ISSN:2229-6093

Constructing Railway Ontology using Web Ontology Language and Semantic Web Rule Language

A. Raja Mohan
Associate Professor
Department of Computer Science
Madurai Kama raj University
Madurai.
prithivimohan@gmail.com

Dr.G.Arumugam
Professor and Head
Department of Computer Science
Madurai Kama raj University
Madurai.
gurusamyarumugam@gmail.com

Abstract

In the traditional railway information system, the users have to create many queries to retrieve specific train information. The results of the queries are based on the syntactic nature of data. A semi-illiterate human being finds difficulty in reading the railway time table. This problem can be effectively handled through semantic technology. Semantic web, future web, will revolutionize the world with machine knowledge processing capabilities. Ontology is the building block of the semantic web. Earlier research on ontology design methodologies shows that manual construction of ontology is a complex process and it is very hard for a designer to develop a consistent ontology. In this paper, we present a methodology based on the usage of protégé 3.4.4 for developing and inferring the railway ontology. This paper describes the features of Protégé 3.4.4, the usage of inference rules framed by using SWRL, query by SQWRL and the inconsistency checking by Pellet reasoner.

Key Words: *Ontology, OWL*, Protégé, *Railway, Semantic Web, WWW*.

1. Introduction

Indian Railway's network is the second largest railway network in the world. For easy operation and management, Indian Railway is divided into fifteen zones and each zone manages its own local affairs. Southern railway is one among the fifteen zones of Indian railway. If a person wants to travel from one place to another place by train, he or she has to plan for the shortest route, minimum traveling time or cost effective route and the details of the trains operating in the identified route for the comfort of his journey. A new form of web content that
is meaningful to computers will unleash a revolution of new abilities. The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in co-operation [1]. This is mainly accomplished by the use of ontologies which contain terms and relationships between terms that have been agreed upon by members of a certain domain. These agreed upon ontologies can then be published to be available for use by other members of the domain. Typical ontologies are : Agrovoc, http://www.fao.org/agrovoc/, Dublin Core, http://dublincore.org/, Open Biological Ontologies, http://obo.sourceforge.net/, Upper Cyc Ontology, http://www.cyc.com/.

2. Representation of railway domain knowledge

Knowledge representation is the symbolization or formalization of the knowledge of the world. Various Knowledge representation methods are predicate logics, production rules, framework, ontology and so on. In this paper, We use the ontology and the rules to represent the knowledge of railways. An ontology is an engineering artifact that is constituted by a specific vocabulary used to describe a certain reality (domain), plus a set of explicit assumptions regarding the intended meaning of the vocabulary. Insufficiency in expressivity and reasoning features are the problems affect the perfectness of ontologies. Rules can lay the foundation for expression and reasoning capabilities. Adding rules with the ontologies will solve the expressivity and reasoning problems.

2.1. Ontology and its need

An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary[2]. It is necessary to define web resources more precisely and make them more amenable to machine processing. Ontology is classified as top level ontology, domain ontology, task or activity ontology and application ontology[3]. Top level ontology describes the very general concepts which are independent of a particular domain. Domain ontology describes the vocabulary related to a generic domain by specializing the concepts introduced in the top level ontology. Task ontology describes the vocabulary related to a generic task by specializing the concepts introduced in the top level ontology. Application ontology describes the concepts which corresponds to roles played by domain entities while performing a certain tasks. The reasons for developing ontology are to analyze the domain knowledge, to enable the reuse of domain knowledge, to make the domain assumptions explicitly, to separate the domain knowledge from the operational knowledge and to share the common understanding of the structure of information among the people or the software agents.

2.1.1. Railway ontology. The railway ontology is defined as a pentad, SRO = <RC, RP, RI, RR, RA>
where SRO-ontology is the ontology which describes the concepts and their relations in the railway domain, RC is a collection of concepts, RP is a collection of attributes related to the concepts in RC, RI is a collection of individuals or instances of the concepts in RC, RR is a collection of relations between the concepts in RC, RA is the collection of axioms which are used to restrict the attributes and relations. An ontology can be constructed manually [4] or semi-automatically [5]. Manual extraction has been done for railway ontology.

2.1.2. Steps for designing ontology. The fundamental rules for designing the ontology are 1) There is no one correct way to model a domain— there are many alternatives. The best solution always depends on the application, 2) Ontology development is necessarily an iterative and a dynamic process [6] and 3) Concepts in the ontology should be close to objects in the domain of interest.
The steps to be followed for designing the ontology[7]
1. Determine the domain and scope of the ontology.: Developers should determine the domain of interest and define the scope of the ontology.
2. Consider the reuse of existing ontology.: Developers should reuse if ontology exist, otherwise develop it.
3. Enumerate important terms in the ontology.: Developers should write down a list of all terms in the domain, either to make statements about or to explain to a user.
4. Define the classes and the class hierarchy.: Classes has to be defined with appropriate names. The hierarchies has to be defined in three different ways, viz. top-down, bottom-up, hybrid. A top-down approach starts with the most general classes in the domain and subsequent specialization of the classes. A bottom-up approach starts with the most specific classes, the leaves of the hierarchy, with subsequent grouping of these classes into more general classes. A hybrid approach is a combination of the top-down and bottom-up approaches
5. Define the properties of classes.: Properties describe the internal structure of classes. The object property relates one class to another class. The data type property assign the value type to the class. The annotation property describes the generic comments of the class.
6. Define the facets of the properties.: Properties can have different facets describing the value type, allowed values, the number of the values (cardinality), and other features of the values, the properties can have.
7. Create instances.: At last create the individuals or instances of the classes.

2.1.3. Ontology tools. Ontology editors are tools that enable the users for inspecting, browsing, codifying, and modifying ontology and support in this way the ontology development and maintenance task [8]. Existing editors vary in the complexity of the underlying knowledge model, usability, scalability, etc. Nevertheless, all of them provide enough support for the initial ontology development. Prot´eg´e [9], OntoEdit [10], OilEd [11], WebODE [12], and Ontolingua [13] are some examples of ontology editors.

2.2. Protégé

Protégé is a free, open-source platform that provides a growing user community with a suite of tools to construct domain models and knowledge-based applications with ontologies. Further, Protégé can be extended by way of a plug-in Architecture and a Java-based Application Programming Interface for building knowledge-based tools and applications.

2.2.1. Protégé editor. Protégé-OWL editor enables the users to Load and save OWL and RDF ontologies, Edit and visualize classes, properties, and SWRL rules, Define logical class characteristics as OWL expressions, Execute reasoners such as description logic classifiers and Edit OWL individuals for Semantic Web markup.

2.2.2. SWRL. Semantic Web Rule Language, an acronym for SWRL, is a rule language. The SWRLTab is a development environment for working with SWRL rules in Protege-OWL. It supports the editing and execution of SWRL rules. It provides a set of libraries that can be used in rules, including libraries to interoperate with XML documents, and spreadsheets, and libraries with mathematical, string, RDFS, and temporal operators. The SWRLTab has several software components, like, SWRL Editor which supports editing and saving of SWRL rules in an OWL ontology, SWRL Built-in Libraries which includes the core SWRL built-ins defined in the SWRL Submission and built-ins for querying OWL ontology.

2.2.3. SQWRL. Semantic Query enhanced Web Rule Language is an extension of SWRL rule language [14]. It takes rules' antecedent as a pattern specification for a query and takes rules' consequent as a retrieval specification. Any valid SWRL antecedent is a valid SQWRL pattern specification. That means, SQWRL places no restriction on the left side of a query. It uses SWRL's built-in libraries as an extension point [15]. SWRL editor can be used to generate and edit the queries. To execute the query SQTab has to be added [16], by adding the jar file jess 7.1p2 in the protege plug-in directory. The core operator is aqwrl:select. The select operator takes one or more arguments, which are variables in the pattern specification of the query, and builds a table using the arguments as the columns of the table. The built-ins like, sqwrl:count, sqwrl:orderBy, sqwrl:avg can be used as in SQL. SQWRL does not support subqueries, but it is achieved by using the intermediate inferences made by SWRL rules. This mechanism is used to decompose the complex queries. The sqwrl:makeSet built-in is used to create a set. Using this set and sqwrl built-ins like, union, difference, isEmpty, size, intersection are used to do set operations. The SQWRL query tab provides a graphical interface to display the results of SQWRL queries.

3. SRO-Ontology structure

The classes are the building blocks of ontology. Classes describe the concepts of the domain. The concepts of this domain are the types of the trains such as mail/express, rajdhani, shatabdi and the fare details [17]. A class can have subclasses that represent concepts that are more specific than the super class. In railway ontology, Southern_Railway is a subclass of owl:Thing. Fare_Details and Train_Schedule are subclasses of Southern_Railway. List_of_Trains is a subclass of Train_Schedule. The subclasses of List_of_Trains are Mail_Express_Trains, Shatabdi_Trains, Rajdhani_Trains. The Subclasses of Mail_Express_Trains are TrainNo0601, TrainNo0602, TrainNo1063, TrainNo1064, TrainNo2605, TrainNo2606, TrainNo2632, TrainNo2631, TrainNo2633 and TrainNo2634. The Subclasses of Rajdhani_Trains are TrainNo2269, TrainNo2270, TrainNo2429, TrainNo2430, TrainNo2432, TrainNo2433 and TrainNo2434. The Subclasses of Shatabdi_Trains are TrainNo2007, TrainNo2008, TrainNo2027 and TrainNo2028. The subclasses of Fare_Details are AC_Chair_Car, First_Class, First_Class_AC, Second_Class_AC, Second_Seating, Sleeper_Class and Third_Class.

The properties can have different facets describing the value type, allowed values, the number of the values and other features of the values that the property can take. The Data type properties in SRO_ontology are hasArrivalTime, hasDay, hasDepartureTime, hasDestinationStation, hasDistance, hasRouteNo, hasRunsOn, hasSourceStation, hasStationCode, hasStationName, hasTrainName and hasTrainNo. The object property has domain and range. The classes to which property is described is called the domain of the property. The allowed classes or type instances for the property are often called as range of the property. The Object properties in SRO_ontology are hasACChairCar_Fare, hasFirstClass_Fare, hasFirstClassAC_Fare, hasSecondClassAC_Fare, hasThirdClassAC_Fare, hasSleeperClass_Fare, hasSecondSeating_Fare, hasGeneralInformations and hasRunningInformations. The last step is creating individuals or instances of classes in the hierarchy. Defining an individual of a class requires i. choosing a class, ii. Creating an individual of that class and iii. Filling in the property values. The typical individuals are TN0601, RI0601_S01, RI0601_S02, RI0601_S03 and General_Information_of_0601.

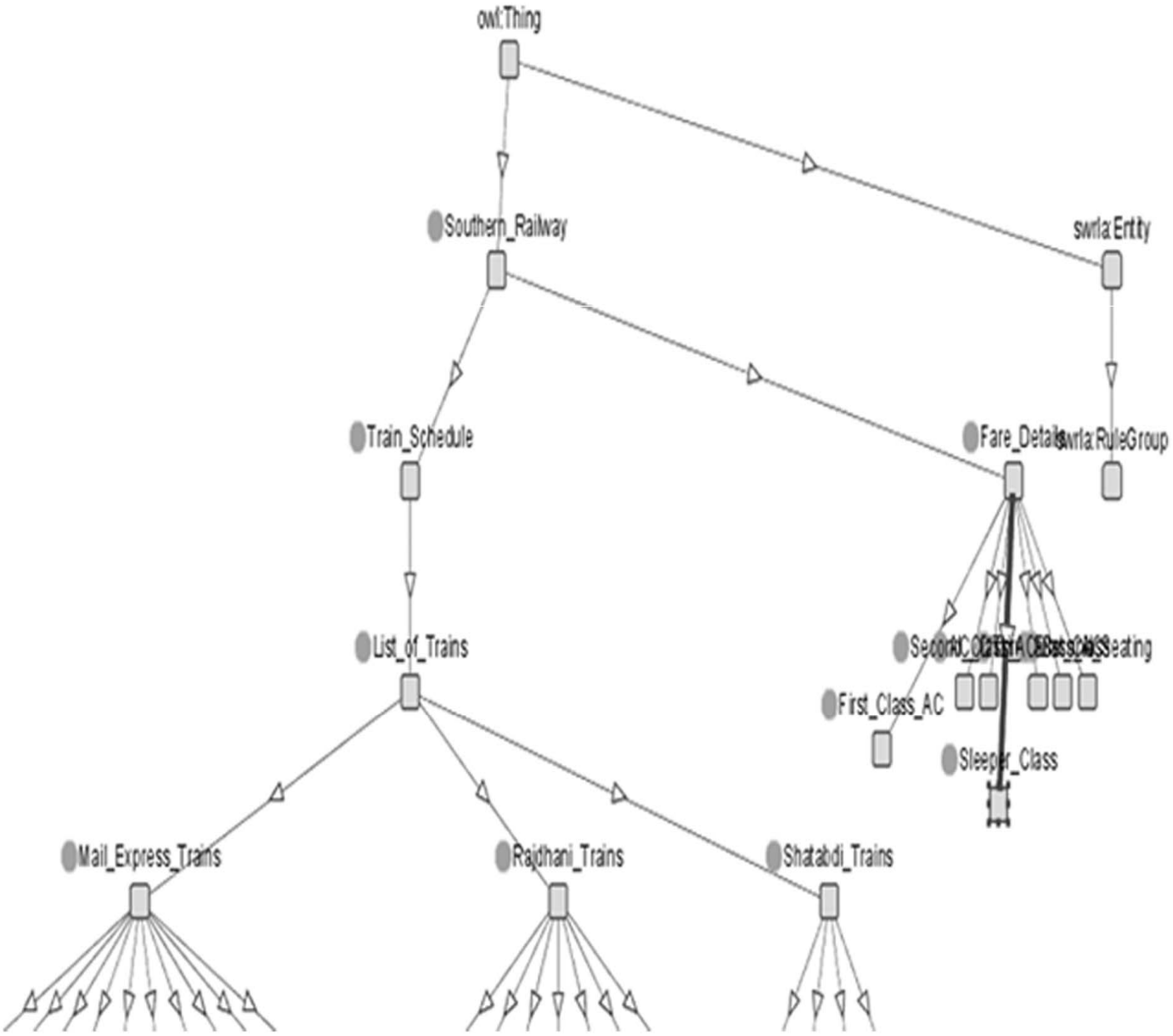

Figure 1 . Class tree of the SRO-ONTOLOGY

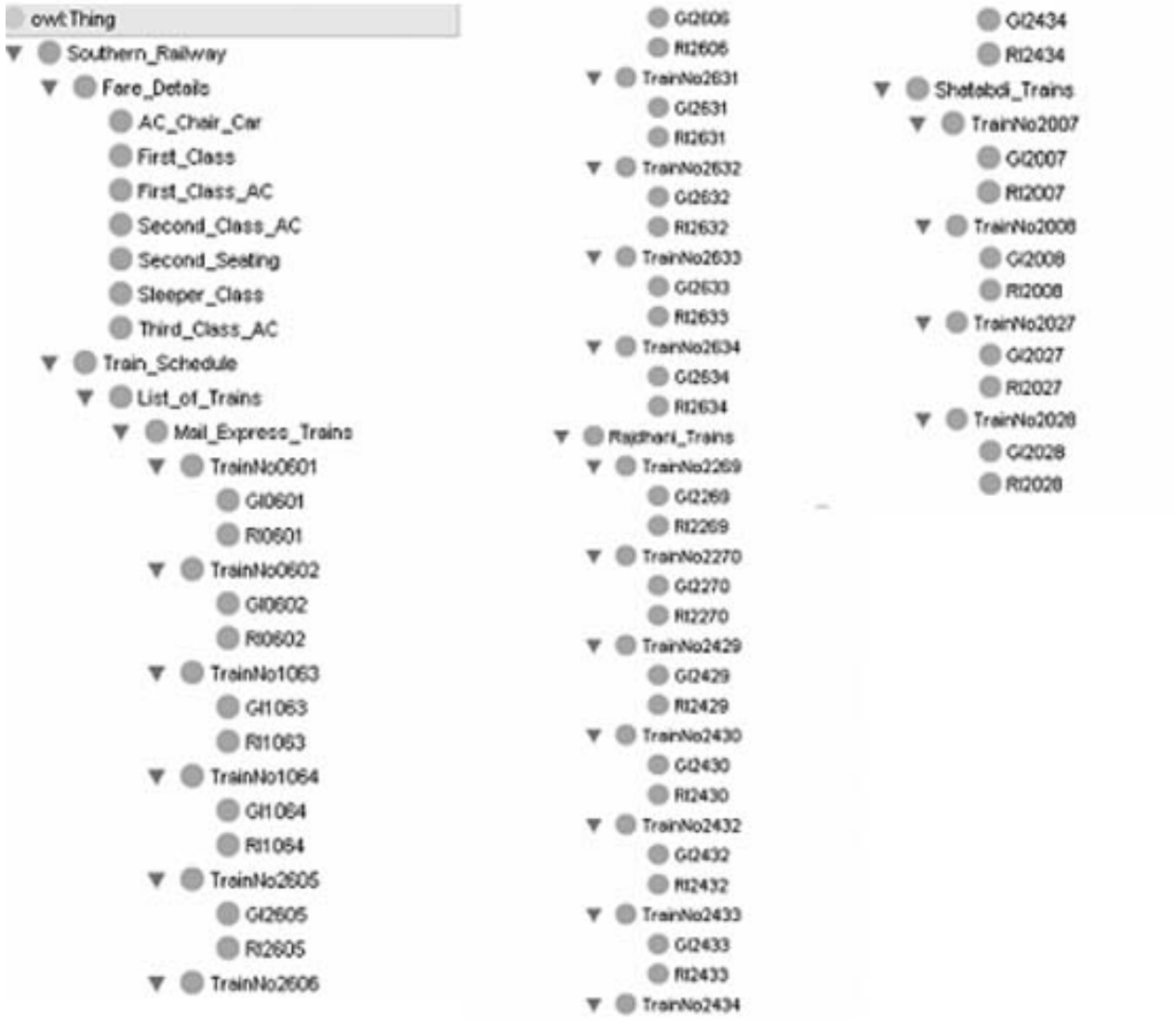

Figure 2 Subclasses of the Class Southern Railway

3.1. SRO-Ontology based reasoning

In our work, Pellet reasoner[18] is used to check the consistency of the ontology, classifying the classes and performing the individual detection etc. checking the consistency is used to identify the semantic contradiction which makes ambiguity in the description of the domain. Classifying the classes is used to test the subsumption relationship between classes and classes are classified into
different level in the class hierarchy. Reclassification is necessary if there exist inconsistency. If Individual detection mechanism determines the appropriate concepts for certain individuals and classify the individuals according to the classes.

3.2. Rule based reasoning

In this work, swrl is used for creating rules and sqwrl is used for supporting OWL queries.

For example, The following query retrieves the running information of particular train
RI0601(?x) ^ hasStationCode(?x, ?y) ^ hasStationName(?x, ?z) ^ hasArrivalTime(?x, ?a) ^ hasDepartureTime(?x, ?b) ^ hasDistance(?x, ?c) ^ hasDay(?x, ?d) → sqwrl:select(?y, ?z, ?a, ?b, ?c, ?d) ^ sqwrl:orderBy(?c) ^ sqwrl:columnNames("StationCode",

"StationName", "RouteNo", "ArrivalTime", "DepartureTime", "Distance", "Day")

To display the arrival time of given train name and station name, We can write the query
GI0601(?x) ^ hasTrainName(?x, "NAGERCOIL EXP") ^ hasGeneralInformations(?y, ?x) ^ hasRunningInformations(?y, ?z) ^ hasStationName(?z, "KARUR") ^ hasArrivalTime(?z, ?a) → sqwrl:select(?a) ^ sqwrl:columnNames("Arrival Time")

To display the train name by giving station name, departure time and destination station, We can query
hasStationName(?x, "ARAKKONAM") ^ hasDepartureTime(?x, 21:25:00") ^ hasRunningInformations(?y, ?x) ^ hasGeneralInformations(?y, ?a) ^ hasDestinationStation(?b, "NAGERCOIL JN") ^ hasTrainName(?b, ?c) → sqwrl:select(?c)

The query for displaying the traveling time from Chennai central to Jolarpettai by Nagercoil exp is
GI0601(?x) ^ hasTrainName(?x, ?y) ^ hasSourceStation(?x, ?x1) ^ hasRunsOn(?x, ?a) ^ hasDepartureTime(?x, ?d) ^ hasGeneralInformations(?z, ?x) ^hasRunningInformations(?z, ?b) ^ hasStationCode(?b, "JTJ") ^ hasStationName(?b, ?b1) ^ hasArrivalTime(?b, ?c) ^ swrlb:subtractTimes(?d1, ?d, ?c) → sqwrl:select(?y, ?x1, ?a, ?d, ?b1, ?c, ?d1) ^ sqwrl:columnNames("TrainNames", "SourceStation", "RunsOn", "Departure Time", "Destination Station", "ArrivalTime", "Tot Time")

The query for retrieving the arrival time at destination of Nagercoil express which runs on Friday is

GI0601(?x) ^ hasTrainName(?x, ?y) ^ hasSourceStation(?x, ?x1) ^ hasRunsOn(?x, ?a) ^ hasDepartureTime(?x, ?d) ^ hasGeneralInformations(?z, ?x) ^ hasRunningInformations(?z, ?b) ^ hasStationCode(?b, "NCJ") ^ hasStationName(?b, ?b1) ^ hasArrivalTime(?b, ?c) → sqwrl:select(?y, ?x1, ?a, ?d, ?b1, ?c) ^ sqwrl:columnNames("TrainNames", "SourceStation", "RunsOn", "Departure Time", "Destination Station", "ArrivalTime")

4. Conclusions

The owl based ontology for Southern Railway, SRO-ONTOLOGY, was constructed with protégé 3.4.4, after following the listed steps for the ontology-development process. We have exploited the use of SWRL and SQWRL for inferring the constructed SRO-ONTOLOGY. Consistencies checking of ontology and classification of classes have been done with the support of Pellet reasoner. Ontology design is a creative process and no two ontologies designed by different people would be the same. The potential applications of the ontology and the designer's understanding and view of the domain will undoubtedly affect ontology design choices. It is hoped that this ontology will be of immense use to the travelers. In future, we will enhance the ontology with the local names of the places with the natural language support. Also we will attempt to integrate this ontology with the ontologies for other zones of Indian Railways and to evaluate the ontology.

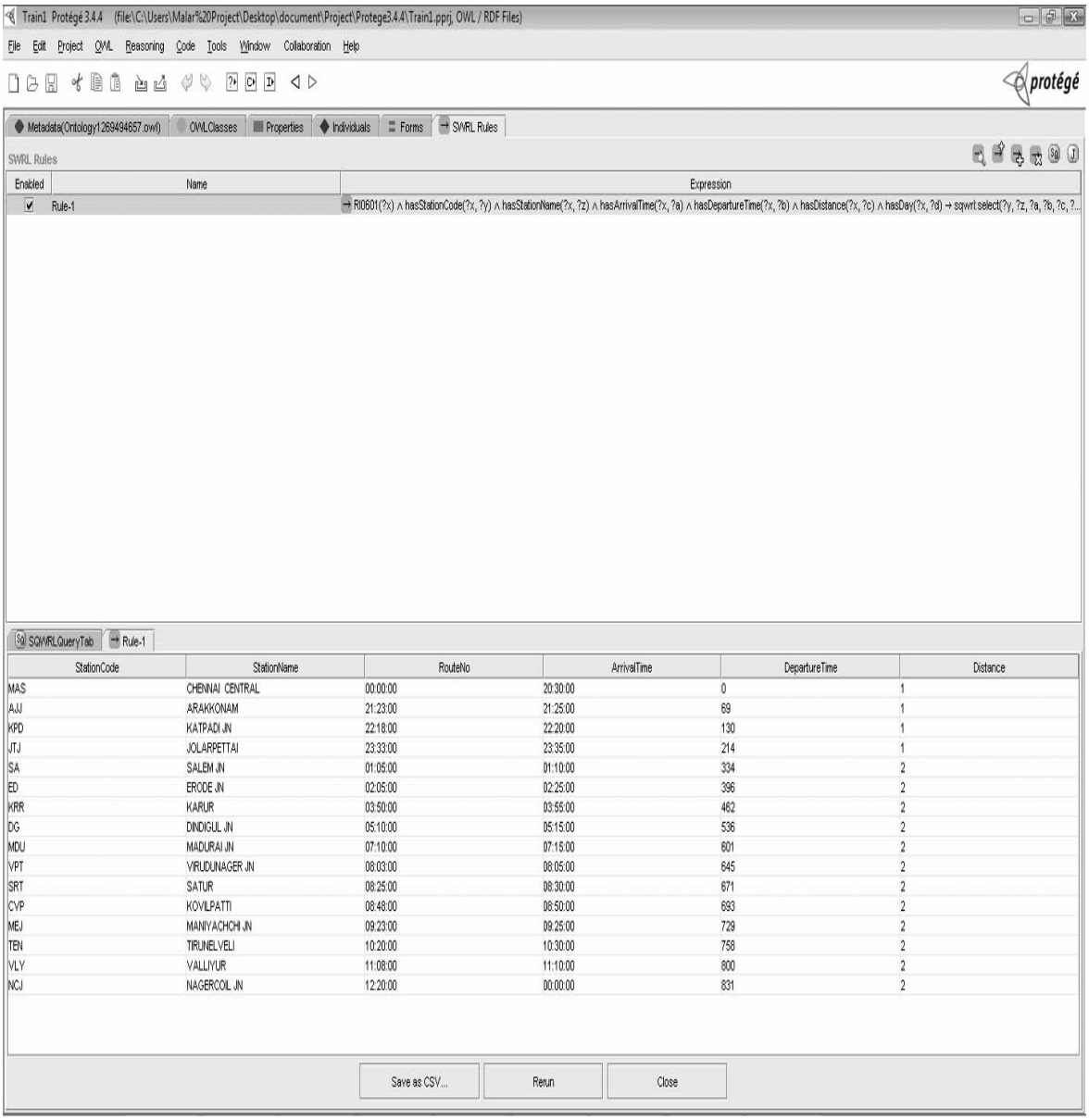

Figure 3 . SQWRL Tab and the result in SRO-ONTOLOGY

5. References

[1] Berners-Lee, T., Hendler, J., Lassila, O, "The Semantic Web", Scientific American, May 2001.

[2] Neches, R., Fikes, R.E., Finin, T., Gruber, T.R., Swartout, W.R, "Enabling Technology For Knowledge Sharing", AI Magazine 12(3), 1991, pp. 36-56.

[3] Alexander Maedche, Ontology Learning For The Semantic Web, Kluwer Academic Publishers, USA, 2002.

[4] Hyvonen, E., Saarela, S., Viljanen, K, "Application of Ontology Techniques To View-Based Semantic Search And Browsing, The Semantic Web: Research and Applications", Proceedings of the First European Semantic Web Symposium(ESWS), 2004.

[5] Ohno Machado, L., Gennari, J.H., Murphy, S.N., Jain, N.L., et al., "The Guideline Interchange Format", J. American Medical Informatics Association (5), 1998, pp. 357-372.

[6] Das, A., Wu, W., McGuinness, D, "Industrial Strength Ontology Management. The Emerging Semantic Web", IOS Press, 2002.

[7] Noy, N., McGuinness, D.L, "Ontology Development 101: A Guide To Creating Your First Ontology", Stanford Medical Informatics Technical Report SMI-2001-0880, 2001.

[8] Sure, Y, "On-To-Knowledge -- Ontology Based Knowledge Management Tools And Their Application", German Journal Kuenstliche Intelligenz, Special Issue on Knowledge Management, 2002.

[9] The Prot´eg´e project, http://protege.stanford.edu

[10] Sure, Y., Erdmann, M., Angele, J., Staab, S., Studer, R.,Wenke, D, "OntoEdit: Collaborative Ontology Engineering For The Semantic Web", International Semantic Web Conference, Sardinia, Italia , 2002.

[11] Bechhofer, S., Horrocks, I., Goble, C., Stevens, C, "OILEd: A Reason-able Ontology Editor For The Semantic Web", KI2001, Joint German/Austrian conference on Artificial Intelligence Vol. 2174, Springer-Verlag, Vienna, 2001, pp. 396–408.

[12] Arp´ırez, J. C., Corcho, O., Fern´andez-L´opez, M., G´omez-P´erez, A, "WebODE: A Scalable Workbench For Ontological Engineering", KCAP-01, Victoria, Canada, 2001.

[13] Farquhar, A., Fikes, R., Rice, J, "The Ontolingua Server: A Tool For Collaborative Ontology Construction", Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, 1996.

[14] SWRL-Submission, http://www.w3.org/submission/SWRL

[15] SWRL Bulit-ins, http://www.daml.org/2004/04/Swrl/builtins.html

[16] SWRLTab plug-in, http://protege.cm3.net/cgi-bin/wiki.pl?SWRLTab

[17] Southern Railway, http://www.southernrailway.gov.in/PassengerTimeTable

[18] Pellet Reasoner, http://www.pellet.owldl.com