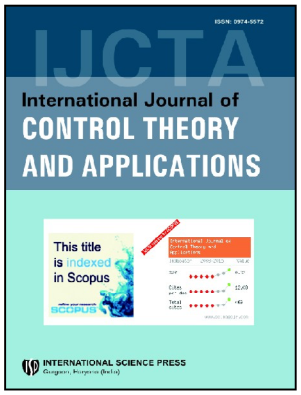


International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Volume 10 • Number 11 • 2017

An Ontology Based Data Model for Railways

Kumar Abhishek[1*], M.P. Singh[2*] and Sachin Gupta[3*]

[*]Dept. of Computer Science &Engg., NIT Patna, Bihar India
E-mail: kumar.abhishek@nitp.ac.in[1], mps@nitp.ac.in[2], sachin.gupta192160@gmail.com[3]

Abstract: The domain of the railway system is very vast and complex for integration. It includes several sub-domains likemechanical and electrical which is a challenge in them. So, to make things easy for the application developers andimproving the efficiency of the railway system, it is necessary to integrate the data. This helps in managing the data and improvingthe efficiency of railway systems. Several techniques exist for integration and this paper proposes ontology to integrate theinformation at data level. By providing the semantics to the data by using ontology, reasoning support can be added to the dataand decision support systems can be developed to automate thing using software agents. A generic model is proposed in thispaper which should satisfy the standards and specifications of most railways. A use-case for Indian Railways is discussed with some examples.

Keywords: Railway Ontology, Rail ML, Basic Formal Ontology

1. INTRODUCTION

A railway is a very large-scale domain and presentslots of challenges to the engineers working to improve it. It also includes various other fields as it sub-domains which makes the task even more complex.Some of them are mechanical and electrical parts which are used in the locomotives, wagons andelectrical instruments used in them. The railway organizations using these parts may or may not manufacturethese by themselves. Thus, it is important to integrate and standardize the data of these parts andalso of the system as a whole. This will help in managing the system when multiple entities are involvedand increase efficiency of the system. First example, if some other organization has already achieved betterresults at completing the same task, then there methods can be used. The key to all this improvement ifthe integration of the data.

As explained in [1], there are many techniques which can be used for integration at various levels andthis may be done using different technologies such as XML, UML or ontologies. The authors have shownthat ontologies would give better results than the UML methodology. Other benefit of using the ontologyis that it would help in building a semantic web, which can later be used to create a knowledge base, whichcan be useful in training the software agents. All of these can be used together to build a decision supportsystem (DSS) for railways.

2. BACKGROUND AND MOTIVATION

Railways are an important and one of the cheapest, modes of transport these days. A large number ofpassengers travel daily in trains all across the world. This leads to generation of large amounts of data.

Some of the data is useful for the operating organization only, but some of it, such as train schedules,number of trains passing through a station or duration of the travel is critical for the passenger as well.

This enormous amount of data can be stored in traditional database management systems such as Oracleor MySQL, but there are many problems related to these technologies. First, they are not able to provide semantic information with the data. Second, the data is not in a state to be used by any inference engine.

For such purposes, where the databases have issues, ontology can be used. The data mapped to a domainspecificmodel can be used to save information and reasoner can be applied thereafter to infer knowledge.

The proposed ontology was prepared with the goal that it will be helpful for the software agents thatmaybe deployed in due time in several areas of the vast railway domain. Also, it can be used to design adecision support system for the railways if enough information is provided to the tools using the ontology.

The ontology proposed in this paper can be used to overcome the different problems related to dataintegration and database management systems. The proposed ontology describes the various concepts ofrailway. It defines these concepts as terms which are modelled as classes and properties. The ontology canbe categorized as a domain ontology which is specific for railways. It should satisfy the structure of mostrailway organizations.

3. TECHNOLOGIES USED

The ontology is developed with the help of protégé, which is an open-source platform, developed by Stanford University, for building ontologies and knowledge management systems. It also supportsplugins such as SPARQL (Simple Protocol and RDF Query Language) Query, SWRL (Semantic WebRule Language) etc., which can be used to define the rules or execute queries on the ontology.

SWRL was used to define rules using which knowledge was inferred from the existing information. SPARQL was used to execute queries and fetch the results for example use cases. Pellet reasoner was used to use the SWRL rules with the ontology to gain knowledge from available information.

4. ORGANISATION OF PAPER

Section 5 provides the literature survey done for the purpose of this research. Section 6 details the structure of the ontology and the method used to design the model. Section 7 provides the example for Indian Railways along with the rules written to extract information. Section 8 provides the information of mapping the proposed railway ontology to the upper-ontology. Section 9 explains thebenefits and use cases for the ontology. Examples are given with SPARQL queries that can be used by the application developers to build tools using this ontology.

5. RELATED WORK

Integration of the data is a complex task and it has been a topic of research for many years. Severalresearch projects have been done to achieve integration in various fields, railways is no exception tothat. Several strategies have been proposed for the integration. Some of them suggested the integrationof applications, using APIs while other considered integrating various applications into a single largeapplication.

An extensive research for data integration was done by authors of [1]. They consider and discuss thevarious levels at which the data integration could be applied. Three approaches were discussed. First, developmentof a specific domain-wide application by integrating all available applications is considered.This approach is not feasible as not every problem can be thought of while developing the application.Developers will always have to face one or more problems because of using tools developed by others.

Second approach was to integrate the applications by providing APIs from the existing applications. Thisis a better option as compared to the first one. But it presents a problem that if an application changesthe API slightly, then all the applications implementing that API will also need to be changed, and thetime and cost involved might not be feasible. Although the API may be made backward compatible, butstill it may raise concerns because of security reasons or any other adverse conditions. In that case, the API would be needed to implement at very short notice, and many application may not coupe up, andeventually fail. This will lead to huge losses to the entities using those applications. The third approachdiscussed in the paper was to integrate the data using a domain specific data-model, which has been agreedupon by the users of that domain. While the first two approaches might need to make major changes tothe applications, the third approach appears to be most feasible as only a few minor changes would berequired to the existing applications. In fact, the advantages of using data model for integration have beengiven using example of SID (Shared Information Data) developed by TM forum, the Tele-Managementforum, which is a consortium for the telecom industry. The authors discuss the two technologies, UMLand ontology, to design the model for data integration. After providing a thorough discussion for the twotechnologies, the authors conclude that the ontology is a better option as it has some advantages over theUML such as reasoning support and the ability to express semantics. The challenges that the authors facedincluded the semantic and structural heterogeneity of data, and the limitations of the data that an organizationcould provide.

Several other papers were reviewed which present the various steps in the development of an ontology andin some cases also explain the various development tools using examples.

Natalya F. Noy et al. [3] , describes the use of ontology in the modern world. The authors discuss the variousadvantages of developing ontology for a domain. The basic concepts of ontology i.e. the classes,properties, facets and their use in a knowledge base has been explained. The two different approaches, thetop-down and bottom-up approach, for creating an ontology, the naming conventions, defining the slots,roles and facets have also been discussed. Overall, an overview of how to create ontology and the issuessuch as creating hierarchy of classes and properties and reusing previously developed ontologies are also discussed.

The problem of integration in railways is not new and hence, a number of projects that existed for thesame purpose were also studied. At the times when XML was being adopted as a de facto standard forintegration, the EuRoMain project [4] proposed schemas but only for a limited cases of data integration.The primary goal of the project was to achieve syntactical integration and thus it failed to provide answerto problems which included the semantics, such as the synonym problem, where the same object can bereferred to by different names. This is when it was realized that a better technique was required to definea single specification of information.

The InteGRail [5] approach was an FP6 EU research project which was started for intelligent integrationof railway systems by using an ontology-based approach. Its aim was to provide semantics to the datathat can be used to create knowledge by processing the information. According to its website,” InteGRailis an Integrated Project, which is a research project addressing a wide number of coordinated objectives ina specific domain. InteGRail deals with railway information systems and their integration for improvedoverall efficiency and performance of the future European railway system[6].

RailML(Railway Markup Language), version 2.2, [7] presents an XML-based approach that simplifiesdata exchange among railway applications by defining schemas for various spheres of the domain. Thisincludes subschemas for infrastructures, rolling-stock and timetable, which describe how the data shouldbe structured so that communication can be made easier in the heterogeneous environment.

Hannes Bohring et al. [8] and Nora Yahia et al. [9] discuss the rules using which an XML/XSD documentcan be converted to OWL. They provide solutions to various issues and the most of the rules given in thetwo publications are similar to each other.A study of these papers, along with [10] and [11] gave a detailedidea of how to approach the problem.

There were several others, who worked on the areas which would benefit by ontology for the railways.

Supriyo Ghosh et al. [12]describe a multi-agent coordination based system to automate the track managementsystem of the railways which was being controlled manually till now. The paper proposes the ideaof using software agents (autonomous entities with own power and decision making capabilities). Multipleagents will be able to coordinate and cooperate with each other while negotiating on various terms toavoid contradiction. The authors guarantee that the system is 100% safe and collision free, theoretically.They also suggest developing an ontology which will help the agents and make a rule base for helping theagents.

Felice Romano et al. [13] propose a system, based on ontology, which would help in maintenance of therailway subsystems. Currently there are two approaches: corrective maintenance, which deals with repairingthe system after the fault has occurred, and planned maintenance, in which a subsystem or part wasreplaced after a given criteria is fulfilled. Each of these techniques had a disadvantage. In case of correctivemaintenance, the impact on the service and efficiency of the system is the main drawback, whereasfor the planned maintenance, early replacement of equipment's reduces their useful life, thereby increasingthe cost. The authors propose a predictive maintenance methodology, which was a part of the InteGRailimplementation, which would be useful in maintaining the efficiency of the system while deviating thecost by little margin. The cost of maintenance would decrease as it would be predicted from an array ofcomponents, but it would be balanced by the specialized components that would be installed in order toget the continuous and accurate data. By the help of an example of door-controller, the authors have shownhow the proposed method will improve the efficiency of the railway systems.

Ontologies were also used by the authors of [14], to develop the design tool SIA, which will help thedesigners to create more efficient and complex structures for electrification of the railway portal frames.The document describes the use of ontology in creating expert systems, as ontologies can be useful ininferring knowledge from the data, and can also be used to integrate and share the information on varioussoftware platforms.

The problem of integrating information in the railways system was also looked upon in [15]. The roleof ontology in decision making was considered and the advantages of ontology over traditional databasedesign were illustrated. Class model retrieval and instance retrieval were illustrated in combination withOpen World Assumption (OWA) as it has the power of uncovering the unspecified information from theheterogeneous systems. The author proposed a way to provide decision support to railways by implementingthe developed ontology to integrate railway condition monitoring data.

After the study of these papers, it was found that several techniques exist for the integration and ontologywould be most helpful in current scenario.

RailML is used as a base for the ontology proposed in this paper. The reason for choosingRailML as abase was that it provided the structure for almost every part of the domain, and is actively maintained. It iscurrently being used by several rail companies in EU, which is another advantage as the companies might be having software to process the data already present in that format. It will be easy for the developers tocreate tools that can process the information from the ontology as they already have the knowledge of thedata model.

6. PROPOSED ONTOLOGY

The ontology proposed in this paper is a domain ontology prepared specifically for railways. Most railwaysystems around the globe might find it useful for implementation for various projects. The ontologyis written in OWL-DL language, which is recommended by W3C as one of the ontology sub-language.

6.1. Design

The design of the proposed ontology is based on RailML (version 2.2). RailML is the XML Schemawhich is used by certain railway companies in the EU for the purpose of data interchange. It categorizes thedata into 3 different categories: infrastructure, rollingstock and timetable. If there is any data that cannotbe categorized into these 3 categories, then it is put under the 'others' category. The RailML hasdefined schemas for all these categories which specify the restrictions on the content of these categoriesi.e. the elements and attributes of the schema. The ontology has been developed by converting theseelements and attributes into classes and properties. Rules were adopted from research works mentioned inprevious section.

6.2. Conversion rules

The rules which were used to convert the XSD to OWL are defined by [8], [10] and [11]. Most of therules were used as defined in these papers, but at some places they needed to be changed because of therestrictions of the OWL, Protégé or the reasoner. This section explains the conversion rules that were usedto convert the RailML schema to OWL. Any exceptions that occurred have been specifically stated. Therules for creating entities in ontology are as follows:

Data-type: New datatypes were created based on the simpleType elements defined in the RailML. Thesedefine the various datatypes that apply to different classes. For example, a class "tLoad" can use "tForceNewton" as a datatype, where tForceNewton extends the xsd:decimal to define the force in Newton. Similarly,"tWeightTons" and "tWeightKG" defines the weight in ton and Kg respectively. Both theWeightTonsand WeightKG extend the xsd:decimal and also apply some restrictions on it, such as maxInclusive andmaxExclusive. There are several simpleType which use enumeration restrictions on strings; these extendthe xsd:string in the ontology and allow the user to choose only from a limited range of values.

There are several restrictions that are defined in the RailML schema, such as fractionalDigits restrictionon xsd:decimal, which are not supported by the reasoner. These restrictions were dropped while applyingthe conversion. Some simpleType elements in RailML extend other simpleType defined in the sameschema, such as tForceNewton extends tForce. But the reasoner only allowed to extend the data-types thatwere defined in OWL 2 . Since the user defined data-type could not be inherited, the pre-defined data-type,using which the base type was created and used as a base type and multiple restrictions applied to createthe new data-type. Table 1 shows an example of the data-type definition.

Table 1
Conversion from simpleType to DataType

Name	*simpleType as defined in RailML*	*Datatype as defined in Ontology*
tAngleDeg	Decimal (fractionalDigits=3, totalDigits=6, minInclusive=360, maxInclusive= 360)	Decimal [maxInclusive=360, minInclusive=-360]
tAngleDegFullCircle	tAngleDeg (minInclusive=-180, maxInclusive=180)	Decimal [minInclusive=-180, maxInclusive=180]
tAngleDegQuadrant	tAngleDeg (minInclusive=0, maxInclusive=90)	Decimal [minInclusive=0, maxInclusive=90]

Classes: The element defined as complex types are mapped to the ontology as classes. The complex type element occurred as: a) global, named complex types, and b) local anonymous complex types. According to the rules, the anonymous complex types would have been converted to the anonymous classes; but Protégé does not support anonymous classes, so they were also mapped as named classes, with a prefix "**anon_**". The attribute groups that were defined in the RailML schema were also mapped as classes. Theprefix 'a' specifies that the class was an attribute group in the RailML.

Table 2
Conversion of elements in RailML to class in OWL

Element name in RailML	*Element type in RailML*	*Class name in ontology*
aRelPosition	attributeGroup	aRelPosition
tBalise	Named complexType	tBalise
anonymous	complexType	anon_zValue

The classes'with't' as a prefix in their name define the restrictions on the classes defined under a subcategory. These classes act as superclass for the classes which will be used to create instances of the entities. Following is an example from the ontology which would better explain it

eTrainPartsubClassOf**tTrainPart**
eTrainPartshas_trainParts**eTrainPart**

Here, **tTrainPart** describes the restrictions for the train part and **etrainPart** is its subclass. The individuals would be created under **eTrainPart**. One or more of these individuals maybe related to an **eTrain-Parts** using the *has_trainParts*objectProperty.

Properties: When a complexType element (which has already been mapped to ontology has a class) encapsulates another element (which is also mapped as Class/Datatype depending on its type), then the relationship is mapped to the ontology using properties. The type of the property i.e. whether it will be an objectProperty or a datatype property is decided by the fact that whether the inner element is a simpleType or a complexType. If the inner element is simpleType, then a datatype property is created. If the inner element is a complexType, then an objectProperty is defined. The outer element is defined as a domain, whereas the inner element is the range of the newly created property. In case of an objectProperty, a prefix "**has_**" has been added.

For example,

1.
```
<xs:complexType name="eTracks">
    <xs:element name="track" type="eTrack" minOccurs="0" maxOccurs="unbounded">
    </xs:element>
    </xs:complexType>
```
2.
```
<xs:attributeGroup name="aSignal">
    <xs:attribute name="sight" type="rail:tLengthM" />
    <xs:attribute name="type" type="rail:tSignalType" />
    .
    .
    .
    </xs:attributeGroup>
```

In example 1, both **eTracks**and **eTrack** are constructed as classes in the ontology. The inner element "track" has been created as an object property "*has_track*". The restriction owl:minCardinality is also applied with a value of 0.

In example 2, "**aSignal**" is an attributeGroup (which is mapped in ontology as a class), with the "**sight**" and "**type**" attributes. These attriibutes will have literals as values, so these attributes are mapped as datatype properties with "**aSignal**" as domain for both and "*tLengthM*" and "*tSignalType*" as range for "**sight**" and "**type**" respectively.

6.3. Railway Ontology

The proposed ontology, named railway ontology, contains a hierarchy of 549 classes and 731 properties out of which 277 are objectProperties and the remaining 454 are the datatypeProperties. Similar to the design of RailML, the ontology consists of 3 main classes: infrastructure, rollingstock and timetable.

Following is a brief description of the some important classes and properties in each category and the entities they represent.

1. **Infrastructure:** This category focuses on the infrastructure on the railway including the network, stations, platforms etc. The information in gathered from the following classes:

   a) **infrastructure**: It acts as a container class for all the infrastructure elements. All elements that form a part of infrastrucutre are directly or indirectly related to the infrastructure class.

   b) **eTrackElements**: This class is used to create the track elements that can be "touched in real life", such as tunnels, bridges, level crossings etc.

   c) **eTrackTopology:** It acts as a container class for the network elements that cannot be "touched in real life", such as trackBegin, trackEnd, borders etc.

   d) **tGeoCoord:** It provides definition for the geographical position of an element in 3-coorindate system i.e. latitude, longitude and altitude. It is important in places where relative position of the element cannot be used.

   e) **eOperationalContnrolPoints**: It is a container class for the operation or control points. An operation or control point (OCP) on a railway line is a place where the traffic is dealt with or at which an authority to proceed is given under the system of working. It has various related classes such as, **eOcpPropOperational** and **eOcpPropEquipment**, which relate to it using some property.

There are many more classes that are a part of the infrastructure category. These classes define the various tangible as well as intangible features of the infrastructure like speed profiles and gradients.

2. **Rollingstock:**This category provide the information for all the vehicles used by railways including the locomotives, freight and passenger wagons and multiple units. Combinations of single vehicles are used to denote the composition of the entire train. Following are the important classes which can be used to create individuals for the rollingstock.

   f) **rollingstock:** It acts as a container class for all the vehicle elements. Every individual that makes up a part of rollingstock is directly or indirectly related to this class.

   g) **eVehicle:** The individuals of this class would represent a single vehicle, with its details, such as vehicle type, efficiency, brake information etc.

   h) **eFormation:** This class provides container for the individuals which depict the formation of a train. It may consist of **tTrainEngine**, **eTrainOrder**, **tResistance** etc.

   i) **eTrainOrder:** It relates to various vehicles using **tVehicleRef**, which is a class containing the reference of the vehicles.

Several other classes are used to completely map the vehicle data to the ontology. Type of train: freight or passenger, engine technology, force required etc. with other details can be represented using these classes.

3. **Timetable:** This category defines the data model for all kind of time table or operational related data including the train parts, operating periods, rostering etc. Following are the important classes that constitute this category:

   a) **timetable:** This is the container class for all the classes and individuals that are a part of the timetable category. All the individuals of this category relate to this class, directly or indirectly.

   b) **tTrainPart:** The basic part of a train with its characteristics such as operating period or formation. This includes the information of the path of the train such as the sequence of OCP with the corresponding schedule of the train.

   c) **tTrain:** A group of one or more trainPart makes up a train and represent the operational and/or commercial view of the train.

   d) **eOcpsTT:** The individuals of this class would contain the information of the OCPs in the path of the train. Every OCP will have an associated counter which can be used to represent the path of the train. The information of the OCP-type, whether it is stop or pass is also stored. The offset and alignment with reference to the ocpRef is also saved.

Several other classes such as **eArrivalDdepartureTimes**, **eRostering**, **eStopDescription**etc. are used to model the timetable data.

7. ONTOLOGY EXTENSION FOR INDIAN RAILWAYS

The railway ontology is able to represent most of the data of the railways and most railways must be able to map their data to this ontology without making any major changes. As an example use-case, the railway ontology was extended to be used by Indian Railways.

7.1. Structural organization of Indian Railway:

The Indian Railway network is too large to be controlled from a single point. Thus, the entire network is divided into 17 zones, which further divide into 69 divisions, with 4-5 divisions in almost each zone. Every division consists of several stations which can be categorized into class A+,A,B etc.

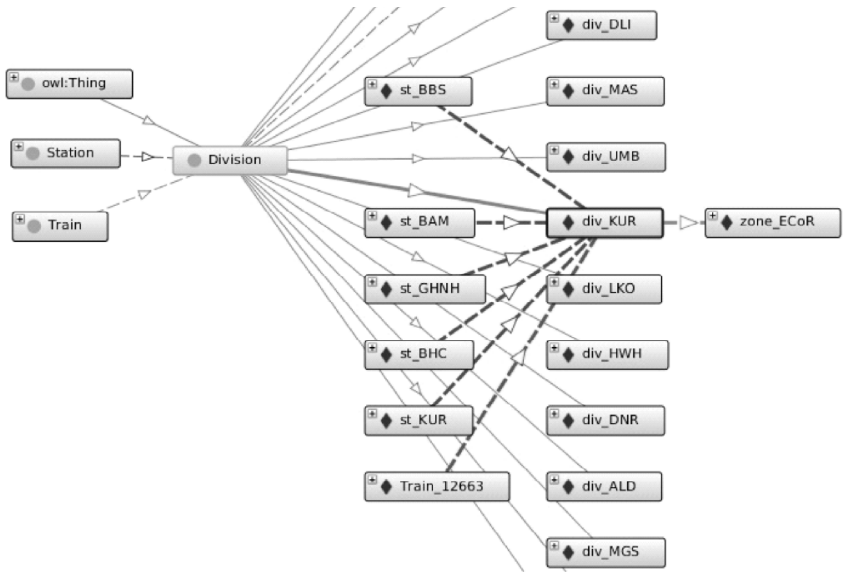


Figure 1: Divisions and Stations in Indian Railway

Based on the structure of the organization, several classes and properties were introduced to the ontology as equivalent class or subclass of the existing ontology. Some of them are described below:

7.2. Classes

The main classes concerning Indian Railways in the proposed ontology are as follows:

1. **Zone:** The individuals of this class are the 17 zones into which the Indian Railway is divided.
2. **Division:** It is the class for all the 69 divisions of the Indian Railways.
3. **Station:** Any place where traffic is dealt with is known as a station. It is equivalent to eOcp class which forms a part of the infrastructure in the ontology.
4. **Train:** Refers to an entire train. This class can be made equivalent class of the eFormation if enough information for the train is available.
5. **StationTimetable:** It provides a model to denote the time of a train for a station. The information about the station, such as if it is the source or destination station, or a hasStopCount if it is not the source or destination, but is somewhere on its path. The hasStopCount is a datatypeProperty, which acts as a counter, whose value is set as '1' at the source station. It is equivalent to the **eOcpTT** in the base ontology. Several restrictions have also been applied to the class. In fact, **this is the only information which is mapped to the ontology, rest all information is extracted using the SWRL rules and the reasoner.**
6. **TrainTimetable:** The individuals of this class would denote the timetable for a given train. It is related to multiple StationTimetable (atleast 2, which would be the source and destination for a train) instances. The instances of these classes are related using the **for_train** property.

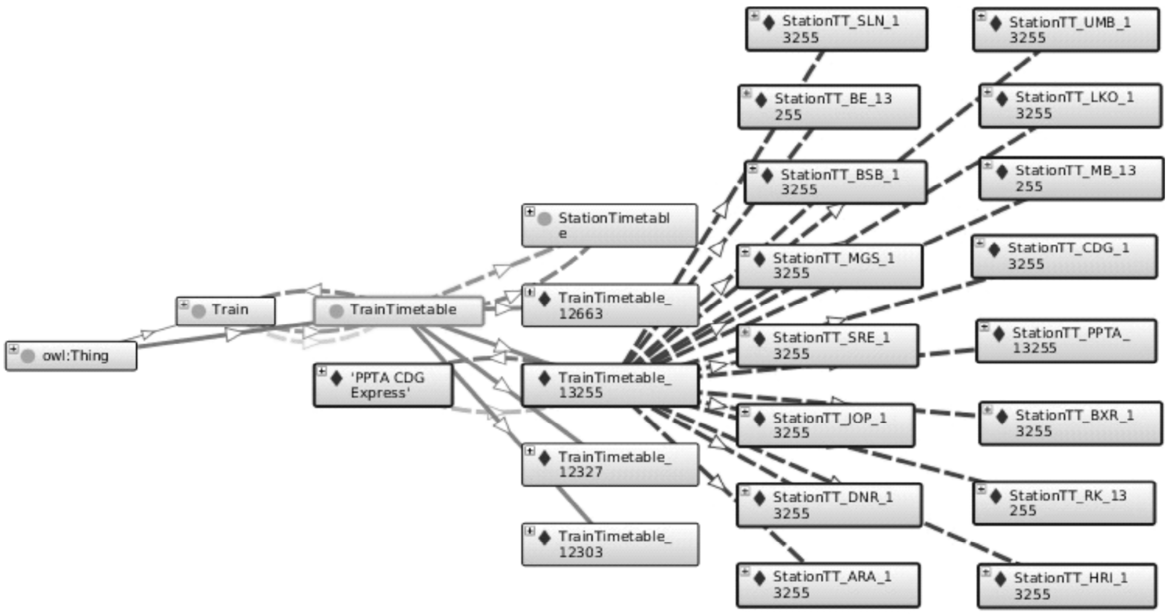


Figure 2: Image showing list of Station Timetableina Train Timetable

Figure 2 shows the relationship between **TrainTimetable** and **StationTimetable**. The 'PPTA CDG Express' has a **TrainTimetable**which has multiple **StationTimetable**.

7.3. Properties

7.3.1. ObjectProperties

The main object properties in the proposed ontology are as follows:

1. **isStationInDivision**: It relates a station to its division.
2. **isDivisionInZone**: It relates a division to a zone.
3. **trainStopsAtStation**: It relates a train to a station on which it has a halt.

Several other objectProperties are there, which have been inferred using the SWRL rules (described in next section). These include *trainStopsInDIvision, trainStopsInZone, isStationInZone, hasSource, has-Destination* etc.

7.3.2. DatatypeProperties

The main datatype properties in the proposed ontology for Indian Railways are as follows:

1. **hasArrivalTime**: It has the range xsd:time and applies to an instance of **StationTimetable**. It stores the arrival time of a train at a station. If the station is the source of the train, then the axiom will be missing from the **StationTimetable** instance i.e. it will only have "*hasDepartureTime*", and the value of "*isSourceStation*" would be set to "true".
2. **hasDepartureTime**: It is similar to *hasArrivalTime*, but it stores the departure time of a train from a station. In case where a station is the destination station of a train, this axiom would be missing from that instance and the value of "*isSourceStation*" would be set to "true".
3. **isSourceStation**: relates a **StationTimetable** to a station and a train. It is set to true be a station is source of a train.
4. **isDestinationStation**: It is similar to *isSourceStation*, but stores the information for destination instead of source.
5. **hasStopCount**: It is a positive counter which stores the stop count of the train for the stations that it stops at. The counter starts at '1' for the source station.

7.4. SWRL rules

For inferring new knowledge from the already given knowledge, some rules were defined using the SWRL. Table 3 describes the rules that were defined

Table 3
SWRL rules used in ontology for Indian Railways

Name	*Statement*
R1	StationTimetable(?stationTT) ^ forTrain(?stationTT,?train1) ^ TrainTimetable(?trainTT) ^ forTrain(?trainTT,?train1) ->hasStations(?trainTT,?stationTT)
R2	TrainTimetable(?trainTT) ^ forTrain(?trainTT, ?train) ^ Train(?train) ->hasTimetable(?train, ?trainTT)
R3	hasTimetable(?train, ?trainTT) ^ hasStations(?trainTT, ?stationTT) ^ forStation(?stationTT, ?station) ->trainStopsAtStation(?train, ?station)
R4	hasTimetable(?train, ?trainTT) ^ hasStations(?trainTT, ?stationTT) ^ forStation(?stationTT, ?station) ^isStationInDivision(?station, ?div) ->trainStopsInDivision(?train, ?div)
R5	hasTimetable(?train, ?trainTT) ^ hasStations(?trainTT, ?stationTT) ^ forStation(?stationTT, ?station) ^isStationInDivision(?station, ?div) ^ isDivisionInZone(?div, ?zone) ->trainStopsInZone(?train, ?zone)
R6	isStationInDivision(?station, ?div) ^ isDivisionInZone(?div, ?zone) ->isStationInZone(?station, ?zone)

Explanation

R1: Given the individuals of **StationTimetable** i.e. all stations with the information of the arrival and departure times of trains stopping at that particular station, and the individuals of **TrainTimetable** i.e. for which train the **TrainTimetable**is to be created, this rule can be used to generate the complete schedule of a train. This is done by using the *for_Train*property, which is common to individuals of both these classes.

R2: Given the individuals of **TrainTimetable**(which is related to train by "*forTrain*" property), this rule is used to assign a **TrainTimetable** to a train.

R3: A train (?train) has a timetable (?trainTT), which contains a number of **StationTimetable** (?sta-tionTT) that belong to a specific station (?station). This information is processed to infer the stations at which the train stops.

R4: Using information from R3 and the property "*isStationInDivision*", using which a station relates to a division, the divisions in which the train makes a stop is inferred.

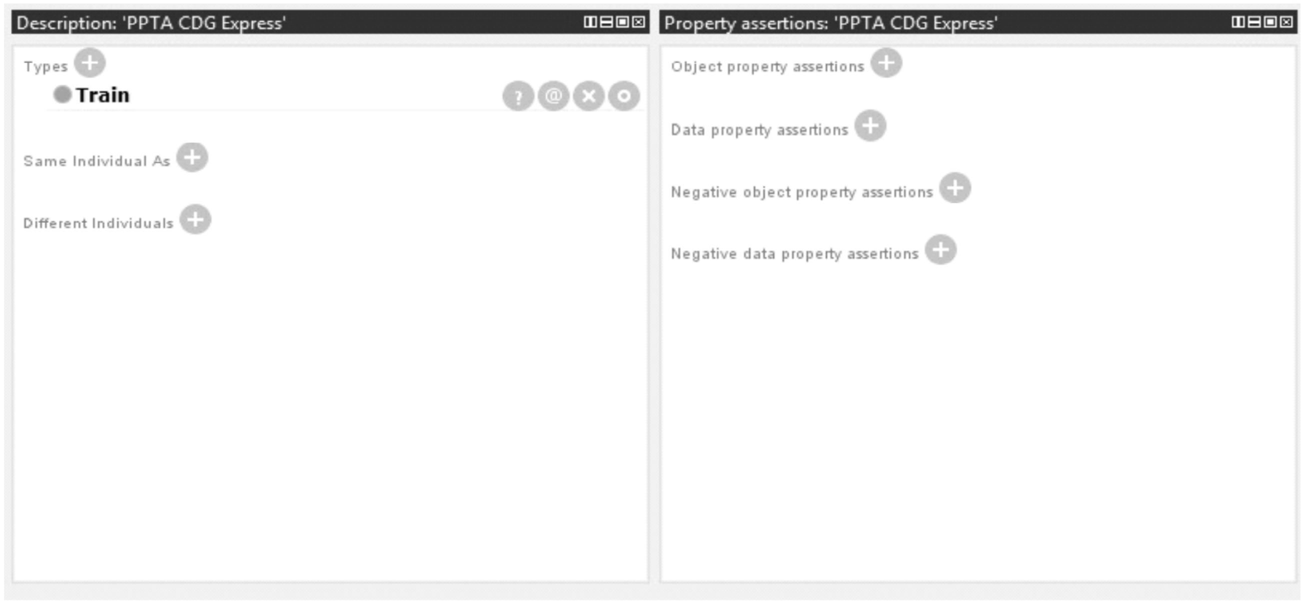
R5: An extension of R4, where the property is used to map the zones where the train would make a stop.

R6: The properties "*isStationInDivision*" and "*isDivisionInZone*" are combined to infer the zone to which the station belongs.

Rules R3,R4,R5,R6 prove useful while comparing the path of two trains. The routes of two trains can be checked using against each other to find any intersection points.

7.5. Reasoner

The Pellet reasoner is one of the most efficient reasoners when the SWRL rules are being used. It was available as a plugin for Protégé, thus it was used. Some of the information that was inferred using the SWRL rules explained above is given below:


Description: 'PPTA CDG Express'

Types

Train

Same Individual As

Different Individuals

Property assertions: 'PPTA CDG Express'

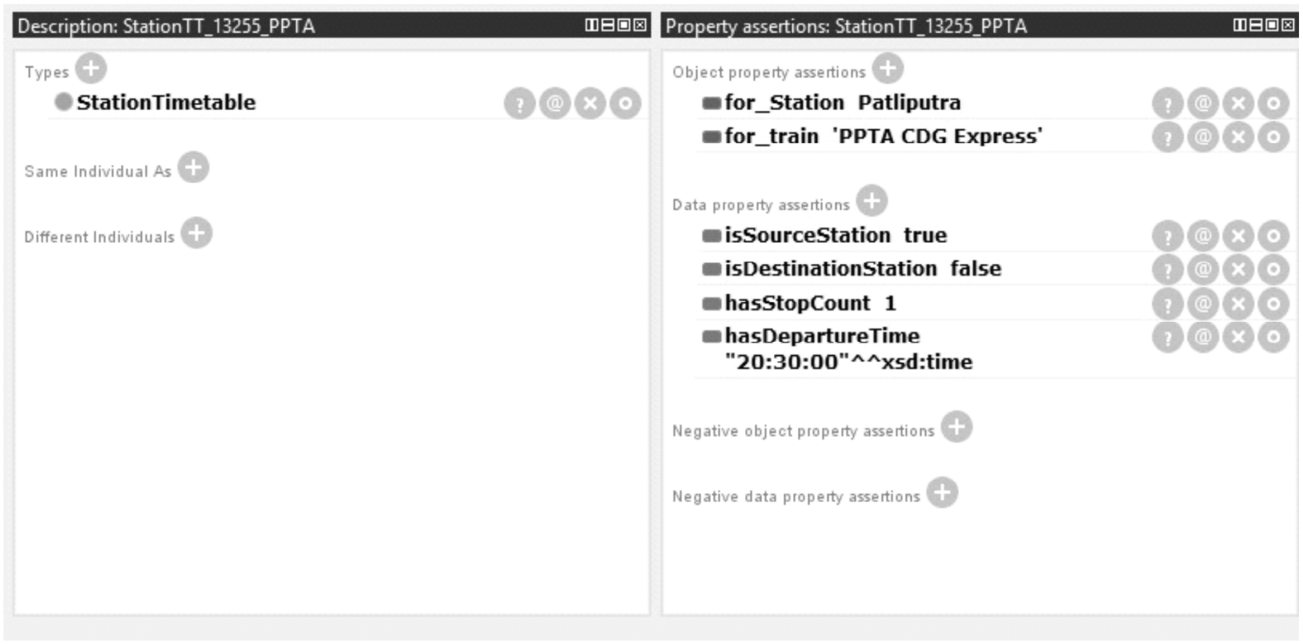
Object property assertions

Data property assertions

Negative object property assertions

Negative data property assertions

Figure 3: Asserted property values for 'PPTA CDG Express'


Description: StationTT_13255_PPTA

Types
StationTimetable

Same Individual As

Different Individuals

Property assertions: StationTT_13255_PPTA

Object property assertions
for_Station Patliputra
for_train 'PPTA CDG Express'

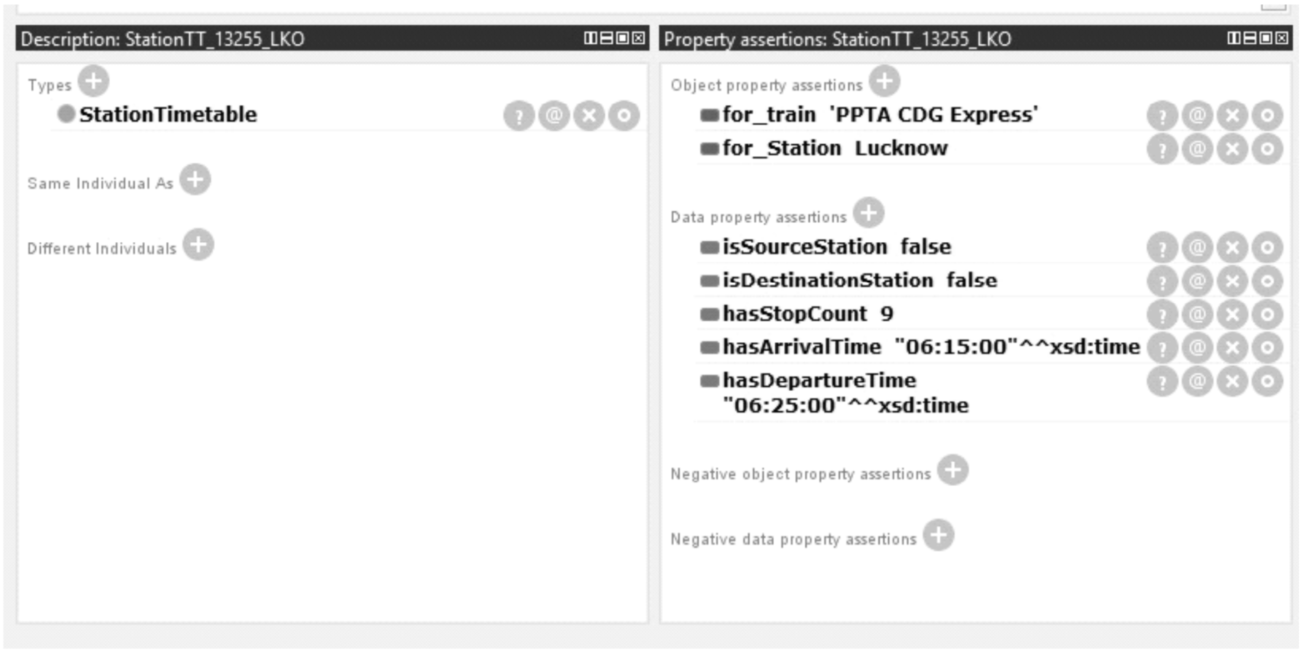
Data property assertions
isSourceStation true
isDestinationStation false
hasStopCount 1
hasDepartureTime "20:30:00"^^xsd:time

Negative object property assertions

Negative data property assertions

Figure 4: StationTimetable for 'PPTA CDG Express' for station Patliputra

Figure 3 shows the properties for the train 'PPTA CDG Express' before the reasoner was started. Figures4, 5 and 6 show the StationTimetables for the same train. These individuals were related to TrainTT_13255 to form the route of the train. Figure 7 shows the stations at which the train stops, along the information of the source, destination and timetable. This information has all been inferred by applying the SWRL rules on the existing values.


Description: StationTT_13255_LKO

Types
StationTimetable

Same Individual As

Different Individuals

Property assertions: StationTT_13255_LKO

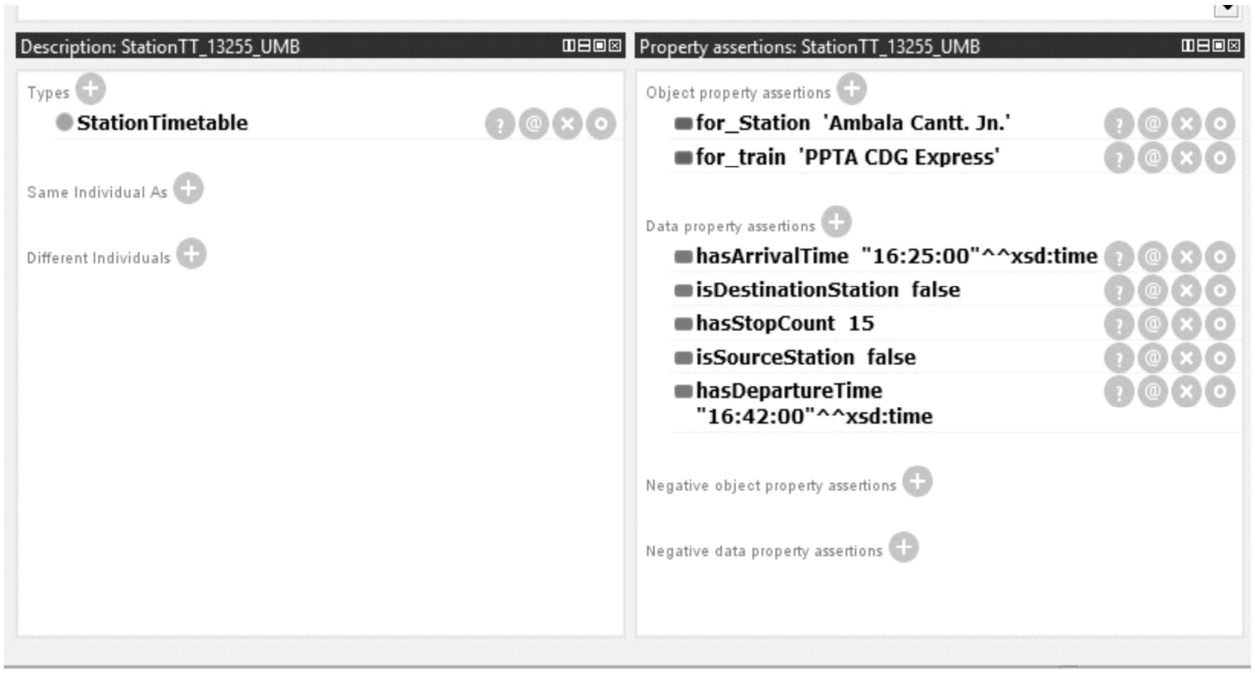
Object property assertions
for_train 'PPTA CDG Express'
for_Station Lucknow

Data property assertions
isSourceStation false
isDestinationStation false
hasStopCount 9
hasArrivalTime "06:15:00"^^xsd:time
hasDepartureTime "06:25:00"^^xsd:time

Negative object property assertions

Negative data property assertions

Figure 5: StationTimetable for 'PPTA CDG Express' for station Lucknow


Description: StationTT_13255_UMB

Types
StationTimetable

Same Individual As

Different Individuals

Property assertions: StationTT_13255_UMB

Object property assertions
for_Station 'Ambala Cantt. Jn.'
for_train 'PPTA CDG Express'

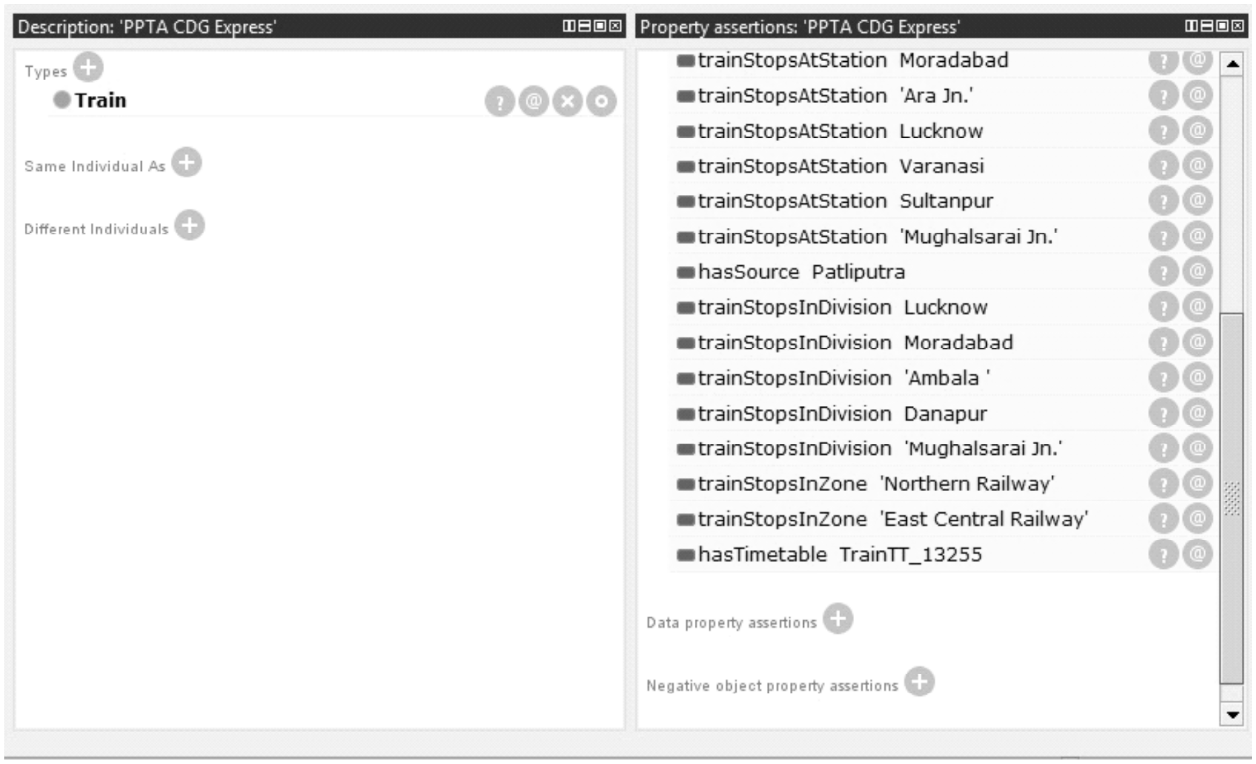
Data property assertions
hasArrivalTime "16:25:00"^^xsd:time
isDestinationStation false
hasStopCount 15
isSourceStation false
hasDepartureTime "16:42:00"^^xsd:time

Negative object property assertions

Negative data property assertions

Figure 6: Station Timetable for 'PPTA CDG Express' for station 'Ambala Cantt.Jn. '

Figure 7 shows the stations at which the train stops, along the information of the source, destination and timetable. This information has all been inferred by applying the SWRL rules on the existing values.


Description: 'PPTA CDG Express'

Types
Train

Same Individual As

Different Individuals

Property assertions: 'PPTA CDG Express'

trainStopsAtStation Moradabad
trainStopsAtStation 'Ara Jn.'
trainStopsAtStation Lucknow
trainStopsAtStation Varanasi
trainStopsAtStation Sultanpur
trainStopsAtStation 'Mughalsarai Jn.'
hasSource Patliputra
trainStopsInDivision Lucknow
trainStopsInDivision Moradabad
trainStopsInDivision 'Ambala '
trainStopsInDivision Danapur
trainStopsInDivision 'Mughalsarai Jn.'
trainStopsInZone 'Northern Railway'
trainStopsInZone 'East Central Railway'
hasTimetable TrainTT_13255

Data property assertions

Negative object property assertions

Figure 7: Inferred property values for 'PPTA CDG Express'

8. MAPPING TO UPPER ONTOLOGY

Upper ontologies provide a model that can be used across multiple domains. Merging the knowledgefrom various domains by mapping multiple domain ontologies to a single upper ontology helps in providinginteroperability and is useful to gain knowledge. BFO [20] is a widely used upper ontology.

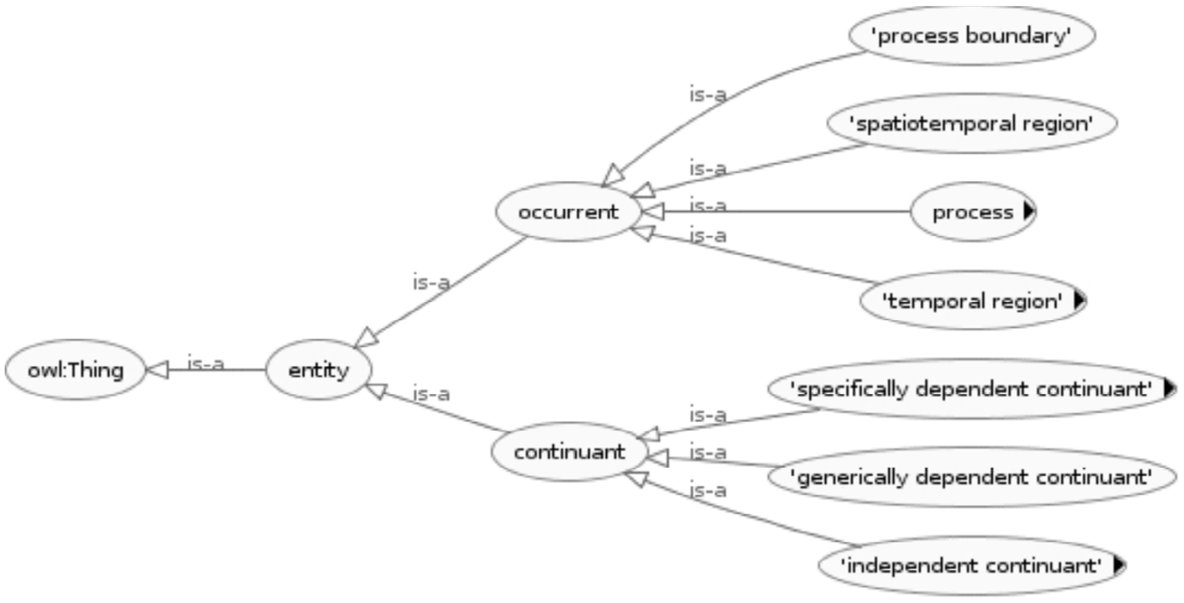


Figure 8: Structure of BFO

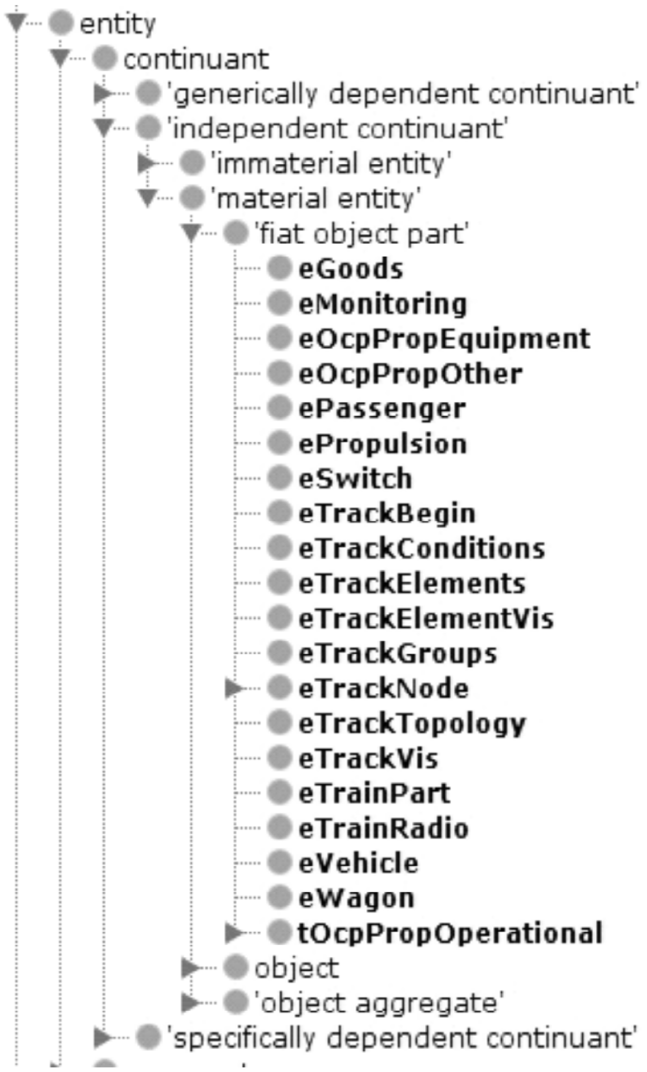
BFO provides a way to represent an entity based on two categories: continuant or substantial entityand *occurrent* or *processual* entity. A continuant is defined as an entity which continues to exist overtime and maintains is identity, whereas an occurrent is defined as an entity that can occur in time, suchas a process. The continuants are further divided into **'generically dependent'**, **'independent continuant'**and **'specifically dependent continuants'**. Similarly, occurrents are divided into **process**, **'processboundary'**, '**temporal region**' and **'spatio-temporal region**'. Figure 8 gives an overview of the structureof the BFO. The classes from the proposed railway ontology are mapped to the relevant classes in BFOdepending upon the usage of the classes and the entity they represent. Some of the classes from BFO aredefined below along with the classes that were mapped to it.

8.1. Continuant

'**generically dependent continuant'** : This class represents the entities which are generically dependenton other entities. An example of this is **eFormations** class which acts as a container for the eFormationclass and depends on the instances of **eFormation**. Thus, it is made a subclass of **'genericallydependent continuant'**. **eEquipmentUsage** class defines the usage of an equipment and depends on theinstances of the **eEquipment** class. Similarly, **eOperatingPeriodRef** depends on the **eOperatingPeriod**,making it a **'generically dependent continuant**'.

'continuant fiat boundary': This class in BFO is a subclass of the **'immaterial entity'**, which in turnis a subclass of the **'independent continuant'**. The **'continuant fiat boundary'** represents the boundaryof another entity (for example, the plane separating the northern and southern hemisphere of earth). Thesecan be divided into zero-dimensional, one-dimensional or two-dimensional fiat boundaries. Several classeswere made subclasses of the **'two-dimensional continuant fiat boundary'** class depending on their usage.These include **eMileageChanges**, which represents the imaginary boundary on the track where themileage system of the train changes. Similarly the classes **eGradientChanges**and **eRadiusChanges** respectivelyrepresent the boundaries where the gradient and radius or superelevation of the system changes.Some other classes are made the subclasses of the **'two-dimensional continuant fiat boundary'** classnamely, **tSpeedChange,tTrainProtectionChange, eAxleWeightChanges, eElectrificationChanges** etc.

'fiat object part' : This class in BFO is used to represent the entities which are a **continuant_part** ofanother entity. The **'fiat object part'** is demarcated from the rest of the entity using a 'two-dimensionalcontinuant fiat boundary'. A number of classes are derived from this class. For example, the **eTrackEnd**and **eTrackBegin** depends on **eTrack** class. Thus, **eTrackEnd** and **eTrackBegin** class are categorizedas the **'fiat object part'**. It also includes various other classes related similarly to **eTrack** such as**eTrackElements,eTrackGroups**etc are also grouped under the same category. An **eOcp** class representsan operation or control point. There are many classes which would represent the objects related toan **eOcp** instance. These classes are also made the subclasses of **'fiat object part'**. Some of these are**eOcpPropOperational**, **eOcpPropEquipmen**t which represent the operational properties of the ocp andthe equipment's at an ocp respectively. Other classes using 'fiat object part' as the base class include**eTrainPart**, **eTrainRadio**, **ePassenger** which are used to represent the individual train part, the type ofradio and passenger services of a system. Figure 9 shows the subclasses of 'fiat object part'.



```
entity
  continuant
    'generically dependent continuant'
    'independent continuant'
      'immaterial entity'
      'material entity'
        'fiat object part'
          eGoods
          eMonitoring
          eOcpPropEquipment
          eOcpPropOther
          ePassenger
          ePropulsion
          eSwitch
          eTrackBegin
          eTrackConditions
          eTrackElements
          eTrackElementVis
          eTrackGroups
          eTrackNode
          eTrackTopology
          eTrackVis
          eTrainPart
          eTrainRadio
          eVehicle
          eWagon
          tOcpPropOperational
        object
        'object aggregate'
    'specifically dependent continuant'
```

Figure 9: Classes mapped to 'fiat object part'

object: object is defined in BFO as a maximally causally unified material entity. It is used to representthe entities that can exist independent of any other entity. The classes such as **eOcp, eTrack, aBalise** fallunder this category. The instance of these classes does not require any other entity to exist. Some classeswhich are made subclasses of object are defined below:

— **aBalise:** represents the electronic beacon or transponders, which is used for automatic train protectionsystem. These can exist independent of any other entity.

— **eTrack:** represents a part of the railway track.

— **eOcp:** represents an independent operation or control point in a railway system.

— **eElectricalCoupler:** represents the electrical coupler (includes cables) between the vehicles.

— **eMechanicalCoupler:** It is the class for the mechanical couplers between the vehicles.

’object aggregate’: This class in BFO represents the material entities which consists of the plurals ofobjects as their members. Most of these classes are the groups of the object that are the defined as thesubclasses of object. Some of the subclasses of **‘object aggregate’** are as follows:

— **eBalises:** represents a group of aBalise.

— **eDerailers:** represent the collection of the derailers.

— **eBridges:** represents the collection of the bridges on the track.

Other classes among the same category are **eStorage, eStopPosts, eTrains, eTracks**etc.

’**specifically dependent continuant’**: This class in BFO represents the entities which are dependent onan **’independent continuant**’. The classes from the railway ontology that have been mapped as a subclass

of this class are:

— **eBlockPartSequence:** represents a number of blocks, arranged in a sequence of tasks. It depends onthe blockParts that it refers.

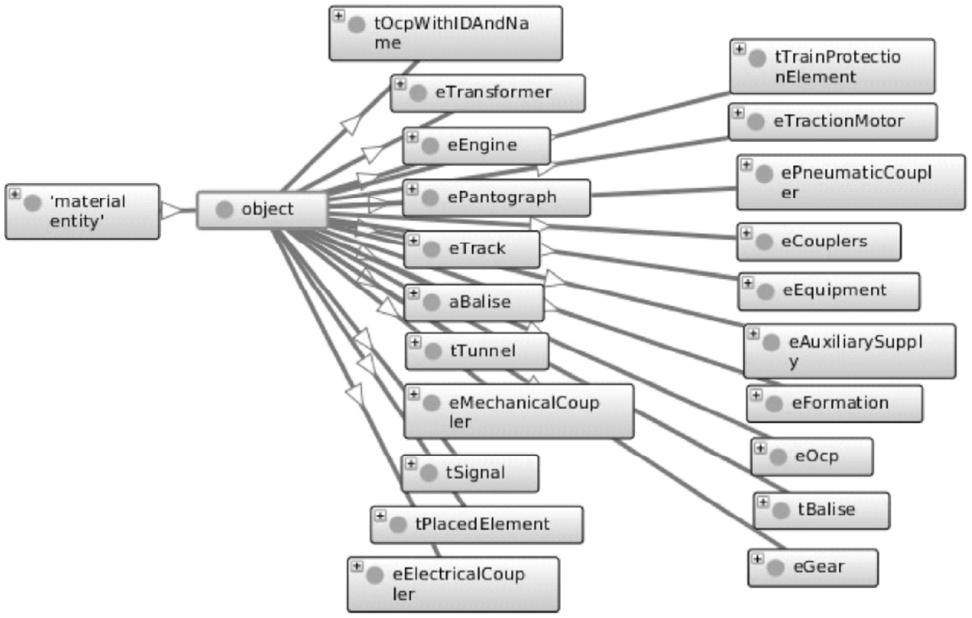


Figure 10: Classes mapped to object in BFO

— **eOrganizationUnitBinding:** bind the data of any infrastructure to the owner. It depends on the infrastructureor rollingstock part.

— **SectionTT:** describes data concerning to one ocpTT with respect to another, making it an**’specificallydependentcontinuant’**.

The **quality** subclass of **’specifically dependent continuant**’ represents the quality or attribute of anentity which exist for all the time at which the entity itself exists.

Below are some of the classes that weremapped as the subclass of quality.

— **eAnnotation:** used to represent the metadata of any infrastructure or rollingstock part.

— **eCategory:** describes the train type or category. For example passenger, goods or mixed.

— **eLoadLimit:** maximum payload of the vehicle, which remains same for its lifetime. quality alsoconsists of several other classes including eVehicleBrake, eTractionInverter, ePulsePattern etc.

8.2. Occurrent

’one-dimensional temporal region’: This class represents the temporal region where a process occurs.This contains the classes relating to the timetables. Some of the classes that are derived from this class are as follows:

– **aArrivalDepartureTimes**: represents the time for arrival and departure of a train.

– **eOcpTT**: represent the times of arrival and departure of a train at a given ocp.

– **eOperatingPeriod**: represents the operating period of a train. It can be in relation to certain days orabstract based on a standard week.

Other classes in the same category include **eRostering**, **eOperatingDay**, **eTimetablePeriod** etc.

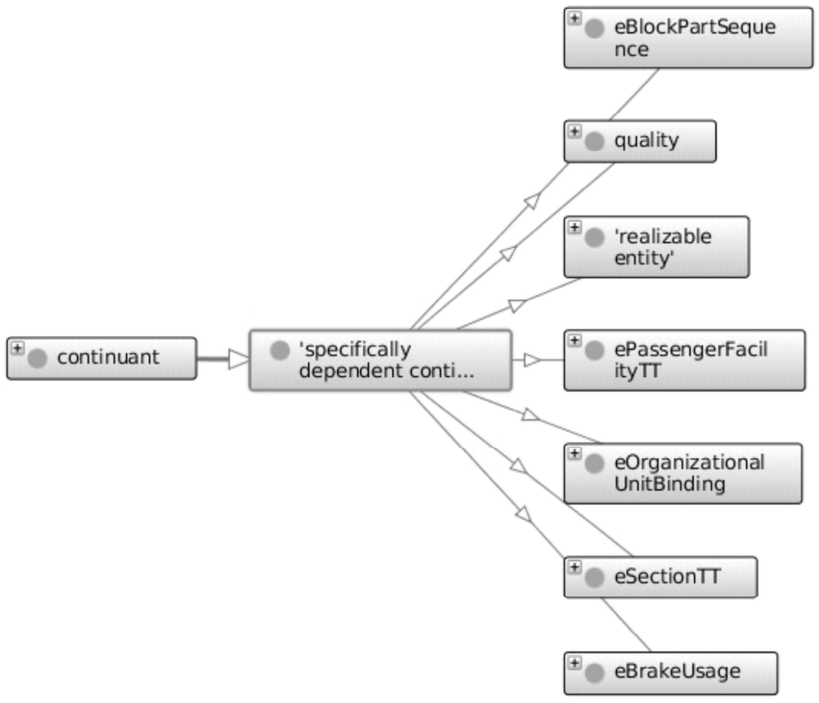


Figure 11: Classes mapped as subclass of ’specifically dependent continuant’

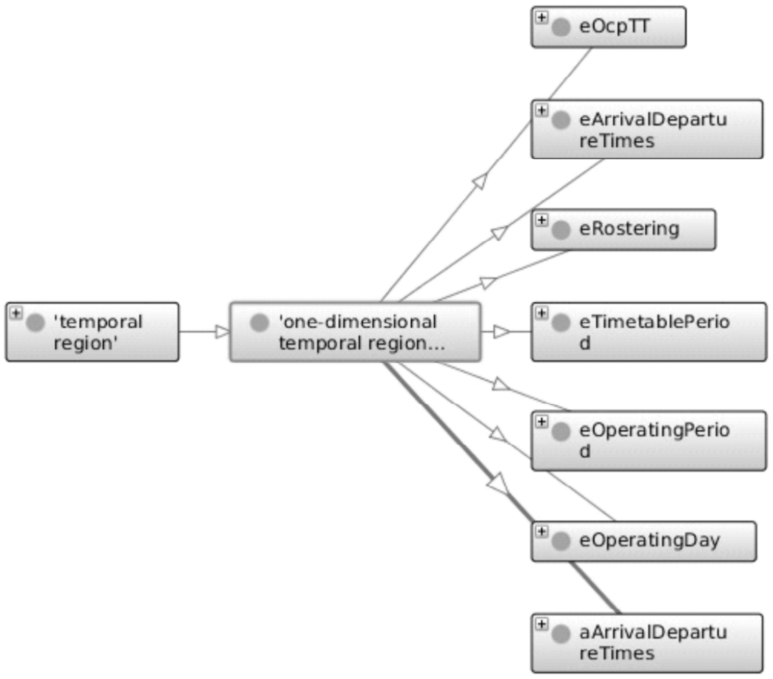


Figure 12: Classes mapped as subclass of 'one-dimensional temporal region'

9. BENEFITS AND USE CASES

Integrating the data from all across the domain is surely going to help both the users and the developers.Using an ontology adds even more benefits to the work. By using the ontology, not only the data isintegrated, but the domain itself moves a step forward towards a semantic web. The use of ontology helpsto create a knowledge base, which can be used by other technologies to gain even more knowledge whichwill eventually make the entire web connected semantically. In addition to this, there are several otherbenefits of using ontology against any other technology such as XML.

Following are some of the advantages of using ontology:

— Sharing common understanding of the structure of information among people or software agents.

— Enable reuse of domain knowledge

— To make domain assumptions explicit.

— Reasoning is supported because of existence of description logic. Thus, the models and data descriptionscan be checked for inconsistencies and also for inferring new knowledge from the givenknowledge.

— It has different sub-languages with different levels of expressive power and inferential complexity.One of these two features can be compromised to increase the efficiency of the other. This helps indesigning the models according to the need of the user.

— XML/UML would only integrate the data. Ontology also adds the semantics to the data and makesthe hidden and implicit data explicit.

— Another advantage is that the OWL can be structured using XML. This is very helpful, especially incases where data models are shared between multiple applications.

9.1. Benefits of using railway ontology

There can be several entities which would benefit from the railway ontology.

9.1.1. Benefit to operators/organization

— **Track management**: Maintenance of tracks can be predicted or adjusted according to the schedulesof the trains on a particular route. Trains can be re-routed if some maintenance work is in progress.

— **Rollingstock**: Using the current GPS systems together with the knowledge from ontology, therollingstock can be easily monitored. The capacity of a vehicle, its location and current status, suchas under maintenance or attached to a train, in transit or stationary can be monitored.

— **Infrastructure**: Combining all the information, it can be predicted whether the infrastructure is satisfyingthe needs at a particular place or not. For example, if 50,000 unreserved tickets are issuedeveryday from a station, and the total seating capacity of all the trains stopping at that station isonly 20,000, then the tools can be used to give an alert to the concerned authorities. Developers canbuild the tools which can use this information from the ontology and notify whenever a pre-definedsituation, like the one explained above, occurs.

— **Predictive maintenance of rollingstock and infrastructure**: The dates and parts on which the particularvehicle/infrastructure had undergone maintenance can be monitored. Alerts can be sent whena pre-defined time has elapsed, or a situation is detected where these parts maybe affected.

— Software Tools that may be developed can also monitor the timetable and the passengers acrossvarious stations, or maybe the entire system as a whole. Thus, using the knowledge from ontologyand the rules defined in the tools, a notification can be generated to suggest new trains, or re-routethe trains or something else depending upon the situation. This can also be used to monitor runningstatus of the trains, whether they are running on schedule or not. The tool should be able to givesuggestion in case of any adverse situations.

— **Software agents:** The ontology can be used to develop software agents, which can be deployedwhere humans may not reach.

— **Decision Support System (DSS):** A DSS is one of the major advantage that the ontology has overthe integration technologies.

9.1.2. Benefits to other users

— **Journey planning**: The timetable data from the ontology can be combined with the other transportmodes, which can then be used to plan journeys.

— Commuters can easily know which trains would stop at a given station, or which train is most likelyto be overcrowded or when is the last train for the day. Information like this would really help theuser to plan a journey.

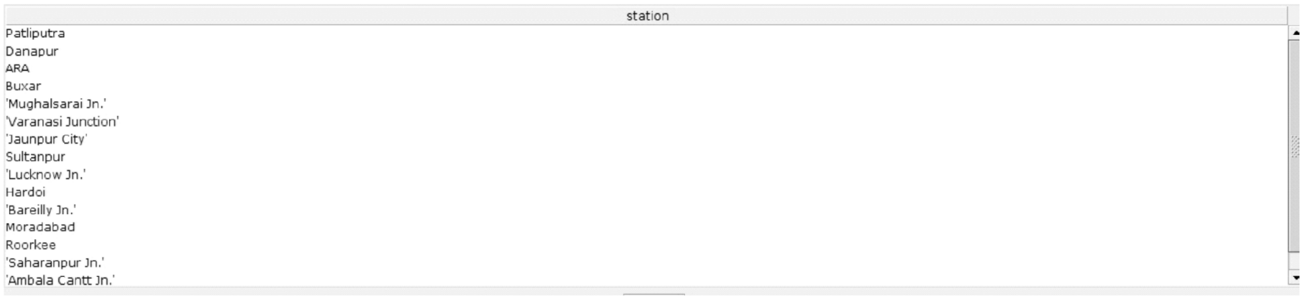
9.2. Use cases

This section describe about the sample query executed on the Railway Ontology for retrieving the route of train , train between station finding direct train between stations and route where direct train are not available. SPARQL has been used for retrieving information from the ontology data model

9.2.1. To retrieve the route of a train

The following query retrieves the route or stoppage of a particular train

```
SPARQL Query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ir: <http://www.semanticweb.org/indianRailway#>
SELECT ?station
WHERE {
?train a ir:Train.
FILTER (?train = ir:Train_13255).
?train ir:hasTimetable ?timetable.
?timetable ir:hasStations ?stations.
?stations ir:hasStopCount ?x.
?stations ir:forStation ?station. } ORDER BY ?x
```



```
station
Patliputra
Danapur
ARA
Buxar
'Mughalsarai Jn.'
'Varanasi Junction'
'Jaunpur City'
Sultanpur
'Lucknow Jn.'
Hardoi
'Bareilly Jn.'
Moradabad
Roorkee
'Saharanpur Jn.'
'Ambala Cantt Jn.'
```

Figure 13: Route of a train

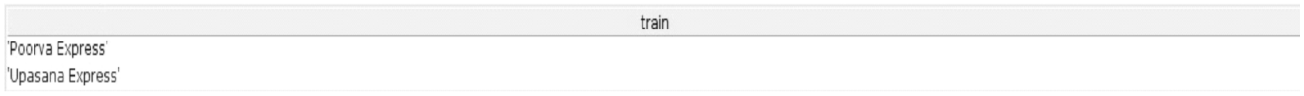
Figure 13, is the output of the SPARQL query executed for the route of train no. 13255.

9.2.2. Trains from station A to station B

This query can divided into two parts, namely (a) first query will look for direction trains between two stations and (b) second query will try to find out intersection points of the train which will help in journey between the station where direct train are not available

a. When direct trains are available

```
SPARQL Query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ir: <http://www.semanticweb.org/indianRailway#>
SELECT ?train
WHERE {
?station1 a ir:Station .
?station2 a ir:Station.
FILTER(?station1=ir:st_HWH).
FILTER (?station2=ir:st_PNBE).
?station1 ir:isStationInDivision ?div1.
?station2 ir:isStationInDivision ?div2.
?div1 ir:isDivisionInZone ?zone1.
?div2 ir:isDivisionInZone ?zone2.
?train a ir:Train.
?train ir:trainStopsInZone ?zone1.
?train ir:trainStopsInDivision ?div1.
?train ir:trainStopsAtStation ?station1.
?train ir:trainStopsInZone ?zone2.
?train ir:trainStopsInDivision ?div2.
?train ir:trainStopsAtStation ?station2.
}
```



```
train
'Poorva Express'
'Upasana Express'
```

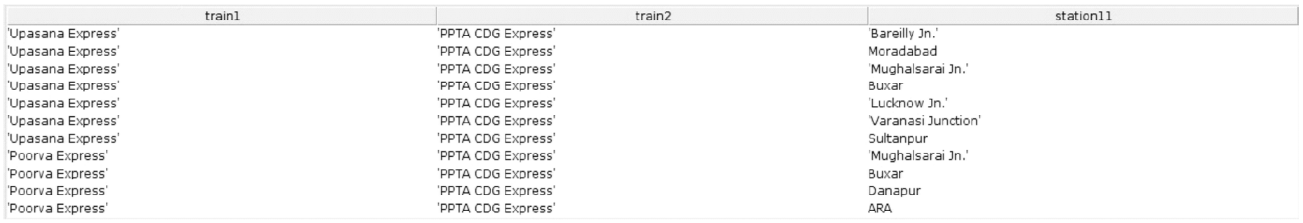
Figure 14: Output of SPARQL Query for finding train from station A (HWH) to station B (PNBE)

b. When direct trains are not available:

SPARQL Query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ir: <http://www.semanticweb.org/indianRailway#>
SELECT DISTINCT ?train1 ?train2 ?station11
WHERE
?station1 a ir:Station .
?station2 a ir:Station.
FILTER(?station1=ir:st_HWH).
FILTER (?station2=ir:st_CDG).
?station1 ir:isStationInDivision ?div1.
?station2 ir:isStationInDivision ?div2.
?div1 ir:isDivisionInZone ?zone1.
?div2 ir:isDivisionInZone ?zone2.
?train1 a ir:Train.
?train1 ir:trainStopsInZone ?zone1.
?train1 ir:trainStopsInDivision ?div1.
?train1 ir:trainStopsAtStation ?station1.
?train2 a ir:Train.
```

```
?train2 ir:trainStopsInZone ?zone2.
?train2 ir:trainStopsInDivision ?div2.
?train2 ir:trainStopsAtStation ?station2.
?train1 ir:trainStopsInZone ?zone11.
?train2 ir:trainStopsInZone ?zone21.
FILTER (?zone11=?zone21).
?train1 ir:trainStopsInDivision ?div11
?train2 :trainStopsInDivision ?div21.
FILTER (?div11=?div21).
?train1 ir:trainStopsAtStation ?station11.
?train2 ir:trainStopsAtStation ?station21.
FILTER (?station11=?station21).
```



train1	train2	station11
'Upasana Express'	'PPTA CDG Express'	'Bareilly Jn.'
'Upasana Express'	'PPTA CDG Express'	Moradabad
'Upasana Express'	'PPTA CDG Express'	'Mughalsarai Jn.'
'Upasana Express'	'PPTA CDG Express'	Buxar
'Upasana Express'	'PPTA CDG Express'	'Lucknow Jn.'
'Upasana Express'	'PPTA CDG Express'	'Varanasi Junction'
'Upasana Express'	'PPTA CDG Express'	Sultanpur
'Poorva Express'	'PPTA CDG Express'	'Mughalsarai Jn.'
'Poorva Express'	'PPTA CDG Express'	Buxar
'Poorva Express'	'PPTA CDG Express'	Danapur
'Poorva Express'	'PPTA CDG Express'	ARA

Figure 15: Result of SPARQL Query for trains from station A to station B showing pair of trains and intersecting stations

Figure 14 and 15 show the trains between two stations, including the stations where passenger canchange trains (when direct train is not available).

10. CONCLUSION

The use of ontology as an integration technology is a relatively new concept, as compared to the othertechnologies. On one hand, it integrates the data like previous technologies. While on the other hand, itsupports description logic, which provides supports for reasoning. Thus an ontology is able to provide boththe features at the same time. Though sometimes the expressive power of the ontology is compromised asa trade-off for gaining reasoning efficiency, it is still a good option if the knowledge gain is much moreessential.

The ontology developed in this project should be able to satisfy the needs of major railway organizationacross the world. The ontology might need to be customized in order to work properly with the existingsystems. The software industry can also try to develop software agents using this ontology to automate thevarious tasks carried out in the railways. This will further improve the efficiency of the complete system.

11. FUTURE SCOPE

As a future work,besides developing software agents, the mapping of the ontology to an upper ontology,such as DUL (Dolce-ultra-lite) is suggested to integrate the knowledge from other fields and share it to othersemantically connected entities. Improvements on current ontology, such as suggesting and/or planningjourney using the operating period of the trains can be implemented.

The use cases in section 9 have been given for simple tasks on timetable data. These can be enhancedby adding more information to the ontology in other categories of the railway.

REFERENCES

[1] Verstichel, S., Ongenae, F., Loeve, L., Vermeulen, F., Dings, P., Dhoedt, B., ...& De Turck, F. , "Efficient data integration in the railway domain through an ontology-based methodology". *Transportation Research Part C: Emerging Technologies*, *19*(4), pp.617-643, 2011.

[2] Indian Railways. *www.indianrailways.gov.in*. Date Accessed: 10th August 2016.

[3] Noy, N. F., & McGuinness, D. L.," Ontology development 101: A guide to creating your first ontology", 2001.

[4] Shingler, R., &Umiliacchi, P. ," Advances in railways maintenance: the EuRoMain project". In *Proceedings of the WCRR, World Conference on Railway Research,* 2003.

[5] Umiliacchi, P., Shingler, R., Langer, G., & Henning, U.,"A new approach to optimisation through intelligent integration of railway systems: the InteGRail project". In *Proceedings of the 7th WCRR, World Conference on Railway Research,* 2006.

[6] InteGRail Consortium.,"Integrail, intelligent integration of railway systems". *InteGRail Consortium,* 2009.

[7] Nash, A., Huerlimann, D., Schütte, J., & Krauss, V. P., "RailML† A Standard Data Interface For Railroad Applications". *WIT Transactions on The Built Environment*, 74, 2004.

[8] Bohring, H., & Auer, S., " Mapping XML to OWL Ontologies". *LeipzigerInformatik-Tage*, *72*, pp.147-156, 2005.

[9] Yahia, N., Mokhtar, S. A., & Ahmed, A. , "Automatic generation of OWL ontology from XML data source". *arXiv preprint arXiv:1206.0570*, 2012.

[10] Bedini, I., Matheus, C., Patel-Schneider, P. F., Boran, A., & Nguyen, B.,"Transforming XML schema to OWL using patterns". *Fifth IEEE International Conference onSemantic Computing (ICSC),*pp. 102-109, 2011.

[11] Wheeler, A., Dike, J., &Winburn, M. , "XSD To OWL: A Case Study".

[12] Ghosh, S., Dutta, A., &Alam, M. A.,"Multi-agent based railway track management system."*3rd InternationalonAdvance Computing Conference (IACC),* pp. 1408-1413, 2013.

[13] Umiliacchi, P., Lane, D., Romano, F., &SpA, A., "Predictive maintenance of railway subsystems using an Ontology based modelling approach". In *Proceedings of 9th world conference on railway research,* pp. 22-26, 2011.

[14] Saa, R., Garcia, A., Gomez, C., Carretero, J., & Garcia-Carballeira, F., "An ontology-driven decision support system for high-performance and cost-optimized design of complex railway portal frames". *Expert Systems with Applications*, *39*(10), pp. 8784-8792, 2012.

[15] Lewis, Richard, Florian Fuchs, Michael Pirker, Clive Roberts, and Gerhard Langer. "Using ontology to integrate railway condition monitoring data." (2006): 149-155.

[16] Berners-Lee, T., Hendler, J., &Lassila, O. ,"The semantic web". *Scientific american*, *284*(5), pp. 28-37, 2001.

[17] Harth, A., Janik, M., &Staab, S.,"Semantic web architecture". In *Handbook of Semantic Web Technologies*. Springer Berlin Heidelberg.pp. 43-75, 2011

[18] Antoniou, Grigoris, and Frank Van Harmelen."A semantic web primer". MIT press, 2004.

[19] Horridge, M., &Bechhofer, S.,"The owl api: A java api for owl ontologies."*Semantic Web*, *2*(1), pp.11-21, 2001.

[20] Smith, B., &Grenon, P., "Basic formal ontology". *Draft. Downloadable at http://ontology. buffalo. edu/bfo*, 2002.