An Intelligence Aided VANET System: A Development of its Ontology Knowledgebase and Rule Set

S. K. Aina, O. N. Oyelade

Abstract— Vehicular Ad-hoc Network (VANET) is a technology that uses moving vehicles as nodes in a network to create a mobile network. VANET turns every participating vehicle into a wireless router or node. Literature has revealed that information sharing among participating vehicles such as V2V (Vehicle to Vehicle), V2I (Vehicle to Infrastructure), lead vehicle and vehicle moving in opposite direction vehicle have been the challenges facing VANET system. In addressing these challenges, this paper seeks to leverage on the semantic web in improving information sharing problems. First, both the distributed and central knowledgebase are modeled with web ontology language (OWL). Secondly, a coordinated rule set was then crafted to help the system inferred missing knowledge which could add up to the existing knowledge so as to make information sharing effective. Thirdly, a communication protocol was created so as to enable the knowledgebase and the rule set to be used by an inference engine to intelligently provide nodes in the VANET system with adequate information needed for effective communication. The result shows that the VANET system implemented with this solution retrieves more relevant and dynamic knowledgebase than others.

Index Terms — Ontology, OWL, Pedestrian, Rule Engine, SWRL, VANET, Vehicle.

u

1 INTRODUCTION

Vehicular ad hoc network (VANETs) is an application of mobile network where every participating vehicle communicates and transfer information with each other [1]. VANETs are also known to be a distributed, self-organizing communication network built up by moving cars as nodes in a network to create a very high mobility of nodes and limited degrees of freedom in nodes movement patterns. It enhances the driving safety and comfort of vehicle users. It turns every participating car into a wireless router or node, allowing cars approximately 100 to 300 meters from each other to connect and in turn, create a network with a wide range [1]. Ontology defines basic terms and relations comprising the vocabulary of a topic area, as well as the rules for linking terms and relations to define extensions to the vocabulary [2]. It is being defined as an explicit, formal specification of a shared conceptualization of a domain of interest ([3]; [4]; [5]), which implies that it is a knowledge base which enables devices to communicate with each other effectively. Ontology can also be defined as a system that permits its designers to represent concepts in a given domain, and further show the relationships among these concepts [6]. This work designed a communication channel or protocol and VANET ontology that will be used alongside with the rule set. The architecture of the resulting VANET system is also presented in this work. Subsequent sections further outlines the sample questions upon which the rules or the communication protocol were built upon.

2 RELATED WORKS

There have been several studies on ontology based VANET. For example, in [7], an ontology modelling approach for assisting vehicle drivers through safety warning messages during time critical situation was proposed. The work aimed at creating alert messages based on the context aware parameters using driving situation and such as driver’s activity and environment.

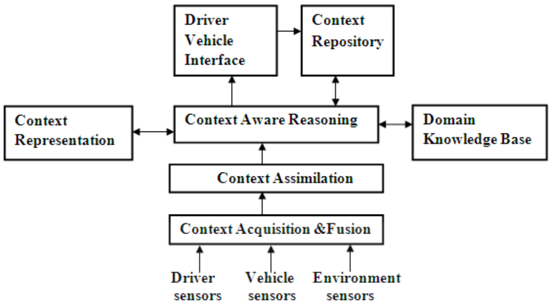


Figure 1 Architecture of Intelligent Driver Assistance System [7]

The work was implemented using ontology modelling and XML format that was used to manage the Intelligence- Driver Assistance System (I-DAS) parameter to illustrate the update /maintenance in the work. Java framework was used to generate safety alerts in several driving situations. The future work was to consider a more comprehensive extension of the overtaking knowledge base.

In the work of [8] a high-level representation of automated vehicle, other vehicles and their environment was proposed which can assist drivers in taking such “illegal” (which means autonomous vehicle should follow its obstacle avoidance algorithm instead of traffic rule) but practical relaxation decisions. The high-level representation, ontology includes topological knowledge and inference rules, in order to compute the next high-level motion an automated vehicle should take as assistance to a driver. Where two instances were given, for example:

a. Situation where a vehicle was stuck on the road because of a vehicle ahead of it stopped on the lane with no good reason that might take a longer time (could be offloading things), meanwhile according to the traffic regulation the vehicle cannot overtake.
b. Circumstance on reaching a roundabout, a vehicle ahead of a driver stopped on the lane with an engine problem, and because of the traffic regulation the driver has to wait behind the defective vehicle until the car continues its movement again, which could take a longer hour to rectify.

Human drivers can manage these two situations (a) and (b) by checking the oncoming vehicle behind it, then reverse to the back and pull off the lane but in the case of autonomous vehicle driven by a computer, in these two instance (a) and (b) since the vehicle is considering the traffic regulation the vehicle will continue waiting (trapped) till the vehicle ahead it continues its movement again. In these situations, for an autonomous vehicle to reason like a human driver (imitate) a decision needs to be taken, that is, should an autonomous vehicle follow its obstacle avoidance algorithm then change lane, or should it follow traffic regulation by staying on its lane?

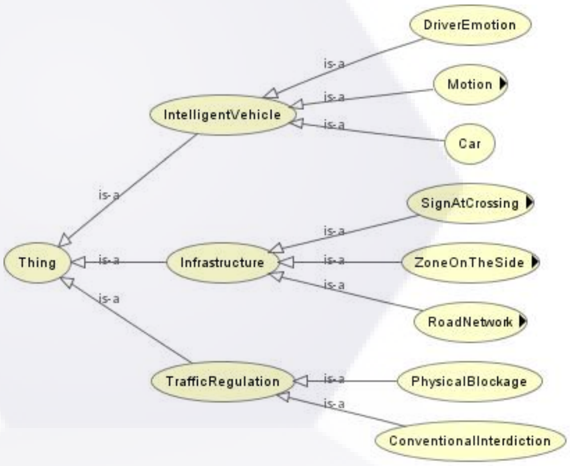


Figure 2 High-level Ontology Representation Model [8]

Table 1: Limitations of [8]

Technology used		Morignot and Nashashibi, (2012)
ADAS		Yes
ADAS&Human		No
Communication type design for the ontology	Leading vehicle	No
	SameDirection vehicle	No
	OppositeDirection vehicle	No
	RoadSideInfrastructure	No
	Pedestrian	Yes

The work focused on the internal part of each vehicle specifically its decisional part, it gives consideration for ADAS and pedestrian while the work do not considered given human driver advice, Leading vehicle, SameDirection, OppositeDirection and RoadSideInfrastructure communication.

[9] Discussed about embedding a symbolic representation that is ontology as a component of each vehicle in order for it to deal with emergency situation and also to manage the situation at intersection in order to reason on the usage of traffic rules. It also discusses about the automation level of the vehicle such that it will observe the state of a driver from fully aware through drowsiness to full asleep by using a camera detecting eye opening level, blink frequency and blink duration. They later proposed to determine the maximum autonomy level that a vehicle can adopt in order to cope with the current state of the environment so that it can ensure a safe driving. Automation means all the modes which imply actions were done by the system through the actuators, including actions against the driver's intention such as emergency braking.

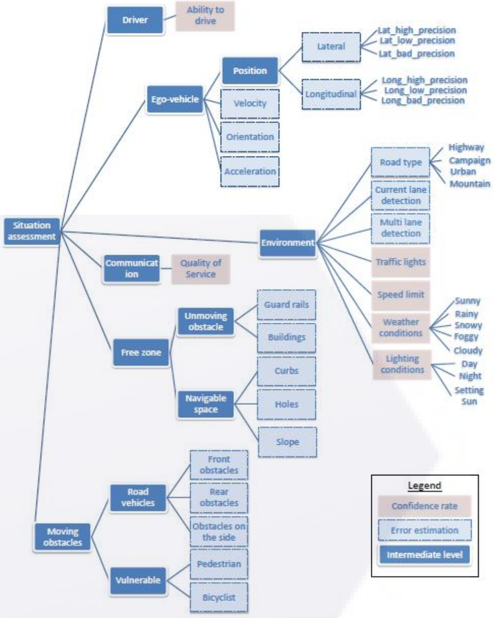


Figure 3 Ontology for Situation Assessment [9]

The limitations of this work were; it did not consider providing advice, warning to the driver, blind spot detection is not considered as an automation mode and communication between vehicles are not considered.

Ontology was used to model VANETS security in terms of identifying intrusion based on semantics [10]. The work has shown different techniques of ontology-based intrusion detection, summary of ontology web language and how ontology can be used to create namespaces and declaration of class in order to identify the relationship between the parameter of vulnerabilities in term of intrusion detection.

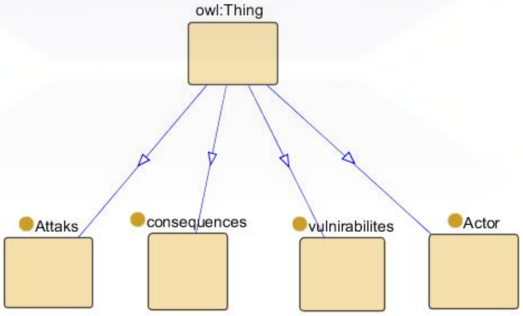


Figure 4 High Level Ontology [10]

This work was based on use of ontology to solve intrusion detection for vehicular ad hoc networks. The work did not reflect on other aspect of ontology of VANET and their environment.

An ontology-based framework was proposed which provides human like reasoning about the driving environment from the facts gathered from on-board sensor, maps and vehicle state [11]. This entails the information from different parameters available in road situations. The work uses terminological box (TBox) and an assertional box (ABox) to describe the concepts of their ontology in term of class and relationships between classes such as object properties and their rules. Also declare an instance of concepts such as individuals. The work used OWL to edit and verify the ontology in protégé and swoops. According to Armand *et al*., (2014) the objective was not to design an exhaustive ontology that considers every type of context entity, but rather to design a coherent, easily extendable ontology based framework and it can only reason on contexts compatible with it, that is contexts which only meet entities which have been defined in the TBox. This means that for an intensive use of the ontology, it has to be extended to take new types of entities into consideration.

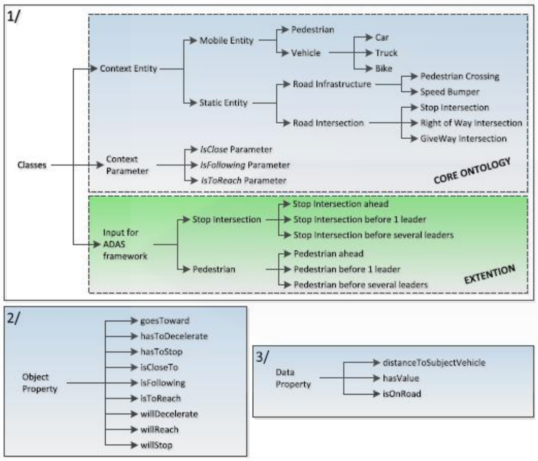


Figure 5 Ontology Classes, Object and Data Properties [11]

Table 2: Limitations of [11]

	Technology used	**Armand *et al.* (2014)**
ADAS		Yes
ADAS&Human		No
Communication type design for the ontology	Leading vehicle	Yes
	SameDirection vehicle	Yes
	OppositeDirection vehicle	No
	RoadSideInfrastructure	Yes
	Pedestrian	Yes

The work primarily based on ADAS (Advance driving Assistance System) was concerning the interaction between the lead vehicle, the pedestrian and the road intersection. The aim of this dissertation to improve performance in VANET system through the development and implementation of an enriched VANET ontology knowledge base which will extend the issue of lead vehicle considering ADAS, vehicle at the opposite lane and as well as gives a suggestion on road usage for a human driven vehicles.

3 THE PROPOSED VANET SYSTEM ARCHITECTURE

This section gives the detail of the architecture of the proposed system. The architecture is divided into four main components, namely:

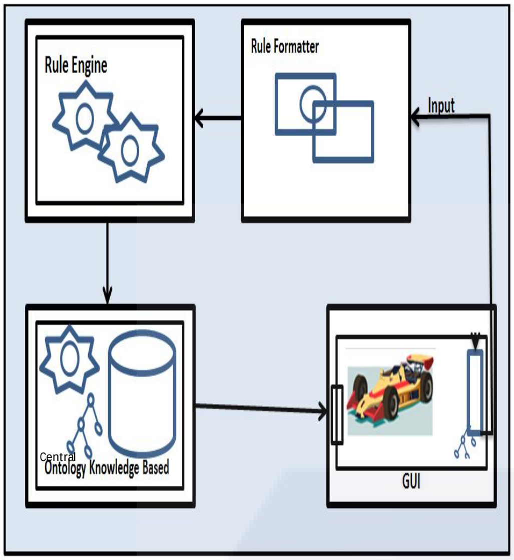


Figure 6: System Architecture

A. User Interface (UI)

The GUI (Figure 1) of the proposed system presents a prototype of what might have been seen as a simulated autonomous vehicle environment. For the purpose of demonstrating the communication pattern that could exist between vehicles in a VANET system, the UI is provided as means of invoking some possible events and actions that generate communication data. The component of the overall system is compartmentalized into two views: the control and event output panels. The control panel provides users with some control keys. Each of this key is denoting a possible action or event that could be thrown in a normal driving scenario. The second panel details the output of the control key sets that must have being triggered. For instance, should a user trigger a control key denoting change of lane, the ontology must first be update that vehicle A has charged lane. This ontology update is done through the rule set or communication protocol proposed in this work. Once this is noted in the ontology, other vehicles in the VANET system will be alerted so as to maintain safety precautions within the system. Meanwhile, in the event output panel, which in itself a canvas for drawing object, an action and reaction effects are seen based on the move of this control key. The overall emphasis of this synergy is to demonstrate that VANET ontology could be employed beyond being just a knowledge base to being a communication channel where all vehicles participating in the system update themselves on the state of the system.

The ontology shown within the UI component of the architecture is to imply that each vehicle in the VANET system has its own miniaturized ontology. This ontology models its personal details and information retrieved from a centralized ontology. The purpose for a centralized ontology is for systemic information sharing. While the aim of the miniaturized ontology for each vehicle is to decentralize each node's (vehicle) self-tuned data – this engenders autonomous-city of each vehicle in a data sharing system.

B. Inputs and Rule Formatter

Data entry are made through the control keys, however, each action triggered by these keys must be semantically interpreted as queries which are in turn formatted in rule format. These formatted rules are then run against the underlying ontology so as to make a fact known. This component in the system architecture illustrated by Figure 1 mediates between the knowledge base and the nodes (vehicles) in the VANET system. Hence the communication protocol proposed in this work.

C. Rule Engine

In Semantic web, rules are implemented against ontology by a rule engine. There are quite a number of rule engines and reasoners, and these include Pellet, Jess, Jena, Helmitt and FACT++. However, this work employs the use of Jess as a rule engine because of its support for SWRL. Rule engine, also called a reasoner, is a piece of software able to infer logical consequences from a set of asserted facts or axioms. It provides automated support for reasoning tasks such as classification, debugging and querying.

D. Central Ontology Knowledge Base

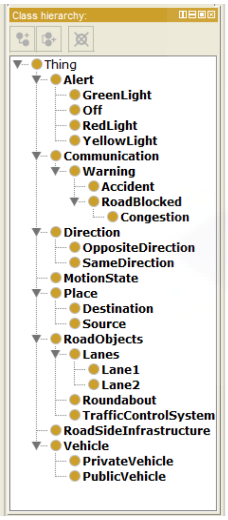
Most possible concepts or objects in the domain of VANET are being modeled in this ontology. This ontology includes both roadside objects and basic elements of a typical driving environment. All relating data are stored in this ontology. Inter-vehicular communication is attained through information sharing provided through this model. All the information are been reported to the central ontology that is, any passing car can report information based on what it has encountered from the direction where it is coming from or from the source where it is coming from, so that other cars, or vehicles can read from this central ontology which is installed on the roadside unit (infrastructures).

4 BUILDING A KNOWLEDGE BASE FOR VANET

This section illustrates how ontology development methodology is being done, it gives the details of the VANETs ontology concepts and also explains VANETs domain knowledge and how it is being built.

A. VANETs Ontology Development

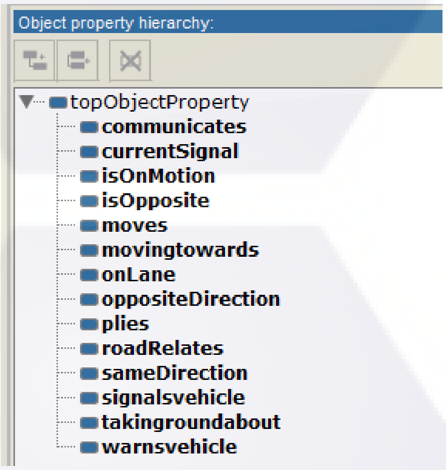
The following are the concepts being modeled for the VANET ontology: Communication, Warning (subclass is Accident), RoadBlocked (subclass is Congestion). Direction (has OppositeDirection and SameDirection as its subclasses). Place (has Destination and Source as it subclasses). RoadSideInfrastructure has no subclass. Vehicle (has PrivateVehicle and PublicVehicle as its subclasses). RoadObject (has Lane, Roundabout and TrafficControlSystem as it subclasses), while Lane has Lane1 and Lane2 as its subclasses and TrafficControlSystem has no subclass. Alert (has GreenLight, Off, RedLight and YellowLight as its subclasses). Lastly is MotionState modeled as a class. Figure 2 illustrates these concepts.



- Thing
  - Alert
    - GreenLight
    - Off
    - RedLight
    - YellowLight
  - Communication
    - Warning
      - Accident
      - RoadBlocked
        - Congestion
  - Direction
    - OppositeDirection
    - SameDirection
  - MotionState
  - Place
    - Destination
    - Source
  - RoadObjects
    - Lanes
      - Lane1
      - Lane2
    - Roundabout
    - TrafficControlSystem
  - RoadSideInfrastructure
  - Vehicle
    - PrivateVehicle
    - PublicVehicle

Figure 7: VANET concepts in ontology

Here, we list out the relations of the classes mentioned above. There are two types of properties (relations) that are used in OWL. These are data property and object property. Some of the properties used here are isopposite, communicates, moves, movingtowards, plies, signalsvehicle, takingroundabout and warnsvehicle. Figure 3 captures these relations as modeled in protégé.



- topObjectProperty
  - communicates
  - currentSignal
  - isOnMotion
  - isOpposite
  - moves
  - movingtowards
  - onLane
  - oppositeDirection
  - plies
  - roadRelates
  - sameDirection
  - signalsvehicle
  - takingroundabout
  - warnsvehicle

Figure 8: Classes relations on VANET ontology

5 A PROTOCOL FOR DYNAMICALLY POPULATING VANET ONTOLOGY

As a means of providing interoperation between the knowledgebase and the rule set, we propose this protocol to coordinate communication in the VANET system.

Table 3: Communication Protocol

Protocol Name	Description
Protocol-V	Insert(VehicleOntology: VehicleClass, CentralOntology: VehicleClass) VehicleParameterInstance (VehicleName, Direction, StartingLane,

Protocol Name	Description
	Source, Destination, Distance).
Protocol-R	Insert(CentralOntology: RoadObjectClass) RoadObjectParameterInstance (Lane1, Lane2, Roundabout, TrafficControlSystem)
Protocol-D	Insert(CentralOntology: DirectionClass, VehicleOntology:DirectionClass) DirectionParameterInstance (OppositeDirection, SameDirection)
Protocol-C	Insert(CentralOntology: CommunicationClass) CommunicationParameterInstance (Accident, Congestion)
Protocol-I	Insert(CentralOntology: RoadSideInfrastructureOntology) RoadSideInfrastructureParameterInstance (nil)
Protocol-P	Insert (CentralOntology: DisplacementClass, VehicleOntology: DisplacementClass) PlaceParameterInstance (Source, Destination)
Protocol-P	Insert(CentralOntology: AlertClass,VehicleOntology: AlertClass) AlertParameterInstance (GreenLight, Off, RedLight, YellowLight)
Protocol P	Insert(CentralOntology:Motion StateClass, VehicleOntology:MotionStateClass) MotionStateParameterInstance (Nil)

The composition of these protocols is to theoretically demonstrate how instances were semantically added into the corresponding ontology they relate with. The syntax of the protocol usually starts with the *Insert* keyword. This then follows with parenthesis and a list of the ontology files, with their classes, that are going to receive the supplied instances. An ontology name is separated from its class, object or concept by the : notation. While a comma separates a list of ontology files to be updated. The instances that were added to the selected class in the ontology were likewise listed in parenthesis, separated by comma.

6 EVENT CONTROL IN VANET THROUGH RULE SYSTEM

In this section basic interoperability of SWRL and OWL, the proposed VANET rule set created and functionality question and definition were discussed.

Table 4: VANETs System Functionality Question

A	Given a particular vehicle and a distance apart, get the data of vehicles which are both on the same lane and parallel lane.
B	Which roadside objects are close to a given moving vehicle?
C	Is the vehicle on motion and in what direction?
D	Which data has vehicle moving in opposite direction: Lane 1 Lane 2
E	Is vehicle at crossroads? Then what is the data from traffic control system.
F	What is the average speed to move with in respect of roadside objects?

In this work, SWRL is used to create rules and SQWRL is used to support OWL queries. The rule takes the format of the popular Horn like structure which comprises of both the antecedent and consequent. The antecedent is on the left hand side while the consequent is placed on the right hand side. Provided that a truth value or result is obtained from the antecedent, then the consequent follows, otherwise, no execution of the consequent.

Here, the outlines of the rules are shown for further implementation in the next chapter.

A. Given a distance apart, get the information of vehicles which are both same lane and parallel lane.

  i. Vehicle(?v) ^ onLane(?v, Lane2) ^ sameDirection(currentVehicle, ?v) ^ distance(?v, ?d) ^ swrlb:lessThanOrEqual(?d, 5) → sqwrl:select(?v)
  ii. Vehicle (?v) ^ onLane2 (?v, Lane2) → sqwrl: select (?v)
  iii. Vehicle (?v) ^ onLane1 (?v, Lane1) → sqwrl: select (?v)

B. Which roadside objects are close to a given moving vehicle?

  i. RoadObjects(?obj) ^ roadRelates(?obj, currentVehicle) ^ distance(?d, ?obj) → ^ swrlb:lessThanOrEqual(?d, 10) →sqwrl:select(?obj)
  ii. RoadObjects(Roundabout) → ^ roundrunrate(Roundabout, ?kmh) speedrate (?km, currentVehicle)
  iii. RoadObjects(Pedestrian) ^ pedestrianrunrate(Pedestrian, ?kmh) → speedrate (?kmh, currentVehicle)

C. Is the vehicle on motion, and in what direction?

  i. Vehicle(?v) ^ isOnMotion(?v, Motion) ^ plies(?v, ?lane) → sqwrl:select(?lane)

D. To obtain information of a vehicle moving in an opposite direction:

  i. Vehicle(?v) ^ oppositeDirection(currentVehicle, ?v) → sqwrl:select(?v)

The direction and lane plied by this vehicle will be programmatically decoded. This will be shown in the continuation of this work title the implementation of java programming with java library to improve performance in VANET ontology.

E. What is the average action to take with respect to traffic control system signal?

  i. RoadObjects(?obj) ^ abox:hasClass(TrafficControl System, ?obj) ^ currentSignal(?obj, ?signal) ^ averageSpeed(?signal, ?r) → sqwrl:select

(?r, ?signal)

F. What is the average speed to move with in respect of destination or roadside objects?

RoadObjects (?r) ^ averageSpeed (?r, ?speed) →sqwrl:select (?r, ?speed)

Vehicle(?v) ^ pathRelates(?v, currentVehicle) ^ distance(?v, ?d) ^ swrlb:lessThanOrEqual(?d, 5) → sqwrl:select (?v)

7 IMPLEMENTATION AND RESULT DISCUSSION

The implementation entails the coupling of the rule engine with the development environment and the tools used are protégé, NetBeans and Jess.

Protégé was used earlier to develop the ontology shown in figure 7. NetBeans 8.0.2 was deployed for creation of the development environment and the Jess is rule engine used for the implementation.

Coupling of Rule Engine with Development Environment (NetBeans)

Figure 9 shows the package listing of the Java files that were written in developing the simulated VANET environment. To configure our rule engine (Jess) with the source code, the jar files that consist of the rule engine are added to the package/project to which the implementation is being executed.

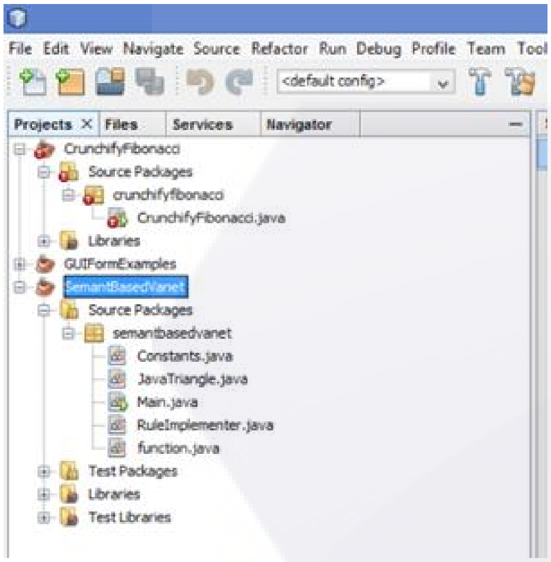
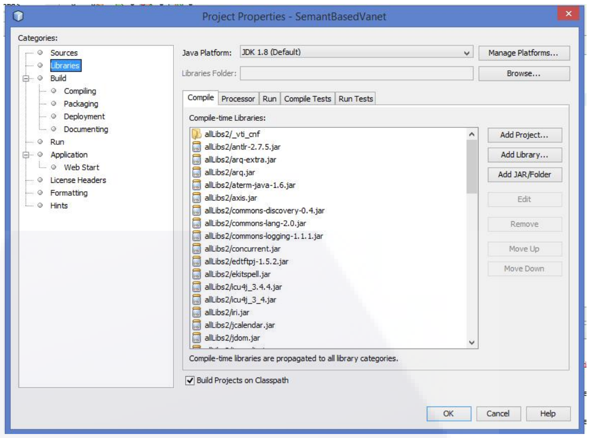


Figure 9 Package Listing of the Java Files

Figure 10 shows how to navigate to the NetBeans section for adding libraries to a project. To add the rule engine jar files, the Add Jar file button is clicked, and then all the jar files are added by selecting them, and then clicking on the Open button. Once this jar files are added, the source codes in the project automatically import the necessary packages/API they have being assigned to use.



Java Platform: JDK 1.8 (Default)

Libraries Folder:

Compile-time Libraries:

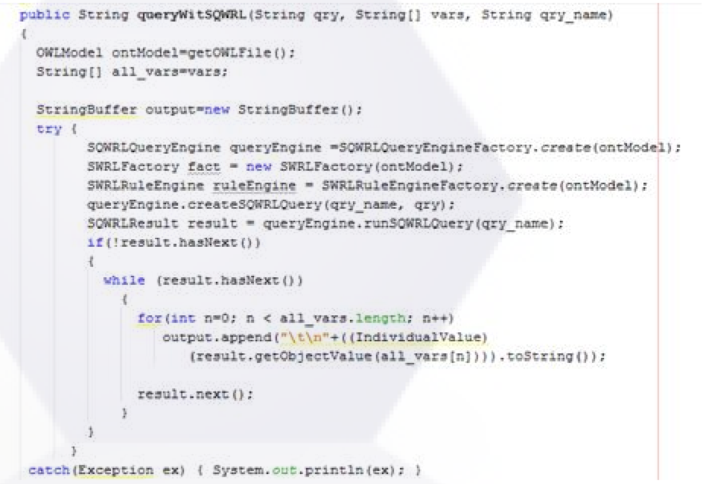
allLibs2/_vti_cnf
allLibs2/antlr-2.7.5.jar
allLibs2/arq-extra.jar
allLibs2/arq.jar
allLibs2/aterm-java-1.6.jar
allLibs2/axis.jar
allLibs2/commons-discovery-0.4.jar
allLibs2/commons-lang-2.0.jar
allLibs2/commons-logging-1.1.1.jar
allLibs2/concurrent.jar
allLibs2/edtftpj-1.5.2.jar
allLibs2/ekitspell.jar
allLibs2/icu4j_3.4.4.jar
allLibs2/icu4j_3_4.jar
allLibs2/iri.jar
allLibs2/jcalendar.jar
allLibs2/jdom.jar

Compile-time libraries are propagated to all library categories.

☑ Build Projects on Classpath

Figure 10 Navigate the NetBeans Section for Adding Libraries to a Project

Figure 11 shows the source code in Java to add rule set in connection with the development environment.



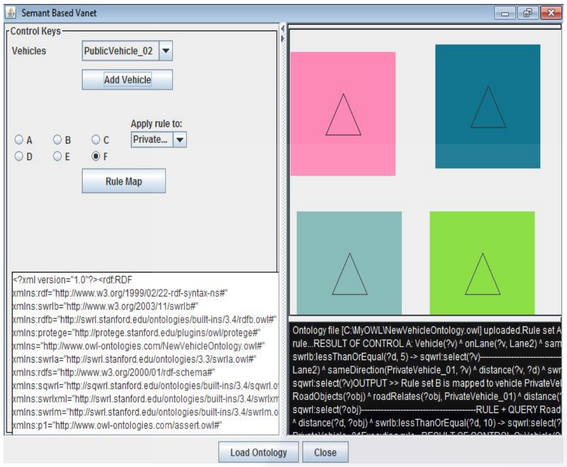
```java
public String queryWitSQWRL(String qry, String[] vars, String qry_name)
{
  OWLModel ontModel=getOWLFile();
  String[] all_vars=vars;

  StringBuffer output=new StringBuffer();
  try {
        SQWRLQueryEngine queryEngine =SQWRLQueryEngineFactory.create(ontModel);
        SWRLFactory fact = new SWRLFactory(ontModel);
        SWRLRuleEngine ruleEngine = SWRLRuleEngineFactory.create(ontModel);
        queryEngine.createSQWRLQuery(qry_name, qry);
        SQWRLResult result = queryEngine.runSQWRLQuery(qry_name);
        if(!result.hasNext())
        {
          while (result.hasNext())
            {
              for(int n=0; n < all_vars.length; n++)
                 output.append("\t\n"+((IndividualValue)
                     (result.getObjectValue(all_vars[n]))).toString());

              result.next();
            }
        }
      }
 catch(Exception ex) { System.out.println(ex); }
```

Figure 11 Rule Set Query Implementation in Java

7.1 Result



Vehicles: PublicVehicle_02

Apply rule to: Private...

○ A ○ B ○ C
○ D ○ E ● F

```
<?xml version="1.0"?><rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
xmlns:rdfb="http://swrl.stanford.edu/ontologies/built-ins/3.4/rdfb.owl#"
xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
xmlns="http://www.owl-ontologies.com/NewVehicleOntology.owl#"
xmlns:swrla="http://swrl.stanford.edu/ontologies/3.3/swrla.owl#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:sqwrl="http://sqwrl.stanford.edu/ontologies/built-ins/3.4/sqwrl.o
xmlns:swrlxml="http://swrl.stanford.edu/ontologies/built-ins/3.4/swrlxm
xmlns:swrlm="http://swrl.stanford.edu/ontologies/built-ins/3.4/swrlm.o
xmlns:p1="http://www.owl-ontologies.com/assert.owl#"
```

```
Ontology file [C:\MyOWL\NewVehicleOntology.owl] uploaded.Rule set A
rule...RESULT OF CONTROL A: Vehicle(?v) ^ onLane(?v, Lane2) ^ sam
swrlb:lessThanOrEqual(?d, 5) -> sqwrl:select(?v)------------------------
Lane2) ^ sameDirection(PrivateVehicle_01, ?v) ^ distance(?v, ?d) ^ swr
sqwrl:select(?v)OUTPUT >> Rule set B is mapped to vehicle PrivateVe
RoadObjects(?obj) ^ roadRelates(?obj, PrivateVehicle_01) ^ distance(
sqwrl:select(?obj)---------------------------------------RULE + QUERY Road
^ distance(?d, ?obj) ^ swrlb:lessThanOrEqual(?d, 10) -> sqwrl:select(?
```

Figure 12 UI showing Vehicle Inserted into Left and Right Lanes

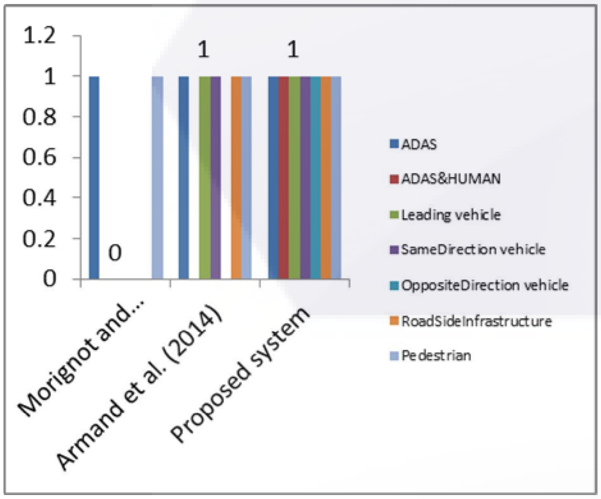


Figure 13 Comparison of the Proposed System with Some Previous Closest Work

7.2 Discussion

The designed ontology work successfully with the rule set and populated protocols thereby resulting to the outcome in figure 12 and 13. The result which shows there was effective communication between participating vehicles.

Figure 12 shows the outcome of the implementation in a form of UI that has two lanes (left and right) to add vehicles where ontology were uploaded, rules were applied in form button A to F and information sharing occurred among moving vehicles. While figure 12 shows how our work considered communication of ADAS & HUMAN and OppositeDirection which other works did not have, which make the proposed work to have more facilities compared to the previous works.

8 CONCLUSION

In this paper, an intelligent protocol for information exchange or sharing in a VANET system has being designed and presented. Furthermore, an ontology base knowledgebase was created for the VANET system. This knowledgebase works with a rule set also created for the VANET system. The proposed VANET system was implemented and the results shows that it provides a more efficient means of information sharing through retrieval among nodes of the VANET system.

REFERENCES

[1] Shastri, A; Dadhich, R. and Poonia R. C. (2011): Performance Analysis of On-Demand Routing Protocols for Vehicular ad-hoc Networks. *International Journal of Wireless & Mobile Networks (IJWMN)* **3** (4): 103 – 111.

[2] Neches, R; Fikes, R; Finin, T; Gruber, T; Patil, R; Senator, T. and Swartout, W.R. (1991): *Enabling Technology for Knowledge Sharing*. AI Magazine. Pp. 36-56.

[3] Gruber, T. R. (1993): *A translation Approach to portable ontology specifications. In Knowledge Acquisition*. **5**: 199-220.

[4] Studer, R; Benjamins, V. R. and Fensel, D. (1998): Knowledge Engineering: Principles and Methods. *Data and Knowledge Engineering*. **25**: 161-197.

[5] Seddiqui, M. H. and Aono, M. (2010): Metric of Intrinsic Information Content for Measuring Semantic Similarity in an Ontology. Proc. 7th Asia-Pacific Conference on Conceptual Mod-

elling (APCCM 2010), Brisbane, Australia. Pp. 89 – 96.

[6] Obiniyi, A. A.; Oyelade, O. N. and Auta G. I. (2014): Ontology Languages, Development Tools and Repositories: Towards a Unifying Platform. *International Journal of Computer Science and Artificial Intelligence* **4** (3): 68 – 74.

[7] Saravanan, K; Arunkumar, T. and RameshBuba, K. (2010): An Intelligent Driver Assistance System (I-DAS) for Vehicle Safety Modelling using Ontology Approach: International Journal Of UbiComp (IJU), **1** (3): 15 – 29.

[8] Morignot, P. and Nashashibi, F. (2012): An Ontology-Based Approach to relax Traffic Regulation for Autonomous Vehicle Assistance. INRIA Rocquencourt, Team IMARA, Domaine de Voluceau, B.P. 105, 78153 Le Chesnay, France. Pp 1 – 8.

[9] Pollard, E; Morignot, P. and Nashashibi, F. (2013): An Ontology-based Model to Determine the Automation Level of an Automated Vehicle for Co-Driving. Published in16th International Conference on Information Fusion. hal-00838680, version 1 – 27.

[10] Erritali, M; EL-Ouahidi, B; Hssina, B; Bouikhalene, B; and Merbouha, A. (2013): An Ontology-Based Intrusion Detection for Vehicular Ad Hoc Networks: *Journal of Theoretical and Applied Information Technology* 31st July 2013. **53** (3): 410 – 414.

[11] Armand, A; Filliat, D. and Ibañez-Guzman, J. (2014): Ontology-Based Context Awareness for Driving Assistance Systems. *Intelligent Vehicles Symposium Proceedings*, 2014 IEEE Dearborn, MI. 8-11 June 2014, 14452153: Pp227 – 233.