

For RailML, we decided to exclude it from our work for the following reasons. This is because RailML is intended as an exchange format in XML for railway data, and its is more about the use of trains rather than their construction. Moreover, it is a proprietary format that actors can not always use in practice. For example, the RailML team refused to participate in Linx4Rail, a European project initiated in 2019 that addresses alignment problems and knowledge exchange in the railway domain.

Considering the discussion above, we now have seven free models implemented as ontologies: ERA, Eulynx, IFC Rail, RaCoOn, RSM, TransModel, and X2Rail-4. We have considered the various criteria provided by Noy et al. [12] for their evaluation.

The table in 1 recaps different measures studied on previously mentioned ontologies. To ease the reading, we use the abbreviations below:

- The reasoning result section sums up the result obtained when we reason over the ontology (more details below). The ”OK” mention indicates the reasoner has successfully ended its inference task and the ontology is consistent. The ”OOM” mention stands for ”Out Of Memory” and means the reasoner could not infer all

[5]https://github.com/UoB-BCRRE/Racoon

implicit knowledge successfully. Finally, ”inconsistent” means the reasoner found the ontology inconsistent.
- OP for Object Property.
- DP for Data Property.
- Ratio(%) I/CA: per cent ratio between the number of instances and the number of class assertions. It allows us to measure the extent to which ontology instances belong to ontology classes, thus identifying if there are orphan instances. This ratio is lower for manually constructed ontologies (i.e. eRA and RaCoOn) and higher for the automatically generated ontologies (i.e. 211 for TransModel). A higher value for this ratio should be interpreted as an insufficient number of class axioms to allow for a satisfying classification of instances.
- Ratio(%) SBO/C: per cent ratio between the number of rdfs:subClassOf relations and the number of classes. It gives a hint about the richness of the taxonomy.
- Ratio(%) OtherP/P: per cent ratio between the number of various properties and the number of domain/range properties. This allows us to see if the ontologies contain axioms a reasoner can exploit or if the ontology relies only on domain/range axioms

We use three symbols to evaluate the corresponding criteria for qualitative criteria: check, tilde, and cross. For the quality section in Table 1, a check means that the criterion is overall respected with few (between 1 and 3) or no counterexamples; a tilde implies that the criterion is overall respected but has many counter-examples (more than 3 counterexamples but the criterion is still respected by the large majority of classes). The cross means too many counterexamples exist to consider the criterion respected. For the ”Scope” section, the check indicates that more than three terms are related to the corresponding infrastructure component. The tilde means at least one term related to the element but less or equal to three. Finally, the cross means that no terms about the corresponding component exist. We now describe the various quality criteria we considered for our evaluation:

- Readable URIs: check if an entity’s URI is readable [2] by a human being.
- Acyclic classes: check no class is a subclass of another class that represents the same concept.
- Unique classes: Check that a label corresponds to exactly one class (there are no classes with the same label; the contrary would introduce confusion)
- Unique properties: Check that a label corresponds to exactly one property (there are no properties with the same label)
- Relevant label: check labels are readable by a human being and refer to a railway class.
- Naming convention: check a naming convention is respected.

To reason over every ontology, we used Protégé in version 5.6.1 https://protege.stanford.edu/. We chose Protégé as it proposes a readable interface to explore the inferred hierarchy and explanations for every inferred knowledge. We chose the Pellet reasoner with its default configuration based on the survey [1]. We justify this choice because Pellet is usable, still maintained, follows OWL2 standards, and checks ABox’s and TBox’s consistency with explanations for every inferred knowledge. This allowed us to discover that TransModel was inconsistent (e.g. the property $transmodel : minCardinality$ whose range is a negative integer has positive integers as objects). We found dozens of inconsistencies

Criterion	Sub-criteria	ERA v3.0.0	Eulynx 09/03/2023	IFC Rail v2021-02-01	RSM v1.2	TransModel v6.56	X2Rail-4 v2022-01-13	RaCoOn
General	Classes	68	1163	1368	227	2209	551	426
	Object Properties	150	2110	3220	190	3666	1766	70
	Data Properties	300	0	8	0	21	0	12
	Individuals	12	5863	4589	1050	6582	3268	73
	Annotations	41	115	105	106	448	95	20
	DL Expressivity	ALUHIQ(D)	AL	AL(D)	AL	AL(D)	AL	ALCROIF
	Manually created	yes	no	no	no	no	no	yes
	Reasoning result	OK	OOM	OK	OK	Inconsistent	OOM	OK
Class Axioms	rdfs:subClassOf	44	921	497	196	2855	52	460
	Other Axioms	0	0	0	0	0	0	9
Object Property axioms	OP_Domain	140	2110	3220	191	3849	1766	60
	OP_Range	145	2110	2309	191	3849	1766	59
	OP_Other	39	0	0	0	0	0	35
Data Property axioms	DP_Domain	291	0	8	0	8	0	10
	DP_Range	294	0	8	0	21	0	9
	DP_Other	129	0	0	0	10	0	3
Individual axioms	Class Assertions	10	11085	7468	1914	13917	5069	27
	Other Assertions	0	0	0	0	6834	0	46
Ratios	Ratio(%)I/CA	83,33	189,1	162,74	182,3	211,44	155,11	36,99
	Ratio(%)SBO/C	64,71	79,19	36,33	86,34	129,24	9,44	107,98
	Ratio(%)OtherP/P	16,18	0	0	0	0,13	0	21,59
Quality	Readable URIs	✓	X	X	X	X	X	✓
	Acyclic classes	✓	✓	✓	✓	✓	✓	✓
	Unic classes	✓	✓	≈	✓	✓	≈	✓
	Unic properties	✓	X	X	X	X	X	✓
	Relevant labels	≈	≈	✓	≈	✓	✓	✓
	Naming convention	≈	X	≈	≈	X	✓	≈
Scope	Rail	✓	X	✓	✓	≈	✓	✓
	Signalling	≈	✓	≈	X	X	✓	✓
	Power Supply	X	≈	✓	X	X	≈	X
	Telecommunication	X	✓	✓	X	X	X	X

Table 1. Table summarising the above ontologies' evaluation

in TransModel, making it too hard to fix for us. Unfortunately, we could not reason with Pellet over Eulynx and X2Rail-4. While other reasoners, such as Hermit or ELK, could end the reasoning task properly on those ontologies, we found they could not detect the inconsistency in TransModel contrary to Pellet. Therefore, we could not conclude about the consistency of Eulynx and X2Rail-4.

Table 1 summarises our evaluation of the examined ontologies.

Six of them, namely ERA, Eulynx, IFC Rail, RSM, TransModel, and X2Rail-4, have been implemented for the project OntoRail. Therefore, they contain cycles, i.e. classes that are subclasses of another classe that represents the same concept (e.g ontorail:AnnotationProperty is subclass of owl:AnnotationProperty). This is due to meta-classes (e.g. Data Property, Annotation Property, Class, Object) defined in the OntoRail vocabulary and redefined in OWL. Five have been automatically extracted from graphic models: Eulynx, IFC Rail, RSM, TransModel, and X2Rail-4. Those ontologies contain

a lot of duplicates. without names, with unreadable names, or with names that do not refer clearly to railway classes (which is the case for X2Rail-4). Moreover, their URIs are unreadable for human beings because they are too long and need to expose precise semantics. Except for X2Rail-4, many ontological class names do not refer to railway classes (e.g. rsm:Enumeration). Finally, we see that manually built ontologies, ERA and RaCoOn, have the highest expressivity and specify more properties other than domain and range properties. This fact suggests that the knowledge from these ontologies can be exploited by a reasoner for automated tasks such as verification.

To have a complete overview of the railway domain, our work must also consider the scope of the evaluated ontologies. Table 1 clearly illustrates that finding a complete and well-structured ontology is hard. While an automatically generated ontology can be interesting regarding its scope, which is the case for Eulynx, IFC Rail, and X2Rail-4, the underlying TBox model presents design and structural issues and does not expose rich semantics. Moreover, no properties are defined other than domain and range properties except in TransModel. We noted this issue for all three ontologies generated automatically from XMI or UML models *i.e.* Eulynx, IFC Rail and X2Rail-4.

4. Railway Systems' ontologies Alignemnts

We aim to align the seven implemented ontologies we obtained and presented in the previous section: ERA, Eulynx, IFC Rail, RaCoOn, RSM, TransmMdel, and X2Rail-4. For that, we consider both manual and automatic methods. In the following, we show each methodology and the related results.

4.1. Manual alignments

Manual alignments were defined between RaCoOn and ERA ontologies because they are the only ontologies correctly conceived, with precise semantics and human-readable.

We first identified alignments between classes. Fifteen relations between classes were identified. Some of them are listed in Table 2[6]. Some relations can be trivially found by comparing the class labels. We can create an equivalent relation between classes if the labels are identical, except for their naming convention. We found relations such as era:Signal owl:equivalentClass racoon:Signal. However, this basic approach does not suffice for a thorough alignment. When labels contain similar words or classes with similar ancestors, we must look at the classes' comments, if any, to establish equivalency or hyperonymy relations. This allows us to find relations such as racoon:Train rdfs:subClassOf era:Vehicle. In this example, we have established the equivalency relation due to matching definitions between those classes. RaCoOn defines a train as a "vehicle or formation of vehicles that provides a service", which is close to the definition of a Vehicle in ERA, which is "A specific vehicle or wagon able to operate on railway lines". Finally to express a parthood relation between classes (*e.g* an era:Signal is a part of racoon:SignalGroup) we use the object properties "$bfo : part\_of$" and "$bfo : has\_part$" from the Basic Formal Ontology (BFO).

We then investigated the different properties of the two considered ontologies. They are listed in Table 3 (using the same prefixes as in Table 2). Given the differences in de-

[6]Table uses the prefixes racoon for http://purl.org/rail/core/, era for http://ontorail.org/src/ERA/era300/

racoon:RailwayTrain	rdfs:subClassOf	era:Vehicle
racoon:InfrastructureConcept	owl:equivalentClass	era:InfrastructureObject
racoon:Platform	owl:equivalentClass	era:Platform
era:Signal	bfo:part_of	racoon:SignalGroup
racoon:OCP	rdfs:subClassOf	era:OperationalPoint
racoon:SidingTrack	rdfs:subClassOf	era:Siding

Table 2. Manual class alignments' between ERA and RaCoOn ontologies

Object properties' alignements		
era:manufacturer	rdfs:subPropertyOf	racoon:manufacturer
racoon:mainDirection	rdfs:subPropertyOf	era:trackDirection
Datatype properties' alignements		
era:vehicleNumber	rdfs:subPropertyOf	racoon:id
era:signalOrientation	rdfs:subPropertyOf	racoon:orientation

Table 3. Some manual property alignments' between ERA and RaCoOn ontologies

sign, fewer alignments were defined, and only one owl:equivalentProperty relation was identified between racoon:trackDirection and era:trackDirection. The main difficulties come from the fact that different design choices were made in each ontology. For example, on the one hand, RaCoOn defines racoon:axleWeight as an owl:ObjectProperty and as "the maximum load per axle in tons for a track element". On the other hand, ERA only considers era:minAxleLoad as an owl:DatatypeProperty and defines it as "Minimum permitted axle load, given in tons". Additionally, some elements are modelled as properties in one ontology and as classes in the other *i.e.* rac:vehicle is an owl:ObjectProperty, while ERA considers it as a class. The different alignments are presented in Table 3, differentiating object and datatype properties.

4.2. Automatic alignments

Let us first present existing alignment tools from the literature we evaluated, and then we will discuss each in detail.

4.2.1. Existing alignment tools

We want to align ontologies using automated tools to save costly expert consultations, which can be laborious. In this work, and to compile a list of automated alignment tools, we have examined two critical sources[4], and [16]. We have identified three tools that are available for free download and may be customised to use with different datasets, namely, Limes, LogMap, and OnAGUI[7].

Limes (for LInk discovery framework for MEtric Spaces) is "a framework for discovering links between entities contained in Linked Data sources" as described in the user manual[8]. The tool reads from two Linked Data sources, such as RDF graphs, and computes a mapping from instances of one source to instances of the other. To map

[7]Respectively https://dice-research.org/LIMES, https://www.cs.ox.ac.uk/isg/tools/LogMap/ and https://github.com/lmazuel/onagui?tab=readme-ov-file

[8]https://dice-group.github.io/LIMES/#/user_manual/index

an instance to another, Limes computes a similarity score between instances based on a method elected by the user among various methods. The methods are classified depending on the type of instances they are comparing (strings, point sets, vectors, time measures, etc.).

LogMap is an ontology-matching tool that uses various methods to compute mappings between instances from two ontologies. Mainly, it computes similarities between instances using String comparison methods and filters results using the knowledge specified by the ontologies.

OnAGUI is a tool that computes string similarities between instances from two ontological sources. It presents a simple user interface allowing users to edit and export the computed mapping. Similarities can be calculated using the I-sub distance or the Levenshtein distance measure.

4.2.2. Alignment tool chosen and results analysis

In the previous section, we introduced selected alignment tools for evaluation. In this section, we discuss each tool.

We begin with Limes, which is highly configurable. First, we can align ontologies using string similarity or semantic proximity following wordnet hierarchy [14]. Once this choice is made, we have different computing methods (mathematical operations) to compute the similarity between terms. When the computing method has been selected, we must specify a threshold between 0 and 1 to filter out results with a similarity score too low to be relevant. From what we observed, we obtained interesting results with a threshold of around 0.7. A threshold less than 0.5 outputs too many irrelevant results (e.g. racoon:ETCS Level 2 Standard and eulynx:limitedBySignal), while a threshold higher than 0.85 provides mappings only when terms are almost identical (transmodel:id and x2rail4:id). Similarity computing method also has a significant impact on results. We have retained four measures with many pertaining scores without too many errors. The cosine and the Ratcliff methods are best for string measures. The Li, Leacock and Chodorow measures are the best semantic measures. However, even with those parameters, we observe many issues in the results. First, Limes does not provide the nature, *i.e.* the type of mapping relation. This requires extra manual labour to sort the mappings between equivalences, ”is a” or ”subclass of” relation (e.g. racoon:Tunnel and eulynx:Tunnel are equivalent, while racoon:Signal group and eulynx:Signal is linked through a parthood relation). We also observe duplicate results that modelling particularities in our ontologies can explain. Moreover, since the ontologies may contain irrelevant labels, we can find absurd output mappings such as the one between lynx:appliesToTrDrBus and transmodel:0..* with a trust score of 0.7. Therefore, asserting reliable and definitive rules regarding correct methods and thresholds is challenging. What we have for now are only heuristics.

Considering LogMap, it is straightforward to use and compute many pertaining mappings. Its internal optimisation to automatically filter irrelevant results decreases the manual labour required to establish proper alignments. Moreover, the framework can compute three types of relations: ”equivalent”, ”more specific than”, and ”more general than”, which helps to understand the calculated mapping. We found 56 relations between classes from RaCoOn and four other ontologies: Eulynx, IFC Rail, RSM, TransModel, and X2Rail-4. Most of the provided alignments are correct; some of those correct align-

class1	Alignment	class2	Trust degree
racoon:Freight Train	owl:equivalentClass	eulynx:FreightTrain	75.4%
racoon:Balise	owl:equivalentClass	ifcrail:Balise	50%
racoon:Crossing(Track Infrastructure)	owl:equivalentClass	rsm:Crossing	73.6%
racoon:Route	owl:equivalentClass	x2rail4:Route	70.3%

Table 4. Table presenting some of the correct alignments computed by LogMap between RaCoOn and other railway ontologies (To ease the reading, we used labels in table instead of actual iris)

ments are presented in Table 4[9]. In this case, we only find equivalent relations. The alignments are between classes with the same labels (e.g racoon:Balise and ifcrail:Balise).

Other alignments have labels with some words in common; in that case, we checked the definition (the comment associated with the class) to decide whether the alignment was correct. With LogMap, we note that if the trust degree is below 0.2 for a given alignment, the output seems incorrect, linking classes with no semantic relations (e.g. racoon:SafetySystem owl:equivalentClass ifcrail:MonitoringSystem). We also observe that the relation type can be incorrect if the trust degree is below 0.6. Table 5[10] lists our modified alignments, changing equivalent relation into parthood relation.

class1	Alignment	class2
racoon:SignalGroup	bfo:	era:Signal
racoon:Signal with function	bfo:has_part	eulynx:SignalFunction
racoon:SetOfPoints	bfo:has_part	eulynx:Point
racoon:Signalwithfunction	bfo:has_part	x2rail4:SignalFunction

Table 5. Table presenting the modified alignments between RaCoOn and other railway ontologies(To ease the reading, we used labels in table instead of actual iris)

For OnAGUI, thanks to its user interface, domain experts familiar with computer science find it straightforward to use. This makes it easy to import source ontologies, export results or manually edit mappings. However, it does not provide the nature of the relationships in the computed mappings, and the user must select a threshold to filter the results. The tool indicates by a colour code which mappings are most relevant, and accordingly, the best threshold value is 0.75. We have found out that the tool is helpful to establish obvious relations between classes that have similar labels automatically: racoon:Border :match era:Border, racoon:ExitSignal :match era:Signal, racoon:HomeSignal :match era:Signal, racoon:Platfrom :match era:Platform. However, we can already see that the tool mixes equivalent relations (e.g. racoon:Border and era:Border) with hyperonymy relations (e.g. racoon:ExitSignal and era:Signal). Moreover, some relations seem wrong regarding the classes definition: racoon:SpeedCapability :match era:loadCapabilitySpeed, racoon:Status :match era:State.

Unsurprisingly, and regarding the Ontology Alignment Evaluation Initiative (OAEI) results [16], LogMap is the more mature tool. We can select data sources to align, and it returns relevant results while filtering many irrelevant mappings. It is also important to note that since our ontologies present several issues and limitations regarding their de-

[9]The prefixes used are eulynx http://ontorail.org/src/Eulynx/eul230309/, ifcrail http://standards.buildingsmart.org/IfcRailBusinessConcepts#, rsm http://ontorail.org/src/RSM/rsm12/, x2rail4 http://ontorail.org/src/X2Rail-4/x2r4_25b/

[10]Uses the additional prefix of bfo http://purl.obolibrary.org/obo/bfo.owl

sign, other tools may have trouble loading them or computing readable mappings. For example, OnAGUI only works with RaCoOn, ERA and X2Rail-4. We faced a similar problem with Agreement Maker Light, a tool we could not evaluate because it could not give a single human-readable mapping. With Limes, we observed a lot of duplicates in the mappings or mappings with empty terms. While those problems also exist with LogMap (for example, we could not load either ERA or TransModel ontologies), there are fewer occurrences, and again, most of the mappings are relevant. We have stored the results of our experimentations with the different tools in a GitLab https://gitlab.com/c4m4r4z0/railway-ontologies-alignment/-/tree/main.

5. Evaluation of our alignments

As previously done in (3), we use Protégé 5.6.1[11] to check the consistency of our proposed alignments. We aimed to check the inferred hierarchy. Therefore, we needed a tool that allows a proper examination of the inferred class relations and not only examining instance knowledge inference. To ensure a complete check, we used the reasoner Pellet. As stated in 3, other reasoners might leave some inconsistencies undetected. However, our previous experiment also showed that using Pellet with Eulynx or X2Rail-4 returns an "Out of memory" error. In this experiment, though, we were only interested in checking the impact of our alignments, i.e. the inferences produces by our alignments in the aligned ontologies. We decided to consider Eulynx and X2Rail-4 classes only if they are part of one of our alignments. To do so, we manually extracted the classes of Eulynx and X2Rail-4 that appear in an alignment, keeping the class hierarchy between them if there are any to build a new ontology we called $eulynx2rail-4$.

Then we imported in Protégé five ontologies: ERA, eulynx2rail-4, IFCRail, RSM, RaCoOn. We excluded TransModel due to its inconsistencies. Once the ontologies were imported, we added in Protégé the alignments previously defined. We excluded the alignments between parthood alignments as they cannot be expressed in Protégé. Finally, we launched the Pellet reasoner over the ontologies.

First of all, no inconsistent inferences were produced. Moreover, we found new taxonomic relations. For example, we inferred that the concepts $ifcrail:TrainDetector$, $racoon:TrainDetector$, $x2rail4:TrainDetector$, classes from IFCRail, RaCoOn and X2Rail-4 are subclasses of the $era:TrainDetectionSystem$ class in ERA. Similarly, the $era:Tunnel$, $racoon:Tunnel$, $x2rail4:Tunnel$ classes from ERA, RaCoOn and X2Rail-4 are subclasses of the $eulynx:LineSideLinearConstruction$ class in Eulynx. Besides relations between ontologies, one relation among RaCoOn's object properties was inferred; namely, the property $racoon:mainDirection$ in RaCoOn was inferred as a subclass of the property $racoon:trackDirection$. Regarding this case, one could argue that adding relations between concepts or properties inside an ontology can falsify its knowledge representation. However, we argue that completing one ontology with knowledge from other ontologies is an important benefit of the alignment task and the alignment itself.

[11] https://protege.stanford.edu/software.php

6. Discussion

This section presents the conclusions we have subsequently drawn from our experimentations. First, we have identified some derelictions in existing railway ontologies. In the complex railway domain, it is hard to conceive a complete domain ontology that efficiently captures all relevant domain classes. We also noted that those derelictions complexify alignment tasks. We could only manually align two ontologies, namely ERA and RaCoOn. However, even in this case, we needed hours to analyse them and provide alignments. With automated tools, once we have chosen and set up the tool, we quickly obtain and evaluate relevant results (in a few minutes). Also, it is important to note that a manual review is required to specify the nature of the computed relations. Without railway experts, evaluating the output relations and asserting correct relations between classes is difficult. Overall, we observe that for a domain as complex as the railway domain, with ontologies that present design errors, the ontology alignment task is difficult and tedious, and it seems difficult to have a fully automated method that does not require domain expert intervention.

Conclusion and Future work

During the modelling process of railway systems, one must establish a shared vocabulary for the railway domain to reach semantic interoperability between considered models. Several vocabularies already exist for this domain but are not semantically aligned. Our paper presented these vocabularies, evaluated them according to several well-established criteria, and then described our methodology for aligning them. While this also falls under the scope of the OntoRail project, we go beyond the non-formal alignments provided by OntoRail and present 62 formal alignments: 24 computed manually and 38 automatically. Indeed, the alignments developed under OntoRail mainly rely on properties from the SKOS vocabulary and thus cannot be exploited by a reasoner. For example, OntoRail alignments cannot be used in an automated inference process for verification purposes. Given our goal, we have established our need to establish a set of alignments between railway vocabularies that are semantically aligned; our contribution lies in the 62 alignments provided. They are available for download in https://gitlab.com/c4m4r4z0/railway-ontologies-alignment/-/tree/main.

Our methodology first considered defining such alignments manually, namely between RaCoOn and ERA ontologies. Given the complexity of the considered ontologies, this task took a lot of work. We then turned to automatic alignment tools, namely Limes, LogMap and OnAGUI. Here, the design issues noted in our evaluation of the different ontologies hindered the outputs. Indeed, given that all ontologies, except ERA, were automatically created from UML-like models and schemas, they did not respect ontology modelling or Linked Data principles. Following our evaluation, LogMap was the best performer, identifying many proper alignments without too many mistakes. It still requires extra manual tuning to precisely determine the kind of relation expressed by the alignment computed. In our previous work [6], we have further specified our research issues related to the general goal defined in this paper (*i.e.* verifying railway systems' modelling processes). We also presented some alignments we could use to federate the different ontologies considered.

Our future work will consider further pushing these results to the railway community, *i.e.* by adding them to the OntoRail platform. Additionally, we aim to pay specific attention to the ongoing work regarding the railML ontology development and its mappings with ERA and IFC-Rail ontologies [13]. Building upon these alignments, we aim to define and implement a federation among railway ontologies by reusing existing architectures such as FOWLA [9]. Coupled with a semantic enrichment module, such federation can serve as a basis for an automated approach to verify railway systems' modelling processes through the different phases. To do so, our work will consider the definition of a top-level domain ontology for Model-Driven Approaches (MDA).

References

[1] Konrad Abicht. Owl reasoners still useable in 2023. *arXiv preprint arXiv:2309.06888*, 2023.

[2] Danny Ayers and Max Völkel. Cool uris for the semantic web. *Woking Draft. W3C*, 2008.

[3] Franz Baader, Ian Horrocks, and Ulrike Sattler. *Description logics*. Springer, 2004.

[4] Mike Bergman. 30 active ontology alignment tools, 2018.

[5] Mario Bunge. *Treatise on basic philosophy: Ontology II: A world of systems*, volume 4. Springer Science & Business Media, 2012.

[6] David Camarazo, Sana Debbech, Ana Roxin, and Annabelle Gillet. Modélisation des systèmes ferroviaires-objectifs, approches et problématiques. In *eduBIM 2023-9e édition des Journées de l'enseignement et de la recherche autour du BIM et de la maquette numérique*, 2023.

[7] Nadia Chouchani, Sana Debbech, and Matthieu Perin. Model-based safety engineering for autonomous train map. *Journal of Systems and Software*, 183:111082, 2022.

[8] Sana Debbech, Philippe Bon, and Simon Collart-Dutilleul. Improving safety by integrating dysfunctional analysis into the design of railway systems. *WIT Transactions on The Built Environment*, 181:399–411, 2018.

[9] Tarcisio M Farias, Ana Roxin, and Christophe Nicolle. Fowla, a federated architecture for ontologies. In *Rule Technologies: Foundations, Tools, and Applications: 9th International Symposium, RuleML 2015, Berlin, Germany, August 2-5, 2015, Proceedings 9*, pages 97–111. Springer, 2015.

[10] Asunción Gómez-Pérez and Mari Carmen Suárez-Figueroa. Neon methodology for building ontology networks: a scenario-based methodology. 2009.

[11] Nicola Guarino and Pierdaniele Giaretta. Ontologies and knowledge bases. *Towards very large knowledge bases*, pages 1–2, 1995.

[12] Natalya F Noy, Deborah L McGuinness, et al. Ontology development 101: A guide to creating your first ontology, 2001.

[13] Linnea Cathrina Olsen. Ontology in norwegian digital infrastructure model (dim) project. In *45th RailML Conference*, 2024.

[14] Ted Pedersen, Siddharth Patwardhan, Jason Michelizzi, et al. Wordnet:: Similarity-measuring the relatedness of concepts. In *AAAI*, volume 4, pages 25–29, 2004.

[15] Klaus Pohl, Harald Hönninger, Reinhold Achatz, and Manfred Broy. *Model-based engineering of embedded systems: The SPES 2020 methodology*. Springer, 2012.

[16] Mina Abd Nikooie Pour, Alsayed Algergawy, Patrice Buche, Leyla Jael Castro, Jiaoyan Chen, Hang Dong, Omaima Fallatah, Daniel Faria, Irini Fundulaki, Sven Hertling, et al. Results of the ontology alignment evaluation initiative 2022. In *17th International Workshop on Ontology Matching*, 2022.

[17] Walter Schön, Guy Larraufie, Gilbert Moëns, and Jacques Poré. Signalisation et automatismes ferroviaires. *Ouvrage en trois tomes. La Vie du Rail*, 2013.

[18] Rudi Studer, V Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data & knowledge engineering*, 25(1-2):161–197, 1998.

[19] Jonathan Tutcher, John M Easton, and Clive Roberts. Enabling data integration in the rail industry using rdf and owl: The racoon ontology. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 3(2):F4015001, 2017.

[20] Xingsi Xue, Chaofan Yang, Chao Jiang, Pei-Wei Tsai, Guojun Mao, and Hai Zhu. Optimizing ontology alignment through linkage learning on entity correspondences. *Complexity*, 2021:1–12, 2021.