FUZZIFIED SEMANTIC WEB REASONING FOR ACTIVITY DETECTION IN WMSN APPLICATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ALI NAIL ÖZDIN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JANUARY 2019

Approval of the thesis:

FUZZIFIED SEMANTIC WEB REASONING FOR ACTIVITY DETECTION IN WMSN APPLICATIONS

submitted by **ALI NAIL ÖZDIN** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Halit Oğuztüzün
Head of Department, **Computer Engineering** _____

Prof. Dr. Adnan Yazıcı
Supervisor, **Computer Engineering, METU** _____

Examining Committee Members:

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering, METU _____

Prof. Dr. Adnan Yazıcı
Computer Engineering, METU _____

Assoc. Prof. Dr. Murat Koyuncu
Information Systems, Atılım University _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Ali Nail Özdin

Signature :

ABSTRACT

FUZZIFIED SEMANTIC WEB REASONING FOR ACTIVITY DETECTION IN WMSN APPLICATIONS

Özdin, Ali Nail
M.S., Department of Computer Engineering
Supervisor: Prof. Dr. Adnan Yazıcı

January 2019, 67 pages

Activity detection in WMSNs is a hot topic for surveillance applications since automated determination of activities is a difficult process. This study aims to increase the reliability of activity detection by using semantic Web technologies extended with fuzzy logic. The proposed approach consists of three layers: sensor, data and semantic Web layers. The sensor layer includes a WMSN including sensor nodes with multimedia and scalar sensors. The data layer retrieves and stores data from the sink of the WMSN. At the top of the architecture, there is a semantic Web layer including a semantic Web application server, a fuzzy reasoning engine, and a semantic knowledge base. When there is a new entity detection at the sensor layer, the related data produced by the sensors and the sink is collected in the data layer and transmitted to the semantic Web application server where the data is converted to subjects-predicates-objects in accordance with designed ontology and saved in RDF format. Then, the fuzzy reasoning engine is automatically activated and fuzzy rules are executed to decide whether there is an activity in the controlled area. The proposed approach is implemented for an example surveillance application in which various threat types

are deduced (i.e., attack, protest or ordinary mobility). In the implementation, the sensor layer is simulated with an application which produces synthetic data according to given scenarios. Users of the system can monitor occurring events in near-real time or replay recent events on their browsers. This implementation proves that the semantic Web technologies extended with fuzzy logic may have a significant impact on activity detection in WMSNs.

Keywords: fuzzy logic, semantic Web, Web 3.0, activity detection, wireless multimedia sensor networks

ÖZ

KABLOSUZ MULTİMEDYA SENSÖR AĞLARINDA AKTİVİTE TESPİTİ İÇİN BULANIKLAŞTIRILMIŞ ANLAMSAL WEB MUHAKEMESİ

Özdin, Ali Nail
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Adnan Yazıcı

Ocak 2019 , 67 sayfa

Kablosuz multimedya sensör ağlarında (KMSA) aktivite tespiti oluşan aktivitelere otomatik olarak karar vermek zor bir süreç olduğundan gözetleme uygulamaları için sıcak bir konudur. Bu çalışma bulanık mantık ile genişletilmiş anlamsal Web teknolojilerini kullanarak aktivite tespitinin güvenilirliğini artırmayı hedeflemektedir. Önerilen yaklaşım üç katmandan oluşmaktadır; sensör, veri ve anlamsal Web katmanlarıdır. Sensör katmanı multimedya ve skaler sensörleri içeren sensör düğümlerinden oluşan KMSA'nı içerir. Veri katmanı veriyi KMSA'nın sink bilgisayarından alır ve kaydeder. Mimarinin en üstünde ise anlamsal Web uygulama sunucusu, bulanık muhakeme motoru ve anlamsal bilgi tabanını içeren anlamsal Web katmanı bulunmaktadır. Sensör katmanında yeni bir varlık tespiti yapıldığında, sensör düğümleri ve sink bilgisayarı tarafından bu tespite ilişkin üretilen veriler veri katmanında toplanır ve veri tasarlanan ontolojiye uygun olarak subject-predicate-object üçlüsüne dönüştürülüp RDF formatında kaydedildiği anlamsal Web uygulama sunucusuna iletilir. Sonrasında bulanık muhakeme motoru kontrollü bölgede bir aktivite oluşup oluşmadığına karar vermek için otomatik olarak aktifleşir ve bulanık kurallar işletilir. Önerilen yaklaşım,

içerisinde çeşitli tehdit tiplerinin çıkarımının yapıldığı örnek bir gözetleme uygulaması için uygulanmıştır(örneğin, saldırı, protesto veya sıradan hareketlilik). Uygulamada, sensör katmanı verilen senaryolara göre sentetik veri üreten bir program ile simule edilmiştir. Bu sistemin kullanıcıları anlık oluşan etkinlikleri gerçek zamana yakın olarak veya son zamanlarda oluşan etkinlikleri tekrar oynatarak izleyebilirler. Bu uygulama bulanık mantık ile genişletilen anlamsal Web teknolojilerinin kablosuz multimedya sensör ağlarında yapılan aktivite tespitinde önemli bir etkiye sahip olabileceğini gösterir.

Anahtar Kelimeler: bulanık mantık, anlamsal Web, Web 3.0, aktivite tespiti, kablosuz multimedya sensör ağları

To My Lover...

ACKNOWLEDGMENTS

First of all, I would like to thank my supervisor Prof.Dr. Adnan YAZICI for his guidance, encouragements, advices and criticisms during this thesis research. When I said “I do not have an experience related to Web technologies; but I want to learn something that I will be able to use in the future” he did not refused my request and advised me to work with futuristic Web technologies like semantic Web. I will always remember his valuable support and effort to finish this research.

I also wish to thank Assoc.Prof.Dr. Murat KOYUNCU, Prof.Dr. Ahmet COŞAR, Asst.Prof.Dr. Mustafa SERT, Dr. Seyyit Alper SERT for their valuable discussions and suggestions about my thesis.

I also would like to thank a lot to PhD candidate Cihan KÜÇÜKKEÇECİ for his data simulator that plays an important role in my project.

My special thanks are to my darling, Burcu BAŞAR for her moral support and endless patience.

Finally, I want to acknowledge that I have spent long working hours in order to finish my thesis and I apologize from my darling, my family and my friends for not separating much more time to them in order to finish my thesis. I promise I will separate much more time after today.

TABLE OF CONTENTS

APPENDICES

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
BSI	Border Safety Informer
CGI	Common Gateway Interface
COG	Center Of Gravity
DB	Database
FOAF	Friend of a Friend
GMM	Gaussian Mixture Model
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hyper-Text Transfer Protocol
ID	Identification
IDE	Integrated Development Environment
KB	Knowledge Base
PIR	Passive Infrared
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RSS	RDF Site Summary
SPARQL	SPARQL Protocol and RDF Query Language
SVG	Scalable Vector Graphics
SVM	Support Vector Machine
URI	Uniform Resource Identifier
WSN	Wireless Sensor Network
WMSN	Wireless Multimedia Sensor Network
XML	Extensible Markup Language

CHAPTER 1

INTRODUCTION

1.1 Motivation

The rapid development of microelectronics over the past decade has resulted in energy savings and smaller sensors in our daily lives. The reduction in sensor size has resulted in an increase in the scale of growth of wireless multimedia sensor networks [4]. As a result, wireless multimedia sensor networks are commonly used to collect data from the physical environment. Wireless multimedia sensor networks have many different applications. Some of them are field monitoring, health monitoring or earth sensing. Particularly in security applications, field monitoring is one of the common uses of wireless multimedia sensor networks, where sensors are deployed in a selected area and events are expected to be detected [5]. On the other hand, another common use of wireless multimedia sensor networks is the earth sensing applications where environmental conditions such as disease detection [6], air pollution [7] or water quality drinking [8] are monitored. More specifically, the position and pulse of patients are recorded in health monitoring applications. If the patient has a pulse-related disease, the disease can be detected and diagnosed early even if the patient is at home [6].

Each monitoring application has an activity detection mechanism. Activity detection has been an important research area in the development of technology. Activity detection can be achieved from environmental measurements done by scalar sensors such as PIR, acoustic and vibration sensors as well as multimedia sensors such as camera and microphone. After collecting sensor measurements in WMSNs, sensors transmit measurements to sensor nodes and those data is processed and transmitted to

the central without human assistance. With this feature, sensor networks are ideally suited for activity detection applications in hazardous environments such as infectious material search [9] or nuclear radioactivity source detection [10].

The difficulty in determining the activity depends on the type of activities to be determined. Identifying an activity involves distinguishing between similar activities and deciding which activity is involved. It is possible to differentiate activities using a rule-based system. Rule-based systems are a type of expert system consisting of a set of rules. Rule-based systems can be created in two ways. The first is to develop the system by learning from real data and the second to establish such a rule-based mechanism by acquiring information from domain experts [11]. In this study, the rule-based mechanism is created using semantic Web technology based on information obtained from field experts. According to Tim Berners-Lee et al. [12], the semantic Web is a continuation of Web 2.0 and allows computers and people to collaborate with semantically well-defined information. The semantic Web aims to explain the meaning of the data, not the raw data. In addition, the semantic Web is used to combine different data sources and obtain new data by reasoning from these sources [3].

For a rule-based system, an activity has occurred or not, there is no third possibility. But in reality, rule-based systems have many decision points, because the truth of some rules is relative. It is very convenient to use fuzzy set theory to increase the reliability of a rule-based system. The fuzzy set, proposed by L. A. Zadeh in 1965, is created by a membership function where each decision point corresponds to a set of values with a degree [0,1], between 0 and 1 [13]. Fuzzy set theory is related to human reasoning and skepticism based on intuitive reasoning. For example, the reasoning does not consist of two outcomes. Unlike computers, tallness of a person varies from person to person and depends on the application domain [14]. On the basis of information received from domain experts, it is important to decide whether an event occurs with a possibility of zero or one or a value between zero and one ([0,1]) or whether a person is tall or not, via the functions of belonging of the corresponding fuzzy sets.

1.2 Problem Definition

In this thesis, activity detection is performed by establishing the wireless multimedia sensor network on a physical environment. While putting this into practice, two fundamental problems are encountered and a new approach is adopted to solve these problems. The first of these concerns is how to make processed sensor data acquired from the sink of WMSN available in a well-defined format for end-users and other systems to consume this data. Second, how to make the decision, or, in other words, which mechanism for activity detection is more reliable. Recent studies in the literature do not focus on solving these two problems together. For example, [15] and [3] aim at collecting data from different types of wireless sensor networks, then using Semantic Web technologies to standardize this data and communicate different types of sensors. Thus, we can deduce new information using reasoning techniques on the semantic Web. Moreover the study given in [1] make WSN data available for end-user's usage by defining a semantic web ontology. On the other hand, the study given in [16] use wireless sensor networks with fuzzy set theory to detect activity in the environment. The reason for the lack of such systems is that such intelligent sensor network data is not available for end-users, because these systems do not have a semantic Web mechanism so there is no well-defined, machine processable data for end users. Therefore, no study that solves these two problems with the presented infrastructure is available in the literature.

In order to solve the two problems described above, the proposed approach consists of three layers which are the sensor, data and semantic Web layers. The sensor layer covers the environment in which the wireless multimedia sensor nodes, gateways and the sink are deployed. In the second layer, the data sensed by sensors and transformed into object recognition data by pre-processing in the sensor nodes and/or the sink, which are ready to be used for activity detection or be shared, are stored. At the top of the architecture, there is a semantic Web layer including a semantic Web application server, a fuzzy reasoning engine, and a semantic knowledge base. When a new data item reaches to the semantic Web application server, it is converted into semantic Web tuples (subjects, predicates, and objects) and stored in RDF format. Then, the fuzzy reasoning engine is activated, executes a process called "fuzzy reasoning" and

the related rules are fired for activity detection.

In the context of putting this approach into practice, an example surveillance application is implemented as a case study. This application simulates a WMSN deployed around a border post and attempts to detect threats (i.e., attack, protest or ordinary mobility), in terms of varying levels of concern. In the implementation, the WMSN is simulated by a program called Environment Simulator. The sensor data collected and transformed into entity recognition data by processing in the sensor layer is transferred to the data layer where a sensor database is available. In the semantic Web layer, there are an application server that takes requests from clients and accordingly provides responses, a semantic knowledge base in which data is stored in RDF format, and a fuzzy reasoning engine with fuzzy rules. In addition, fuzzification and defuzzification processes are performed at this layer. A new semantic Web ontology based-on fuzzy logic is defined for activity detection. The reasoning results are presented to the user with a hazard ratio that is calculated using the detected entity types (person, vehicle, animal), time, directions of the detected entities, and proximity to the post center. This calculation is augmented with the fuzzy set theory. The activities detected in the WMSN, are presented with a GUI to the end-users. In this thesis, a new approach is introduced using a combination of semantic Web, wireless multimedia sensor network, fuzzy set theory and reasoning for activity detection in monitoring applications using WMSNs and more complete use is therefore compared to other approaches in the literature.

1.3 Contribution

The first contribution of this thesis is to translate the processed sensor data obtained from the sink of wireless multimedia sensor network into semantic Web data in order to make this data available for other software or end-users. The second contribution is the integration of fuzzy logic into the semantic Web-based reasoning system. There are no studies in the literature using these two contributions together. For example, the studies given in [3] and [15] aimed at collecting data from different types of wireless sensor networks and standardizing sharing of collected data with semantic Web technologies. Moreover, some studies make WSN data available for end-users

by defining a semantic web ontology [1]. However, these studies do not use the fuzzy set theory. On the other hand, there are studies using the fuzzy set theory to detect activity in the environment without using semantic Web technologies [16]. Therefore, the approach used in this study differs from existing studies.

Throughout the activity detection process, this study provides intuitive results in semantic Web applications by using fuzzy set theory. In other words, fuzzy set theory is used in this study because it offers the possibility of inferencing similar to that of humans. With this feature, the reliability of the reasoning results generated by the rule-based system increases because of future data for updating rules by field experts. On the other hand, when semantic network technologies are not used in applications where only the wireless sensor network and fuzzy set theory are used, it may be impossible to open processed sensor data produced in the sink to other software or end-users and identify critical activities. Our approach eliminates these problems.

In this thesis, to demonstrate the proposed concept, a surveillance application which uses processed WMSN data as a case study is implemented. The Environment Simulator is used to simulate and acquire processed sensor data or in other words entity recognition data, as if source of this data is the sink, to detect activities. The study was tested with different scenarios by Environment Simulator. In addition, a semantic Web application has been implemented to use processed real-time sensor data from this simulator, as if source of this data is the sink. This application converts the simulator data into semantic Web RDF tuples according to the designed ontology. The resulting RDF tuples are stored in the database. After executing the fuzzy logic in the RDF data, the sensors detect the objects and then intuitive results are visually displayed on the Web by the user. The reasoning results correspond to the default hazard rate values that are calculated using the detected entity types (person, vehicle, animal), the time, the directions of the detected entities, and the proximity of the protected area in the WMSN environment. This calculation is augmented with the theory of fuzzy sets for fuzzy reasoning by the Semantic Web. In addition, with this GUI, any activity in the WMSN environment can be shown to the user in real time or offline.

1.4 Outline

The rest of the work is organized as follows: The next section provides background information on fuzzy set theory and semantic Web technologies. Subsequently, the studies linked to sensor networks, semantic Web and fuzzy set theory concepts are briefly discussed. In addition, the reasoning used in the systems included in the literature is compared to the approach proposed in this thesis. Chapter 3 describes our approach and the theoretical use of the semantic Web and fuzzy set theory to identify activities through the WMSN. Chapter 4 describes the case study; architecture, and capabilities of our example surveillance application. Finally, Chapter 5 describes the conclusions and possible future work of this thesis.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Background

In this thesis, by combining fuzzy set theory and the semantic Web, an infrastructure in which processed sensor data in the sink can be represented in a well-defined and machine-processable format is introduced, and aims to increase the reliability of decision-making mechanisms. In this part, the semantic Web and fuzzy set theory technologies used in this thesis are explained in detail.

2.1.1 Semantic Web

The semantic Web was announced by Tim Berners Lee as a revolution in Web 2.0 in 2001 and it was stated that there is a standard for digitally storing well-defined information to work collaboratively with computers and users [12]. The rise of the digital world began with the Artificial Intelligence concept introduced at the Dartmouth Conference in 1956 [17]. Today, the World Wide Web connects billions of documents and search engines quickly find the necessary information on these sites. We believe that the use of Semantic Web will become more and more common and important as AI, because semantic Web makes machine-readable Web documents.

Documents on the Web are designed so that people can read and manipulate them. The semantic Web, on the other hand, represents an innovation that defends the theory that computers can do it. Semantic Web establishes logical connections, which are also referred to as "meaning of Web data", for different systems to work in harmony with each other. For example, suppose that a person books a theater ticket on a

website and books a vacation for another date on another website. Let's say that this person has a calendar application that can list all the reservations made by him / herself over another website. The existence of such an ecosystem is only possible by the semantic Web theory. Different Web applications can communicate with each other and in this way the person sees his / her made reservations from a single Web application. In order for computer programs to extract from Web content, the content must be produced in a machine-readable form. The contents produced for this purpose are expressed as ontology. The languages used to create these ontologies are expressed as ontology languages [18]. For example, the Resource Description Framework Schema is one of the most widely used ontology languages. RDFS is an ontology language that can be created using the RDF conceptual data model. In addition, the query language SPARQL is used to query the semantic data. In this part, we talk about the issues that are the cornerstones of the semantic Web such as RDF[19], RDF Schema and reasoning.

2.1.1.1 Resource Description Framework

It is a formal language for creating a common data structure to enable Web applications to exchange data with each other. The main difference from languages such as HTML and XML is that processing activities such as reasoning on data can be performed rather than just displaying the Web documents correctly. RDF is the most basic representation format used in semantic Web applications. RDF has been used in RSS 1.0 previously.

An RDF document defines a directional graph. A set of nodes is connected to each other by directional arrows. Node and edge are distinguished by their labels. For example, Figure 2.1 describes the relationship between a film and its producer in RDF graph representation. Looking at this figure it will be inferred that the film1 is produced by producer1. At this point, separating RDF from standards such as XML, film1 and producer1 are labeled using Uniform Resource Identifiers. In this way, RDF aims to have a single URI accepted for each resource. Thus, for instance, the same sensor may have different names in the XML files produced by two different systems expressing the data of a temperature sensor while the RDF standard assumes this and

argues that the same resource must have the same URI. To achieve this, it is necessary to work with well-defined vocabulary according to RDF standard. Vocabulary can be selected depending on context. For example, if it is planned to produce RDF in a subject where people are communicating with each other, the Friend of a Friend (FOAF)[20] vocabulary should be used. Data values displayed in RDF are kept in objects called "literal". Literals store data such as numbers, text, Boolean, time. In Figure 2.1, "Title of film1" and "Name of producer1" are literal objects in the graph.

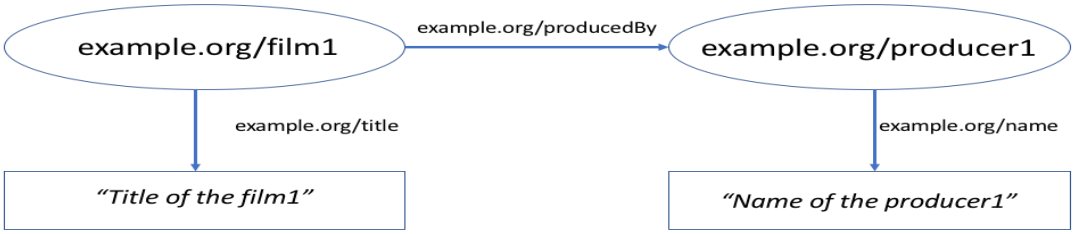


Figure 2.1: A basic RDF graph representing relationship between a film and producer

Processing of an RDF data by computers is not possible with graph representation. Therefore, RDF data must be serialized. They are converted to triples called subject-predicate-object for serialization. For example, in Figure 2.1, "example.org/film1", "example.org/producedBy" and "example.org/producer1" are referred to as subject, predicate and object respectively. Similarly, "example.org/film1", "example.org/title" and "Title of the film1" can be expressed in the same way. As can be understood here, while a URI type may be a subject, predicate or object, the literal may be the only object. Triple syntax was first published in 1998 by Tim Berners-Lee as Notation 3 (N3). Subsequently, N-Triples and Turtle syntaxes followed this development in 2004. In Turtle syntax, URIs can be abbreviated using namespaces, and also URIs are written with angular brackets. Literals are shown in the quotation marks. Each line ends with a dot. Below is an RDF document written using the Turtle syntax:

```
@prefix ex:  http://example.org/ .
ex:film1 ex:producedBy ex:producer1.
ex:film1 ex:title  "Title of the film1".
ex:producer1 ex:name  "Name of the producer1".
```

2.1.1.2 RDF Schema

While defining a new domain of interest, some inferences that can be easily made by the human brain cannot be done by computers. For example, let's assume that there are classes such as "person", "book", "library", individuals such as "Ali Ozdin", "METU Library", "Calculus Book" and relationships like "worksIn". Human brain can easily deduce that "worksIn" relationship is between the "person" class and the "library" class, while the computer cannot do it without providing related information. RDF Schema provides additional descriptions of the RDF data that the computer can understand. The most common ones are rdf:type, rdfs:Class, rdfs:subClassOf, rdf:Property, rdf:subPropertyOf, rdfs:comment, rdfs:seeAlso tags. The following describes the use and contribution of these tags.

```
@prefix ex:  http://example.org/  .
ex:film1 rdf:type ex:HorrorFilm .
ex:HorrorFilm rdf:type rdfs:Class .
```

Type and Class: The first thing that is needed when defining a resource in Domain of Interest is to define this resource as the element of a certain aggregation. **rdf:type** is used to define the classes to which the resources belong. However, these classes should be explicitly emphasized to be perceived as class by computers. Because the fact that a resource has a URI does not necessarily indicate that it is a class. In the example above, ex: film1 is not a class. Therefore, the class that the resource belongs to is explicitly expressed by using **rdf:type** with the reference to the class **rdfs:Class**.

```
@prefix ex:  http://example.org/  .
ex:film1 rdf:type ex:HorrorFilm .
ex:HorrorFilm rdf:type rdfs:Class .
ex:Film rdf:type rdfs:Class .
```

Subclass: For example, suppose we have an RDFS document as shown above. The human brain may conclude that a horror movie is also a film, but the same is not possible for computers. ex:film1 will not be among the results in the query we will make to this document to see all the films of "ex:Film" type. Therefore, it is necessary to specify the relations between the classes on the RDFS document. For this, it is

necessary to state that every horror movie is a movie by using the **rdfs:subClassOf** tag as follows.

```
@prefix ex:  http://example.org/  .
ex:film1 rdf:type ex:HorrorFilm .
ex:HorrorFilm rdf:type rdfs:Class .
ex:Film rdf:type rdfs:Class .
ex:HorrorFilm rdfs:subClassOf ex:Film .
```

```
@prefix ex:  http://example.org/  .
"Mr. X" ex:graduatedFrom ex:METU .
Ex:graduatedFrom rdf:type rdf:property .
```

Property: Tags that express the relationship between resources belong to the property class. For example, ex:graduatedFrom can be thought of as a resource because it has a URI, but in fact, although the human brain is able to deduce that it is a marker that expresses the relationship between two resource is given; reference is made to the class **rdf:property** to make it machine-processable as shown above.

```
@prefix ex:  http://example.org/  .
"Mr. X" ex:graduatedFrom ex:METU .
ex:graduatedFrom rdf:type rdf:property .
ex:studiedIn rdf:type rdf:property .
```

Subproperty: Suppose for example that we have an RDF document, as shown above. When we query this document to see all people registered in the system with the query "ex:studiedIn", "Mr X" individual will not be included among the results. In fact, the human brain can deduce "ex:graduatedFrom" relationship involves "ex:studiedIn", but computers can not make the same deduction directly. For this reason, it is necessary to specify the relations of the properties between each other on the RDFS document. For this, it is necessary to state that the relation ex:graduatedFrom involves ex:studiedIn relation in RDF document by using **rdfs:subPropertyOf** tag as follows.

```
@prefix ex:  http://example.org/  .
"Mr. X" ex:graduatedFrom ex:METU .
ex:graduatedFrom rdf:type rdf:property .
ex:studiedIn rdf:type rdf:property .
ex:graduatedFrom rdfs:subPropertyOf ex:studiedIn .
```

Domain and Range: Domain and range are used to describe the class to which the subject and object respectively using that property belongs. For example, as shown below, the human brain can understand that the relationship of "ex:graduatedFrom" is between a human and a university, but in order for the machine to understand this, reference should be made to the "ex:Person" and "ex:University" classes using the **rdfs:domain** and **rdfs:range**.

```
@prefix ex:  http://example.org/  .
"Mr. X" ex:graduatedFrom ex:METU .
ex:graduatedFrom rdf:type rdf:property .
ex:graduatedFrom rdfs:domain ex:Person .
ex:graduatedFrom rdfs:range ex:University .
```

Other Relations: When describing a new resource, adding human-readable explanations using natural language will be useful to increase comprehensibility. Therefore, **rdfs:comment** is used. On the other hand, **rdfs:seeAlso** is used to access detailed information about a resource.

2.1.2 Fuzzy Set Theory

Fuzzy set theory is a theory that aims to replace human inference and thereby solve complex decision-making problems. Foundations were introduced by L. A. Zadeh in 1965. Systems such as weather forecasting and disease diagnosis systems can be cited as examples of some decision-making problems that require the use of fuzzy logic. The data to be used in such decision making systems does not express accurate and precise results. Crisp logic is used to express these values in Boolean logic. For example, a crisp value is absolutely either true or false. There can be no value between these two. For example, the period during which a person is detected near

the border station is called "Dangerous" or "Safe" and is classified in the following image control function sets:

$Safe = \{x \mid \text{for all } x, \text{time} < 15{:}00 \text{ PM } \}$

$Dangerous = \{x \mid \text{for all } x, \text{time} >= 15{:}00 \text{ PM } \}$

The aforementioned sets belong to the safe group of persons identified near the border station before 15:00, while those identified after 15:00 belong to the category of dangerous persons. But a crisp logic presents important limitations. For example, the detections made at 8:00 pm and 14:59 pm, despite the difference of about 7 hours between them, belonged to the same crisp set "Safe", while a determination was made at 15:00 pm although it was a difference of one (1) minute it's in the dangerous category. That's why crisp sets are not the right way to classify the time of day. For this reason, fuzzy set theory is used to eliminate the limitations of crisp logic. In fuzzy set theory, each element is a member of a set with a certain degree of membership (u). In other words, the degree of belonging to a crisp set is $u \in \{0, 1\}$, but the degree of membership in a fuzzy set is $u \in [0, 1]$.

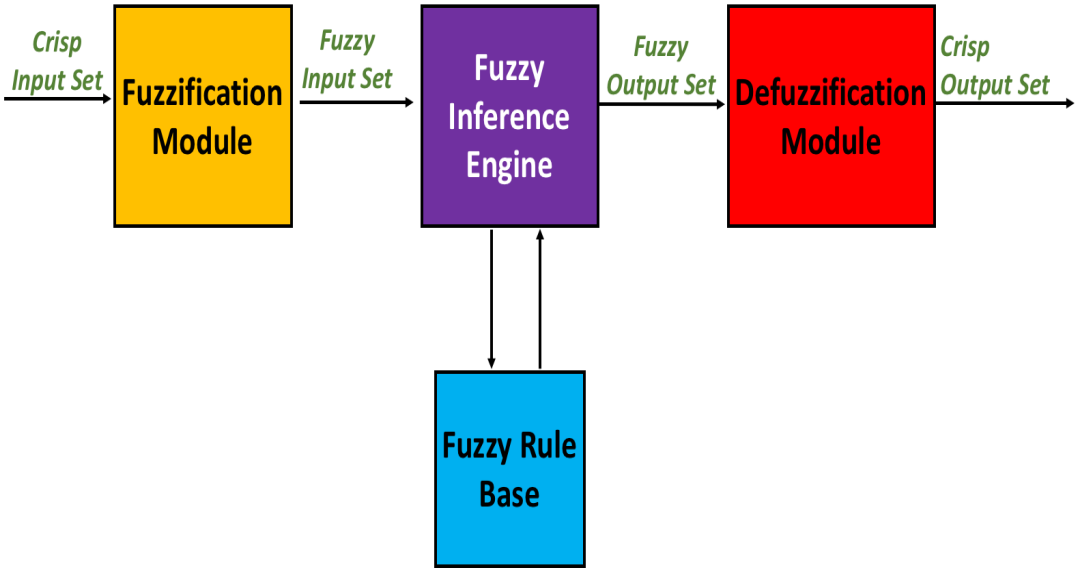


Figure 2.2: A general schema of a fuzzy logic controller

While F is a fuzzy set and X is a crisp set that belongs to this set, the membership function is shown as u_f: $X \rightarrow [0, 1]$. In this case, the F set consists of a sequential

pair of crisp input x and the corresponding fuzzy membership value u_f [21].

$$F = \{(x, u_f(x)) : \forall x \in X\} \tag{2.1}$$

The fuzzy logic controller system uses crisp sets to make decision-making mechanisms more accurate. In order to achieve this, the X crisp set received as input is transformed into Y fuzzy set after processing $x \in X$. Fuzzy Logic Controller, as shown in Figure 2.2, consists of 4 main components, Fuzzification, Fuzzy Rule Base, Fuzzy Inference Scheme and Defuzzification.

Fuzzification: It is about taking the crisp input values of all the variables that the system needs at the time of decision making and converting them into fuzzy sets by placing them in a membership function. The fuzzifier is responsible for the fact that each crisp input value is received and given to the corresponding membership function. The fuzzy set elements that will be entered for the fuzzy inference engine are called state variables [22], while the values they receive are called fuzzy linguistic labels. For example, the "distance to border" described in Chapter 4 is a state variable, while the words "very close", "close", and "far" are referred to as fuzzy linguistic labels. Each linguistic label has a corresponding membership function. The commonly used forms of functions are triangular, trapezoidal, Z-shaped, sigmoidal, Bell and Gaussian shapes. In the surveillance application performed in this thesis, membership functions for each different crisp input dataset are described in Chapter 4.

Fuzzy Rule Base: Fuzzy Rule Base is a set of rules created by using logical AND, OR operators, with state variables and linguistic labels. Using these rules, the linguistic labels of the output state variable and their degree of membership are calculated. For example, the R_i fuzzy rule taking m input and producing an output is defined as follows:

$$R_i : IF(\ I_1\ is\ A_{i1}\ AND\ I_2\ is\ A_{i2}\ AND\ ...\ AND\ I_m\ is\ A_{im}\ )\ THEN\ (\ O\ is\ B_i\ ) \tag{2.2}$$

where $\{I_1, I_2, \ldots, I_m\}$ are the input state variables, O is output state variable, $\{A_{i1}, A_{i2}, \ldots, A_{im}\}$ are the input linguistic labels and B_i is the output linguistic label.

Fuzzy Inference Scheme: Following fuzzification, the state variables, obtained as a fuzzy set, are taken with related linguistic label and its membership value pairs as input. Then, those pairs are processed with the fuzzy rule base and fuzzy output sets are produced [22]. To achieve this, output linguistic labels and their degrees of membership in a set of fuzzy output are calculated using equations 2.1 and 2.2 as follows:

$$eval(R_i) : (u_{Ai1}(I_1)\ AND\ u_{Ai2}(I_2)\ AND\ ...\ ANDu_{Aim}(I_m)) \rightarrow u_{Bi}(O) \quad (2.3)$$

The fuzzy output to be obtained as a result of the matching rules calculates the membership value of each linguistic label in the fuzzy output set using T-operators[23], such as MIN, MAX, NOT, instead of AND, OR, NOT as follows:

$$u_{Ai}(I_i)\ AND\ u_{Aj}(I_j) = u_{A_i \cap A_j}(I_{ij}) = MIN(u_{Ai}(I_i), u_{Aj}(I_j)) \quad (2.4)$$

$$u_{Ai}(I_i)\ OR\ u_{Aj}(I_j) = u_{A_i \cup A_j}(I_{ij}) = MAX(u_{Ai}(I_i), u_{Aj}(I_j)) \quad (2.5)$$

$$NOT(u_{Ai}(I_i)) = u_{Ai}(I_i) = 1 - u_{Ai}(I_i) \quad (2.6)$$

Defuzzification: The produced fuzzy output linguistic variables from the fuzzy inference scheme arrive in the defuzzification module with membership degrees. At this point, all fuzzy outputs are aggregated into one crisp output by Defuzzifier. To perform defuzzification, many methods are available, such as maximum membership, centroid, weighted average, mean max membership [24]. In this thesis, the centroid method is preferred. Centroid, in other words, the center of gravity method can be

formulated as follows:

$$z^* = \frac{\sum_{a=1}^{N}(z_a) * u_c(z_a)}{\sum_{a=1}^{N} u_c(z_a)} \tag{2.7}$$

where $u_c(z_a)$ is the modified membership functions of fuzzy output set C, $\sum$ denotes the summation and N is the number of points the modified fuzzy output span over X-axis.

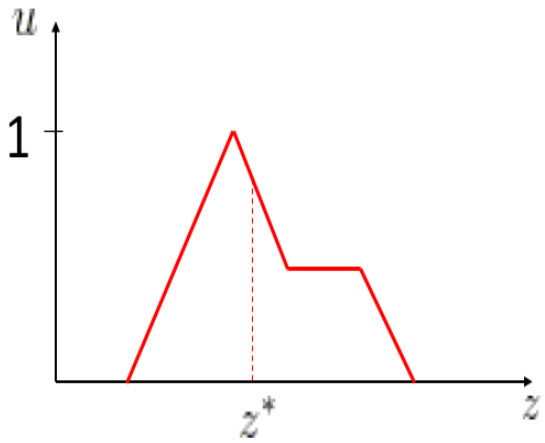


Figure 2.3: Center of gravity defuzzification method

2.1.3 Semantic Web Reasoning with Fuzzy Set Theory Augmentation

Through the semantic Web, reasoning can be implemented by directly defining your own rules, as well as by using RDFS tags. In this way, new information is deduced. However, the accuracy of the data obtained is 100%, given the nature of the semantic Web. However, unlike binary reasoning, fuzzy set theory states that a judicial power is correct according to a certain degree of membership. As part of the study conducted in this thesis, the goal is to produce results closer to human inference by increasing semantic web reasoning with fuzzy set theory. To achieve this, the input and output membership functions generated using fuzzy set theory have been implemented in reasoning rules on the semantic Web. Then, the results produced were saved in tuples as literal values to match the semantic Web tuple format. As part of this study, a separate ontology is developed to record the membership values obtained.

2.2 Related Work

In the last decade, semantic Web technologies are integrated with wireless sensor networks for different purposes such as activity detection, data management [2], raw data processing [25], communication of different wireless sensor networks [3], analysis of sensor networks and their data [26] , optimization of energy consumption of sensor networks [27] or estimation of fire time indices [28]. In this thesis, the purpose of using the semantic Web with wireless multimedia sensor networks is to use the reasoning capabilities offered by the semantic Web during activity detection and to open up the reasoning results obtained in the semantic Web. These results are for external world, allowing other users to use this data with the use of semantic Web ontology. The difference between this study and other applications that allow activity detection using the semantic Web increases semantic Web reasoning with fuzzy logic. Thus, the results of this theory are relatively more intuitive and close to human reasoning.

Activity detection can be done using semantic Web technology as well as other methods. For example in order to make activity learning, study [29] was implemented by applying dynamic K-means clustering. By introducing a smart home system; in order to measure the orientation, they place a PIR motion sensor in various parts of the house and places an object sensor on some objects, such as a telephone. It defines seven activities consisting of brooming, cooking, washing, cleaning, eating, sleeping and talking. Using K-Means Clustering, the first four activities are defined to the system by passive learning and the last three activities are expected to be learned by the system. Moreover, the study given in [30] focuses on voice activity detection and uses the SVM and GMM classifiers to label and classify acoustic event sounds (applause, cup sound, etc.). After classification, voice activity detection is done by separating acoustic event sounds from speech. Finally, study [31] focuses on activity monitoring within the smart home systems that emerged as a result of the widespread use of health-care monitoring applications. In general, smart home activity monitoring applications uses supervised activity detection. On the other hand, study [31] focuses on unsupervised activity recognition because of the fact that; it is very expensive to make manual labeling and violates the privacy of the personal life of the individuals for applications using supervised activity detection. In order to perform unsupervised

activity recognition with the framework called NECTAR, a hybrid approach consisting of ontology and segmentation was introduced. Segments, via feedbacks from the environment to avoid errors due to heterogeneous environmental conditions, consist of sensor events that cannot be defined by ontology.

A study by Zafeiropoulos, Anastasios, et al. investigates data management on sensor networks using semantic web technologies[1]. The problem pointed out by this study is that due to the lack of ontological definitions, end-users cannot produce the required data by using raw data produced by the sensor networks. As a solution, as seen in Figure 2.4 3-Layered architecture is presented: A data layer that collects and groups this data, a processing layer that processes this grouped data and finally, semantic layer that responsible for context annotation and reasoning [1]. The study of Zafeiropoulos et al. covers only the classical semantic web rule-based reasoning techniques without fuzzy set theory while detecting activities.

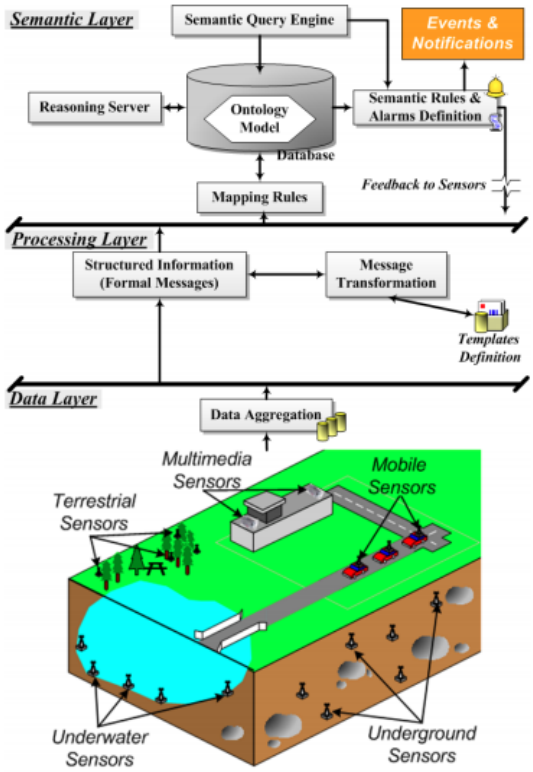


Figure 2.4: Three layered structure for data management in sensor networks[1]

Similar to the study described in [1], Moraru et al. have also established a framework

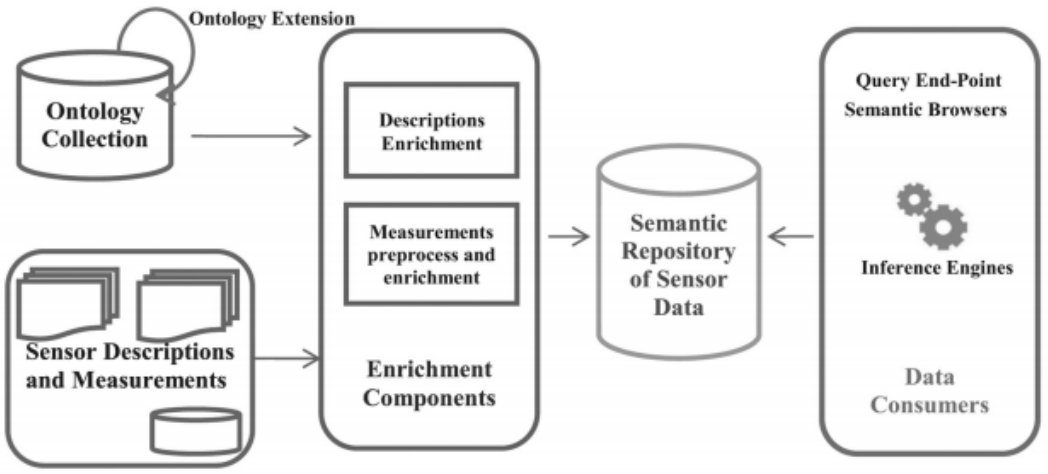


Figure 2.5: A framework for transforming and publishing sensor measurements[2]

that automatically converts sensor raw data to a comprehensive semantic sensor data in [2]. Moraru et al. aims to automatically annotate sensor data, publish sensor data using well-known vocabulary and support event detection applications by applying reasoning techniques. Therefore, they present a concept framework shown in Figure 2.5. On the other hand, the study presented in this thesis uses semantic web reasoning and fuzzy set theory together and increases the reliability of detected activities. Moreover, instead of annotating raw sensor data, our study focuses on publishing processed sensor data acquired from the sink of a WMSN.

One of the goals of using the semantic Web with wireless multimedia sensor networks is, by keeping the data in RDF format, to use processed data acquired from WMSN. Contrary to the study conducted in this thesis, the reasoning using fuzzy set theory is not the focus of studies [15] and [3]. In fact, these studies do not focus on activity detection. Moreover, these studies concentrate on using different types of wireless sensor networks and semantic Web services. For example, the study specified in [3] introduces two approaches that allow different types of sensor networks to communicate. The first method is that each node can have its own semantic Web Service. It is not necessary to have a generic web service to publish the data of the node. The second method is to have a base station node and to deploy the Semantic Web Service on this node to communicate with different sensor networks. In this case, as shown in Figure 2.6, requests are issued by end-users using third-party software, the

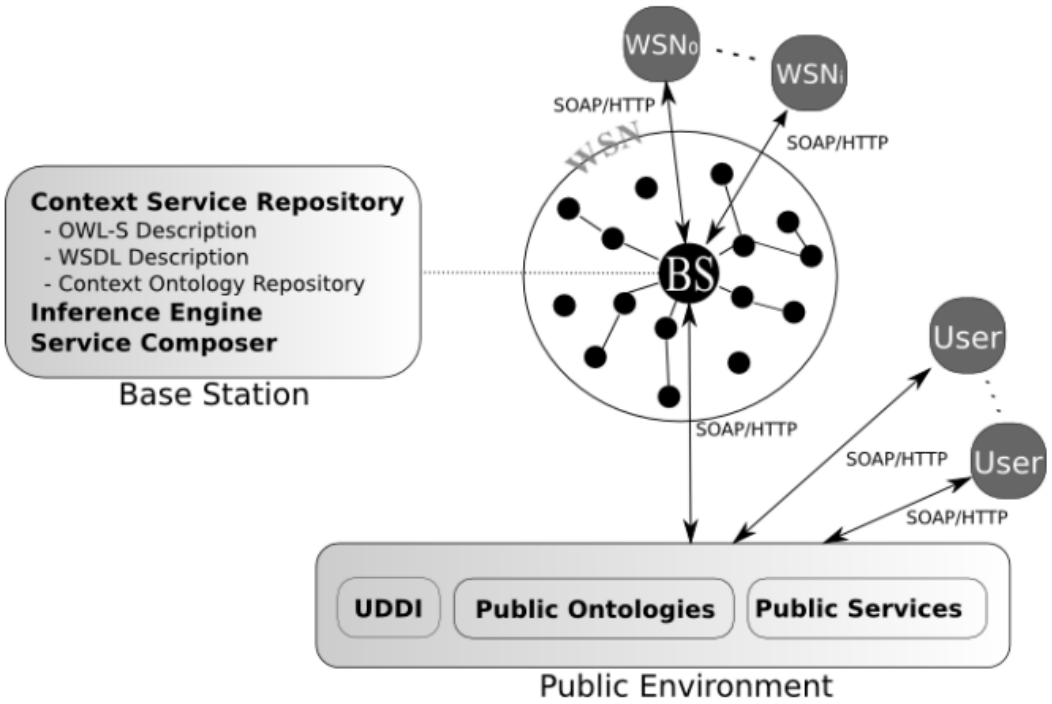


Figure 2.6: Interaction of WSNs and end-users[3]

corresponding response is retrieved from the base station node and forwarded to the end-user [3]. As in the work specified in [3], between different applications, [32] also aims to reuse the data produced by different types of sensor networks that perform measurements such as sound, location, and temperature between different applications. Therefore, for all types of sensor networks, the raw sensor data is converted to semantic sensor data in accordance with the ontology described previously.

Kapitanova et al. put forward a study that makes event detection in WSNs with fuzzy set theory[16]. Furthermore, their study aims to tolerate data from untrusted and low-precision sensors. To achieve this, sensor data is fuzzified without applying semantic Web techniques to obtain a membership degree. On the other hand, the study in this thesis uses the semantic Web with fuzzy logic to create rules for reasoning. Through the semantic Web approach, processed sensor data acquired from the sink of WMSN and associated reasoning results can be opened to the outside world. However, study [16] focuses on reducing the growing number of rules resulting from the use of fuzzy set theory. Because if fuzzy set theory is used, the number of rules will automatically reach m^n for m different values that n different variables can take.

CHAPTER 3

FUZZY LOGIC ON SEMANTIC WEB REASONING FOR ACTIVITY DETECTION IN WIRELESS MULTIMEDIA SENSOR NETWORKS

The goal of this study is to design a reliable reasoning system to be used in activity monitoring applications and make acquired data available for other users or applications. To achieve this goal, semantic Web technologies extended with the fuzzy set theory is integrated with a WMSN which enables environmental monitoring in an efficient way. The theory of fuzzy sets is necessary for reasoning systems to obtain a reasoning mechanism close to the human reasoning. The architecture presented in this thesis according to requirements is composed of three (3) main layers, as shown in Figure 3.1. According to this architecture; the sensors, the living beings and their environment form the layer of sensors. The raw data sensed by these sensors is processed at the sensor nodes and/or sink to determine whether there is a target object in the surveillance area. These data are then stored in the data layer. Finally, these processed data are transferred to the semantic Web layer. Data transformation process between layers is shown in Figure 3.2. The semantic Web layer manages the entire semantic Web and its fuzzy reasoning process for activity detection. It is extended with fuzzy sets theory to obtain a reasoning mechanism close to the human reasoning.

3.1 Sensor Layer

The sensor layer is the lowest layer of architecture. In this layer, by distributing sensor nodes of equal distance in a selected area, environmental conditions and living things in the environment are monitored to recognize objects using the established wireless multimedia sensor network. As shown on Figure 3.3, Sensor layer's wireless mul-

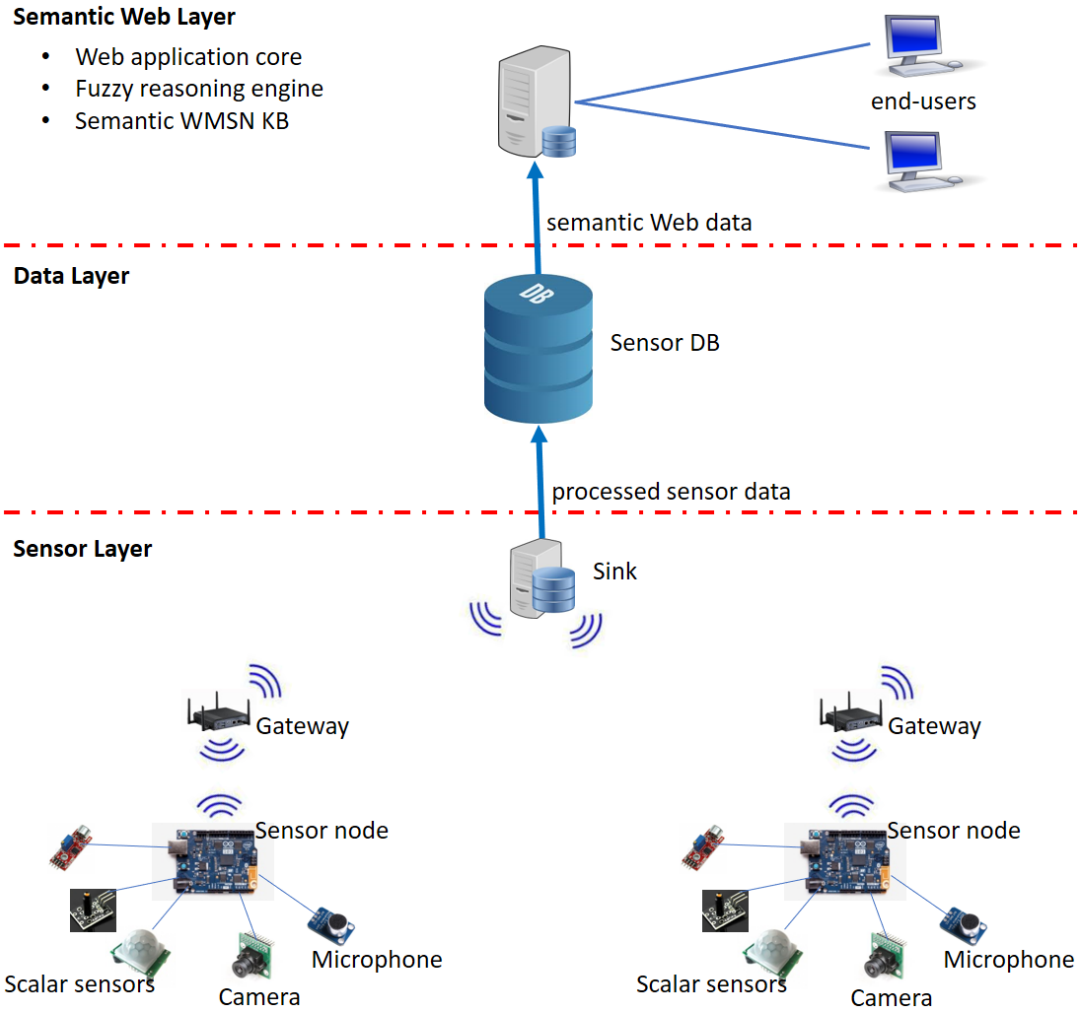


Figure 3.1: Activity detection architecture on wireless multimedia sensor networks

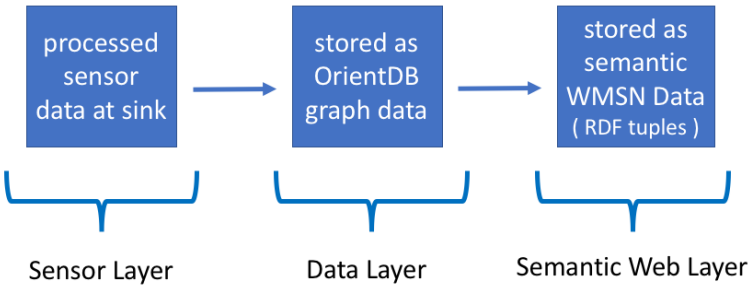


Figure 3.2: Data transformation flow

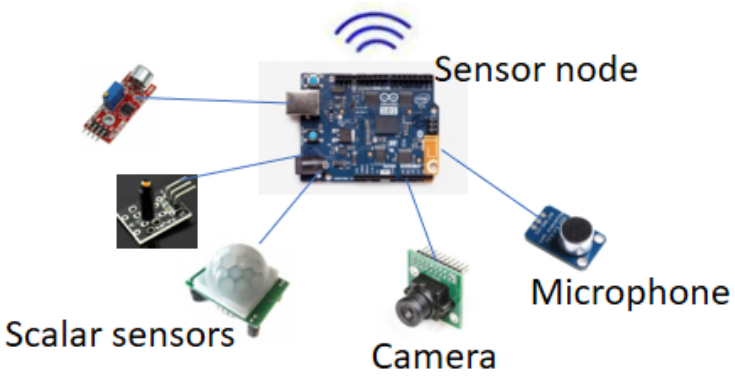


Figure 3.3: Wireless multimedia sensor node architecture

timedia sensor node architecture consists of multimedia sensors including cameras, microphones and scalar sensors including passive infrared (PIR), acoustic and vibratory sensors. Normally, multimedia sensors are kept in sleep mode to decrease energy consumption. On the other hand, scalar sensors are always in active mode. Cameras and microphones are decided to be awakened according to the scalar sensors' measurements. When the camera and microphone are turned on, first of all, both type of audio and video are processed using automatic learning methods and fused at the sensor node. Then, generated information is transferred to the sink via gateways and more complex fusion is performed here to accurately detect and recognize objects in the monitored area of established WMSN shown in Figure 3.4. This processed sensor data is transferred to the data layer, located in the upper layer of the architecture.

3.2 Data Layer

The pre-processed sensor data taken from the sensor layer is stored in this layer and to be used in the semantic Web layer. The data stored in this layer include information about sensed entities such as entity type, timestamp and information about sensors such as their positions. Entities are stored in the sensor DB which works with a data layer server. As an example, the data stored in the data layer for the surveillance application described in Chapter 4 contains the following items:

- Latitude,

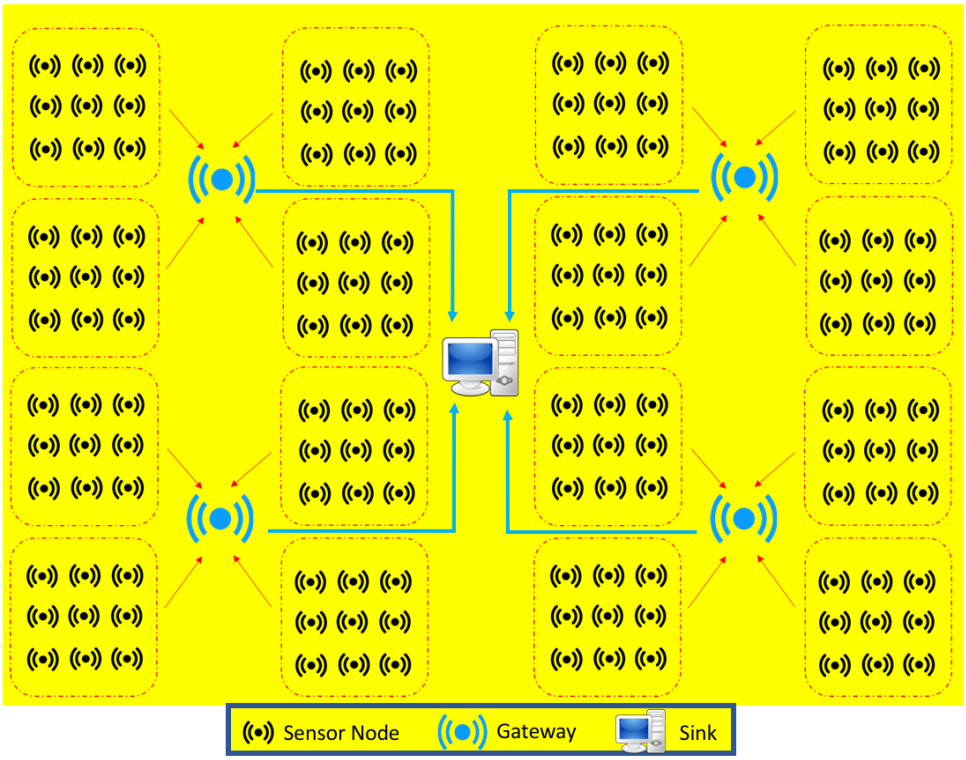


Figure 3.4: Wireless multimedia sensor network architecture

- Longitude,
- Altitude,
- Sensors in the neighborhood,
- Timestamp,
- Type of entity detected.

The position of the sensor expressed with latitude, longitude and altitude, and the type of entity with time stamp play an important role in the event detection performed by the semantic Web layer. The type of entity can be, for example, a person, a dog or a car. The data layer server sends a notification to the Semantic Web layer about arrival of a new WMSN entry and triggers the operation of the semantic Web layer.

3.3 Semantic Web Layer

This layer is responsible for managing all operations associated with the semantic Web. In semantic Web layer, client requests are processed, components are initialized and executed. This layer has three main components: Web application core, semantic WMSN KB and fuzzy semantic reasoning engine. Activity detection is performed on this layer. The semantic Web layer converts processed sensor data taken from the data layer into semantic tuples (subject, predicate, and object) in accordance with designed ontology. In addition, Semantic Web layer is required to check the semantic sensor database for the presence of new fuzzy semantic reasoning data.

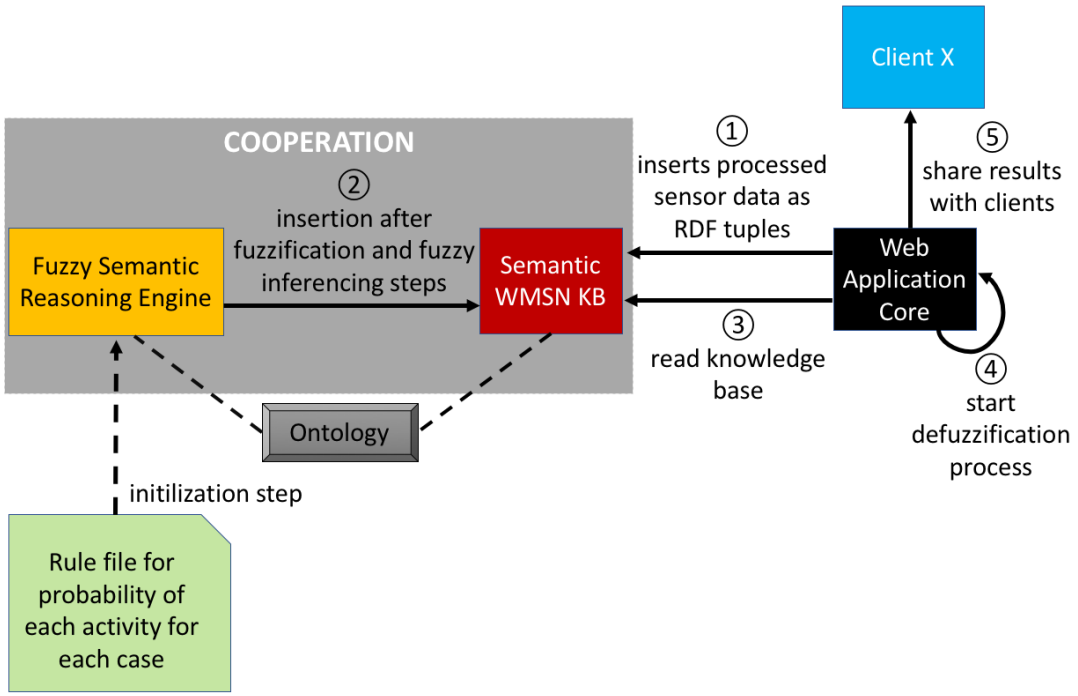


Figure 3.5: Coordination of components in semantic Web layer

Figure 3.5 shows the coordination between semantic Web layer components while performing activity detection. First of all, Web application core converts processed sensor data to RDF tuples and then inserts them into semantic WMSN KB. Secondly, as soon as a new insertion to knowledge base happens, fuzzy semantic reasoning engine is activated and fuzzification and inferencing is performed. Then Web application core reads knowledge base and starts defuzzification. Finally defuzzification results are shared with clients.

3.3.1 Web Application Core

The Web application core, works on a server, manages and coordinates the entire process of the semantic Web layer. The communication between the data layer and the semantic Web layer is bidirectional. The initialization of communication depends on a setting in the Web application core. In online mode, the instantaneous WMSN data is monitored and the sensor DB located in the data layer warns the Web application core to inform about the availability of new WMSN data. In offline operation mode, the Web application core initiates communication and request previously stored WMSN data. Communication with end-users are also managed by this application.

In Algorithm 1, between server and client application, established communication in order to monitor the WMSN environment instantly is observed. To be informed that newly processed sensor data has been added to sensor DB by the sink, it is firstly subscribed to the database via data layer server. After that, some indicators are checked and the algorithm is routed. For example, thanks to this algorithm, the entities that are sensed from the sensors are shown to the users with their instantaneous positions. In addition, how many entities are detected for each entity type and the inferences obtained with semantic web reasoning are shared with the client via this algorithm. As for the computational complexity of this algorithm, $generateGraphicsInnerContent$ has $O(n^2)$, $findNumberOfEntityTypes$ and $getFuzzyReasoningResults$ have $O(n)$ complexity. Therefore, the algorithm complexity is calculated as $O(1) * MAX(O(n), O(n^2)) = O(n^2)$ because of the functions used in this algorithm.

In Algorithm 2, between server and client application, it is observed that a communication is established in order to monitor activities occurred previously in the WMSN environment. Here, as in Algorithm 1, the algorithm is routed by checking some flags and the entities that are sensed from the sensors at the selected date are shown to the users with their instant positions. In addition, how many entities are detected for each entity type and the inferences obtained with semantic Web reasoning are shared with the client via this algorithm. As for algorithm complexity, $generateGraphicsInnerContent$ has $O(n^2)$ complexity. Therefore, the algorithm complexity is calculated as $O(n) * MAX(O(n), O(n^2)) = O(n^3)$ because of the

Algorithm 1 Request handling mechanism for live sensor data

procedure HANDLEREQUESTSFORLIVEDATA( ) ▷ Currently acquired data

$subscribeToSensorDB()$ ▷ in Data Layer, for sensor and its data

if $sendVisualContent = true$ **then**

$sendVisualContent \leftarrow false$

$response.EventType \leftarrow$ "$svgUpdate$"

if $waitForSensorListSemaphore()$ **then**

$takeSensorDataListSemaphore()$

end if

$response.Data \leftarrow generateGraphicsInnerContent()$

$releaseSensorDataListSemaphore()$

else

$sendVisualContent \leftarrow true$

$findNumberOfEntityTypes()$

if $chosenConceptToSend = ENTITY\_TYPE\_X$ **then**

$chosenConceptToSend \leftarrow ENTITY\_TYPE\_Y$

$response.EventType \leftarrow$ "$numOfEntityTypeX$"

$response.Data \leftarrow numOfEntityTypeX$

else if $chosenConceptToSend = ENTITY\_TYPE\_Y$ **then**

$chosenConceptToSend \leftarrow SEM\_FUZZY\_REASONING$

$response.EventType \leftarrow$ "$numOfEntityTypeY$"

$response.Data \leftarrow numOfEntityTypeY$

else if $chosenConceptToSend = SEM\_FUZZY\_REASONING$ **then**

$chosenConceptToSend \leftarrow ENTITY\_TYPE\_X$

$response.EventType \leftarrow$ "$semanticFuzzyReasoning$"

$response.Data \leftarrow getFuzzyReasoningResults()$

end if

end if

$sendResponse()$

end procedure

Algorithm 2 Request handling mechanism for offline sensor data

procedure HANDLEREQUESTSFOROFFLINEDATA($requestedDay$) ▷

Previously acquired data

$getPreviousSimulationList()$ ▷ from SensorDB in Data Layer

while $i \neq previousSimulationListSize$ **do** ▷ Look up each sim. entity

if $previosSimulationList[i].name = requestedDay$ **then**

if $requestedParameter = "numOfEntityTypeX"$ **then**

$response.Data \leftarrow numOfEntityTypeX$

else if $requestedParameter = "numOfEntityTypeY"$ **then**

$response.Data \leftarrow numOfEntityTypeY$

else if $requestedParameter = "semanticFuzzyReasoning"$ **then**

$response.Data \leftarrow getFuzzyReasoningResults()$

else if $requestedParameter = "visualContent"$ **then**

$response.Data \leftarrow generateGraphicsInnerContent()$

end if

end if

$i \leftarrow i + 1$

end while

$sendResponse()$

end procedure

functions used in this algorithm.

In summary, algorithms 1 and 2 are required to provide the data requested by the client side. The previously obtained and processed sensor data is extracted from the sensor database in the data layer, which is different from that of Algorithm 1. In addition, since the instantaneous processed sensor data is examined in Algorithm 1, different from Algorithm 2, registering in the sensor database of the data layer, it is possible to obtain data layer notifications if newly processed sensor data is obtained. The newly processed sensor data obtained by this notification is converted to semantic WMSN data in the semantic Web layer. In addition, the membership values of the output linguistic labels obtained from the fuzzification and fuzzy inference applied by the fuzzy semantic reasoning engine to these semantic sensor data are periodically checked

from the semantic WMSN knowledge base. Interactions with semantic WMSN KB must be compatible with designed ontology. Moreover, defuzzification process are executed using the $getFuzzyReasoningResults$ function by periodically checking semantic WMSN KB. With defuzzification, the system performs activity detection according to the specified fuzzy rules. Defuzzification is performed using the center of gravity method with the equation specified in 2.7. Since activity detection is done using fuzzy set theory, reasoning is closer to human reasoning than crip logic.

3.3.2 Semantic Wireless Multimedia Sensor Network Knowledge Base

Semantic WMSN KB is a type of graph database containing RDF tuples. RDF, called the Resource Description Framework, is a concept that refers to the fact that data is stored on the Web. RDF is a concept that has been put forward with the Semantic Web idea and can be considered as a cluster containing triplets consisting of subjects, predicates and objects. For example, if the phrase "Mary lives in Istanbul" is converted to RDF, "Mary", "livesIn" and "Istanbul" refer to subject, predicate and object, respectively.

In this study, the purpose of using the semantic Web is to integrate this work with other studies in the future. With the idea of the semantic Web, RDF recommends linking subjects by moving the idea of Web 2.0 to link documents a step further [33]. RDF provides this by giving a unique resource identifier (URI) to each subject. Thus, if the same subject is mentioned on different web pages, the data of these two subjects can be related (linked) to each other. In addition, this study can be used to make processed sensor data acquired from the sink of WMSN available in a well-defined format with a typed ontology for any other systems to consume this data. This feature makes applications using the semantic Web extensible.

Semantic WMSN KB is based on an ontology shown in Figure 3.6, using RDFS. This ontology is designed not only to cover the data obtained from the sensor layer, but also to allow the input and output membership functions to be implemented in order to perform fuzzification and record the results in the form of triples compatible with the ontology. Thanks to this ontology, the data obtained from the sensor layer, which are essentially position information, the type of entity, time stamp information

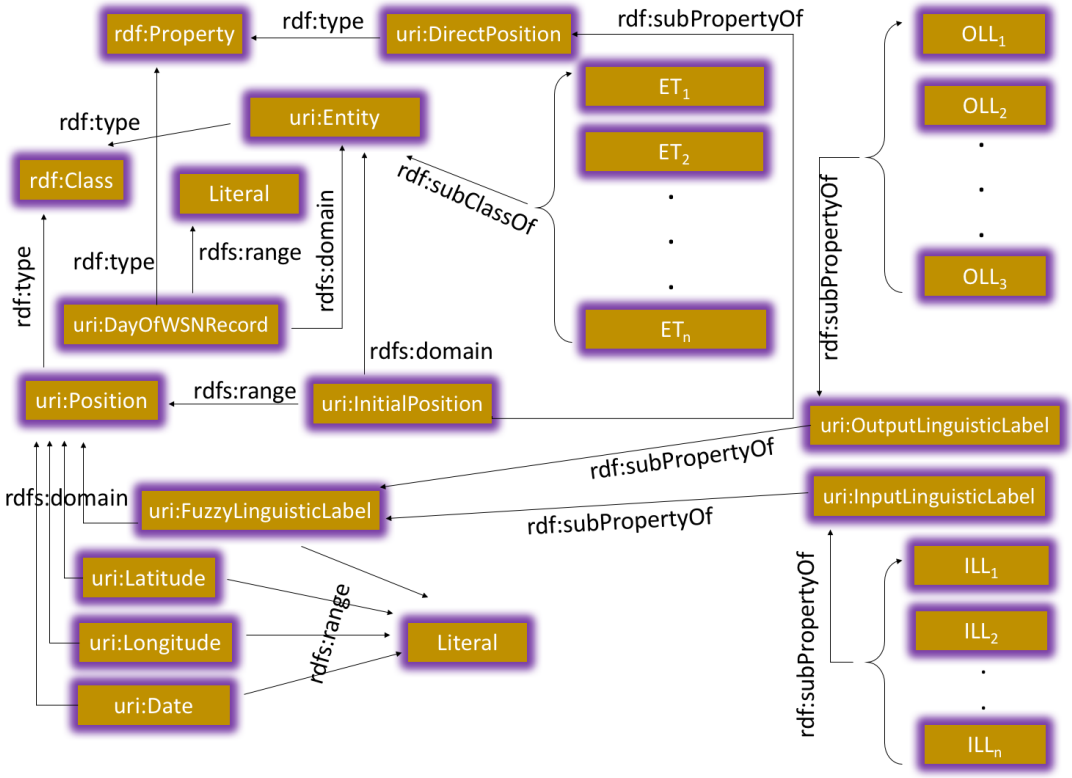


Figure 3.6: Ontology design of semantic WMSN KB

Figure 3.7: Semantic KB's domain of interest

of when the entity is detected by the sensor nodes as shown in Figure 3.7 are represented. Moreover, inferencing data obtained after the fuzzification and/or inferencing are stored on semantic WMSN KB in accordance with this ontology.

Update to the ontology is performed easily with a minimal code change on Web application core thanks to the generic architectural design of semantic Web layer. For example, if a new input linguistic label ILL_x is added to the ontology; new fuzzification rules should be added and inferencing rules should be updated on fuzzy semantic reasoning engine. There is no need to update anything on Web application core. Because Web application core is only responsible for defuzzification and not interested in input linguistic labels. If a new output linguistic label OLL_y is added to this ontology; there is no need to update or add new fuzzification rules. Inference rules should be updated and/or added. Moreover, defuzzification implementation in Web application core should be updated. Thanks to minimal code change of Web application core, the system has very low test and verification cost.

3.3.3 Fuzzy Semantic Reasoning Engine

Fuzzy semantic reasoning engine, one of the main components of this layer, is responsible for fuzzification and inferencing process. Reasoning process is the process of obtaining new semantic tuples from existing tuples. In other words, the reasoning process gives the system the ability to learn. In this study, fuzzy semantic reasoning engine automatically passes over tuples after each tuple is inserted into semantic WMSN KB in accordance with the ontology and tries to extract a new activity detection.

Fuzzy semantic reasoning engine is responsible for managing the fuzzification and inferencing process by implementing input membership functions and fuzzy inference rule set in accordance with the ontology on the engine, as opposed to classic rule-based reasoning of the semantic Web, as seen in Figure 3.8. Table 3.1 shows the fuzzy rule set architecture used by fuzzy inferencing scheme. In this table; $ILL_{0,1,..,n}$ refers to input linguistic labels of different input state variables; while $Activity_{0,1,..,n}$ refers to different values (eg: walking, running, etc.) that the output state variable can take, or, in other words, the type of activity to be detected (eg: movement)

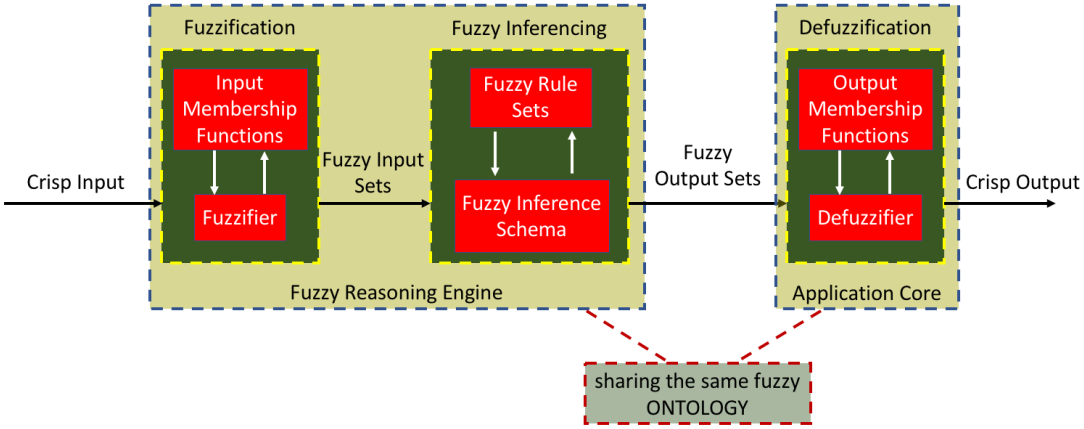


Figure 3.8: Fuzzy logic controller implementation on semantic Web reasoning

Table 3.1: Architecture of fuzzy rule base

Rule	**InputStateVar 0**	**InputStateVar 1**	**....**	**InputStateVar n**	**Activity**	**Confidence**
R_0	ILL_{00}	ILL_{10}	....	ILL_{n0}	$Activity_0$	c_0
R_1	ILL_{01}	ILL_{11}	....	ILL_{n1}	$Activity_1$	c_1
.	.	.	....	.	.	.
.	.	.	....	.	.	.
.	.	.	....	.	.	.
R_{a-2}	ILL_{0a-2}	ILL_{1a-2}	....	ILL_{na-2}	$Activity_{a-2}$	c_{a-2}
R_{a-1}	ILL_{0a-1}	ILL_{1a-1}	....	ILL_{na-1}	$Activity_{a-1}$	c_{a-1}
R_a	ILL_{0a}	ILL_{1a}	....	ILL_{na}	$Activity_a$	c_a

according to our theory. Furthermore, there is a confidence coefficient $c \in [0, 1]$ for each rule. Even if any rule is triggered, confidence parameter can reduce the effect of the rule by multiplying the detected activity's output membership degree with the triggered rule's confidence coefficient before defuzzification.

Figure 3.9 shows the insertion of crisp inputs in accordance with the ontology described in Figure 3.6 in the RDF tuple format. Prior to fuzzy inferencing, fuzzification should be performed by Web Application Core by passing crisp inputs(RDF tuples) as literal values into membership functions. Membership functions usually have trapezoidal, reverse trapezoidal or triangular form. The membership function equations with these forms are shown in Equations 3.1, 3.2 and 3.3. In these equations x represents crisp input data, a, b, c and d represent the range values in the membership functions. These equations are implemented within the membership func-

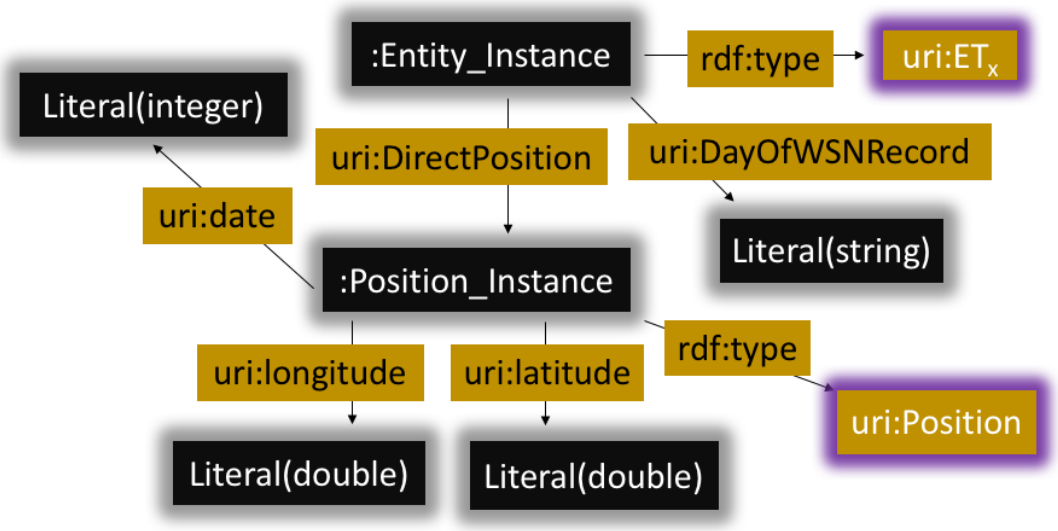


Figure 3.9: Crisp input insertion in accordance with the ontology

tion rules and the fuzzification is completed by using these equations to calculate the $ILL_{xy}(InputStateVar_x)$, $ILL_{tz}(InputStateVar_t)$ input linguistic labels and their membership degrees. For example, Figure 3.10 shows that how a trapezoidal membership function belonging to ILL_{xy} varies according to $uri : date$, is implemented in fuzzy reasoning engine.

$$Trapezoidal(x; a, b, c, d) = MAX(MIN(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}), 0) \tag{3.1}$$

$$ReverseTrapezoidal(x; a, b, c, d) = MIN(MAX(\frac{b-x}{b-a}, 0, \frac{x-c}{d-c}), 1) \tag{3.2}$$

$$Triangular(x; a, b, c) = MAX(MIN(\frac{x-a}{b-a}, \frac{c-x}{c-b}), 0) \tag{3.3}$$

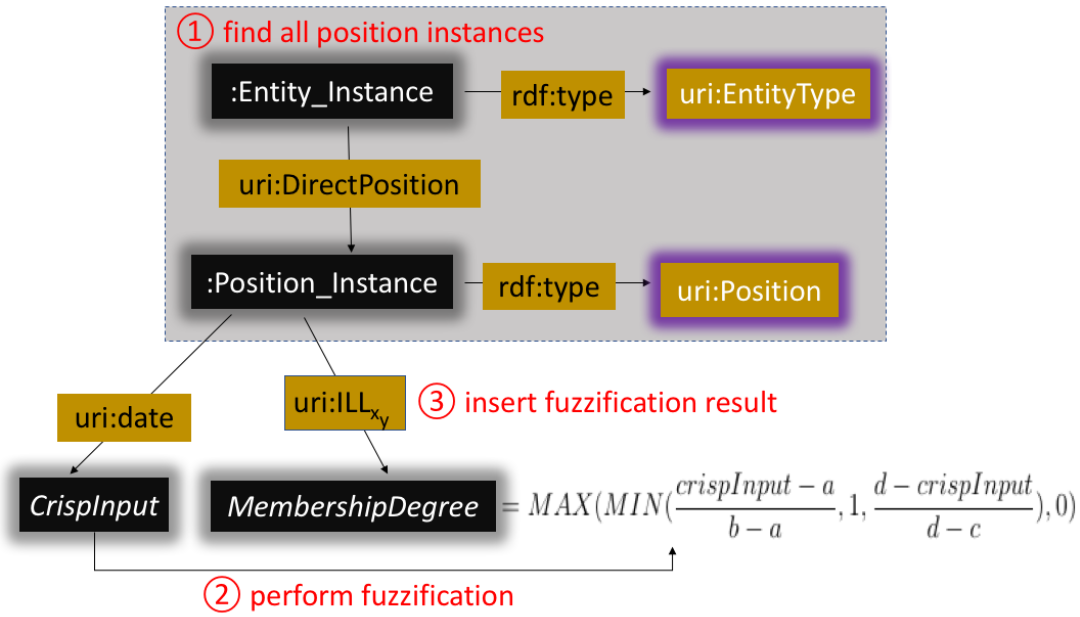


Figure 3.10: Fuzzification with the ontology

```
@prefix uri:  http://www.example.org/   .
[ ILL_xy(MembershipFunctionOfInputStateVar_x):
( ?entityId uri:DirectPosition ?pos ) ,
( ?pos uri:date ?timestamp ) ,

difference(?timestamp, a, ?diff1 ),
difference(b, a, ?diff2 ),
quotient(?diff1, ?diff2, ?val1 ),

difference(d, ?timestamp, ?diff3 ),
difference(d, c, ?diff4 ),
quotient(?diff3, ?diff4, ?val2 ),

min(?val1, ?val2, ?val3 ),
min(1.0, ?val3, ?val4 ),
max(?val4, 0.0, ?membershipDegree )
notEqual( ?membershipDegree, 0.0 )

→ ( ?pos uri:ILL_xy ?membershipDegree )
]
```

After fuzzification, the fuzzy inferencing process starts with the fuzzy rule set given

in Table 3.1. In this process, activities and related membership degrees are calculated with one or many triggered $R_{firedRule}$ rules by using the Equation3.4. Figure 3.11 shows the implementation of Equation 3.4 in Fuzzy Reasoning Engine.

$$\begin{aligned} eval(R_{firedRule}) : & MIN(ILL_{xy}(InputStateVar_x)\ AND\ ILL_{tz}(InputStateVar_t)) \\ & * Confidence_{firedRule} \\ & \rightarrow OutputStateVar(Activity_m) \end{aligned} \tag{3.4}$$

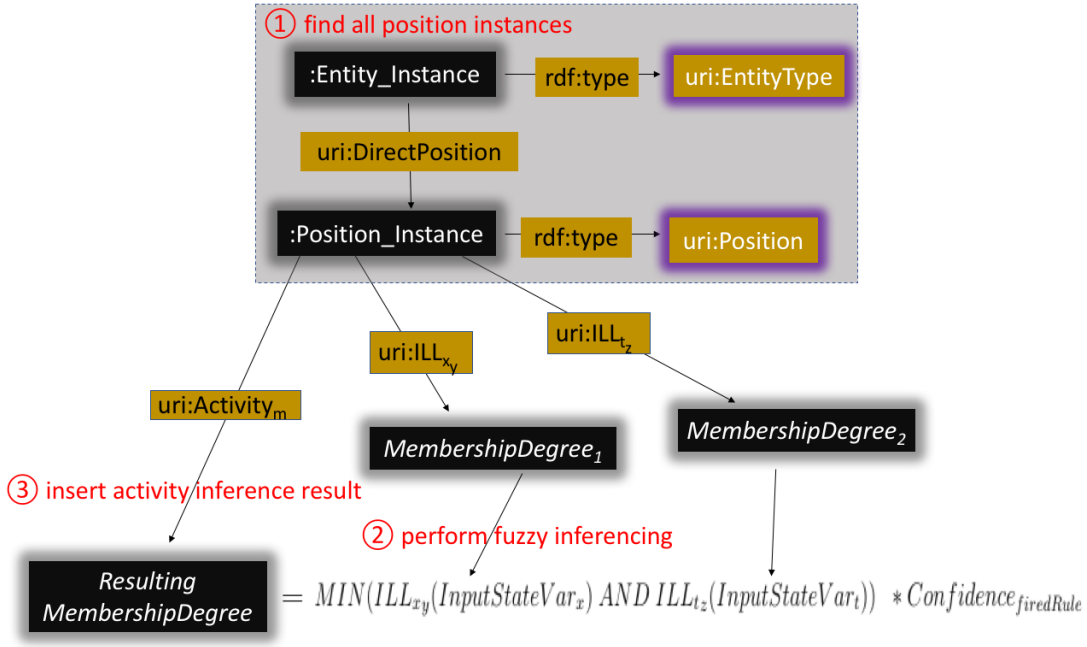


Figure 3.11: Fuzzy inferencing with the ontology

```
@prefix metu:  http://www.example.org/   .
[ eval(R_firedRule):
( ?pos metu:type metu:Position ) ,
( ?pos metu:ILL_xy ?membershipDegree1 ) ,
( ?pos metu:ILL_tz ?membershipDegree2 ) ,
min( ?membershipDegree1, ?membershipDegree2,
?minimumOfMembershipDegrees ),
product( ?minimumOfMembershipDegrees, Confidence_firedRule ,
?resultingMembershipDegree )
→ ( ?pos metu:Activity_m ?resultingMembershipDegree )
]
```

By applying the method described above, the semantic Web's reasoning process, known as binary or in other words rule-based, is transformed into heuristic reasoning. Thanks to heuristic reasoning, the reliability of deduction system is increased. Fuzzy set theory, together with the reasoning process, gives the system the ability to think like a human. The reasoning rules must be given to the fuzzy semantic reasoning engine during the initialization phase for reasoning with semantic Web.

To summarize, fuzzy semantic reasoning engine is used to implement input membership functions with the rules defined on it. Input membership functions are used for implementing fuzzification. In addition, inference processes which are made after fuzzification and input to defuzzification process are also fulfilled by the rules defined for this purpose on the engine.

CHAPTER 4

CASE STUDY: AN EXAMPLE SURVEILLANCE APPLICATION

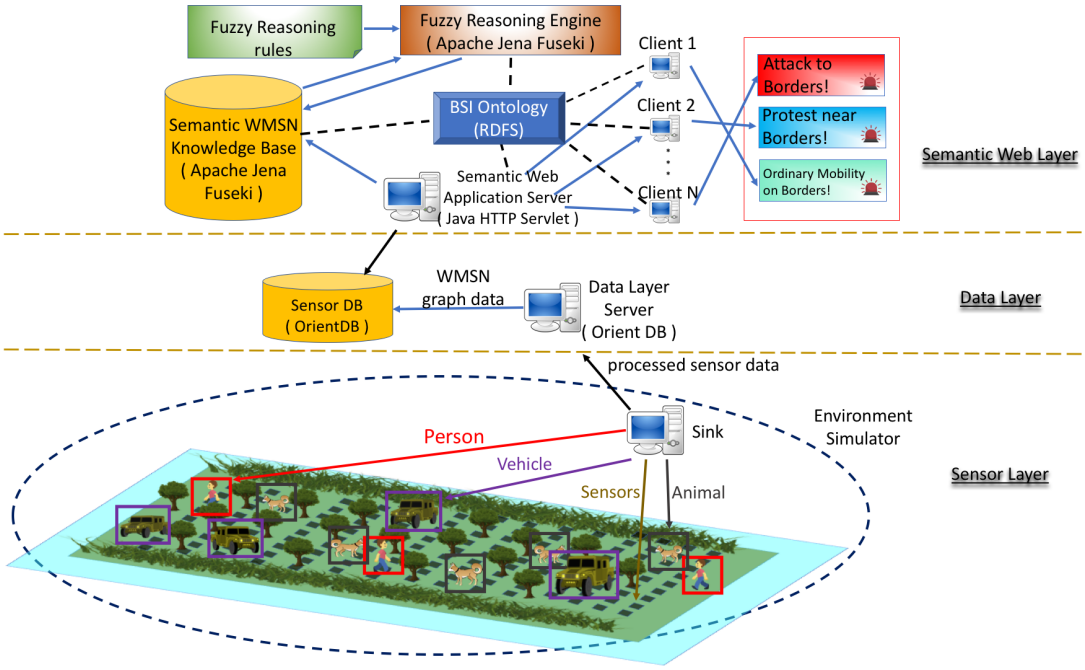


Figure 4.1: Surveillance application architecture

National security is one of the most important issues of this century. The soldiers at border stations who use binoculars or radar to monitor the environment can detect terrorist attacks at national borders. The problem here is that it will take hours or even days to fix it when the radar breaks down. It is precisely at this moment that the soldiers who protect the borders can face a sudden terrorist attack or some other activities. The surveillance application called Border Safety Informer, uses the theory described in Chapter 3 to solve this problem, instead of using a radar. As a result, it establishes a synthetic wireless multimedia sensor network of hundreds of sensor nodes around the border station and takes processed sensor data produced from the

sink. Its purpose is to monitor the environment with the system in place and to inform troops waiting at the border post by calculating the percentage of danger. In order to achieve this goal, the activity detection architecture defined in Figure 3.1 is implemented with the software components to form the structure of Figure 4.1.

4.1 Sensor Layer

The sensor layer is simulated by a program called Environment Simulator. According to defined parameters, Environment Simulator simulates the WMSN environment and produces processed sensor data model based on graphs [34]. This graph data is stored in sensor DB of data layer. As shown in Figure 4.2, using the Environment Simulator, the number of clusters in the area and the distance between sensors in a cluster are given as parameters before simulation. It deploys sensors in a rectangle are. The latitude and longitude values of the upper left and lower right corners of the rectangle should be given to deploy the WMSN to a place in a specific world coordinate. After parameter definitions, the simulator creates a WMSN and inserts related data to the sensor DB. It can produce sensor data according to the given scenarios. When a simulation is started according to the given scenario, the instantaneous positions of the simulated entities with their timestamps are inserted into the sensor database as if there is a real WMSN.

At this point, the start timestamp of the simulation, the velocity information of the moving entities in the simulation are entered into the system. The speed factor information of the simulation is entered in seconds in the system. The simulation determines the frequency of data generation using the speed factor parameter. Using Environment Simulator, entities such as Person, Animal, Vehicle, Group of People and Group of People with Vehicle can be created. In order to create entities in the rectangular area where the sensors are placed, the simulation drawer of Environment Simulator can be opened to set the waypoint of the entities to be created at the time of the simulation. For example, in Figure 4.3, twenty-five (25) simulations of different entities are prepared, including a person, a vehicle, an animal, a group of people or a group of people with a vehicle.

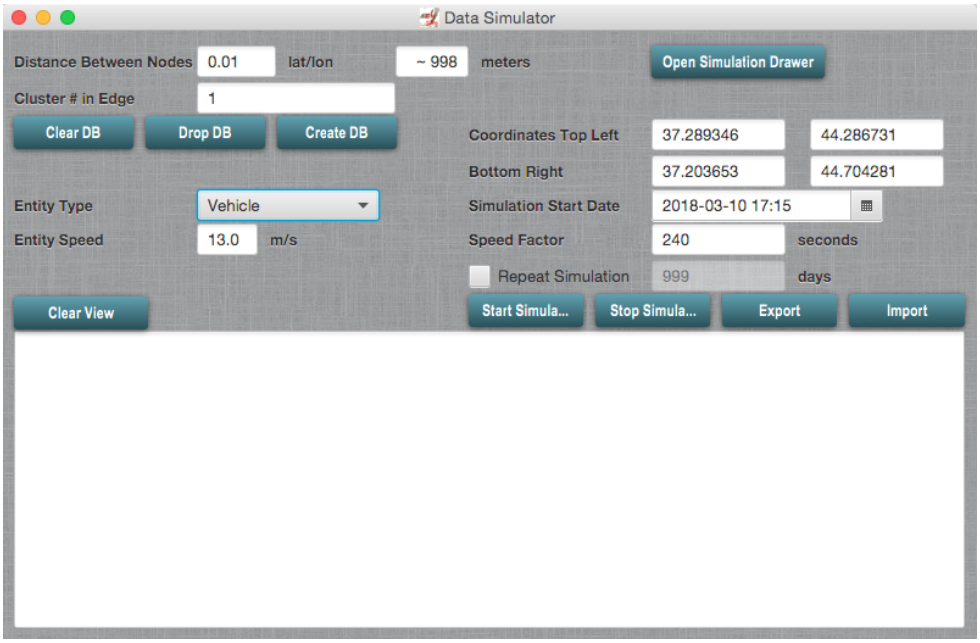


Figure 4.2: Environment Simulator

Once the entities to be simulated have been drawn in the drawer, the simulation is started from Environment Simulator. With the timestamp, the instantaneous positions of the simulated entities are inserted into the sensor database as if they were the entity positions detected by the sensor nodes close to that entity. Because sensor DB is controlled by data layer server, the server must be enabled before Environment Simulator can be run.

4.2 Data Layer

The data layer consists of a sensor database running on a server. The sink's WMSN data generated by the Environment Simulator is stored in sensor DB via data layer server. Since the data produced by the simulator is in graph model [34], we have a big graph database system at the data layer. In the surveillance application, OrientDB which is a graph database is used as the sensor DB. The database model of OrientDB consists of vertices and edges. The position information (latitude, longitude) of each sensor is the first data inserted into the sensor DB graph store. After launching a simulation from Environment Simulator, with timestamp, type and position of the entities with the identification of the detection sensors are inserted into the sensor

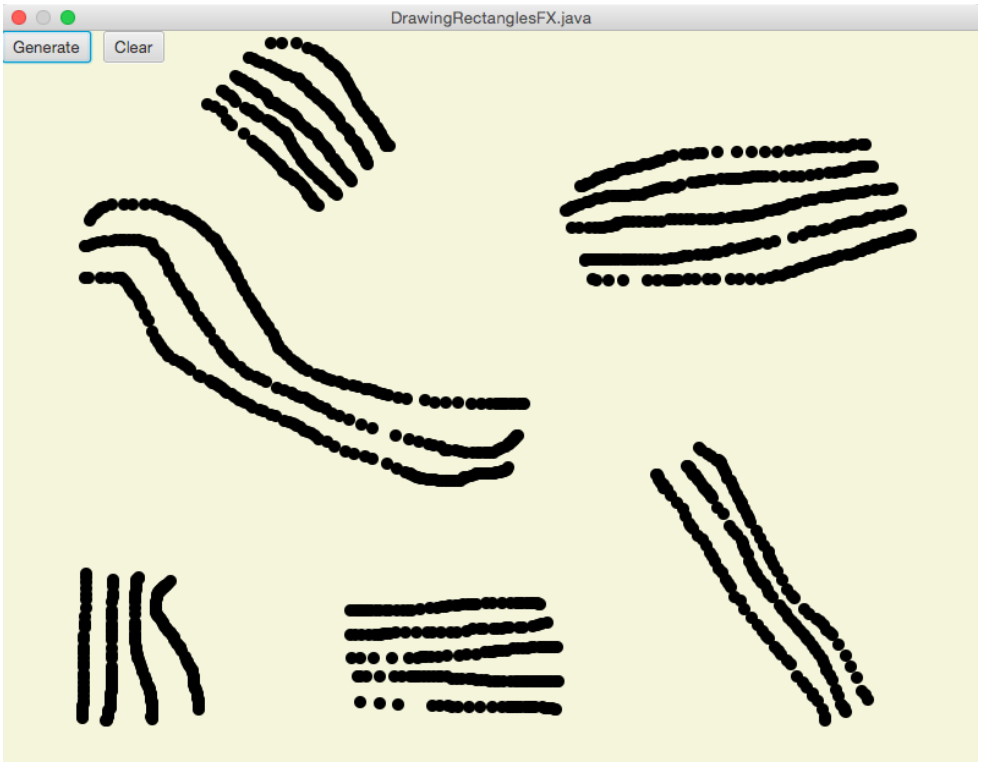


Figure 4.3: Simulation drawer of Environment Simulator

DB.

OrientDB has a push notification service. Semantic Web servlet application which is in the semantic Web layer registers to push notification service of sensor DB. A notification is sent to the semantic Web servlet application server to initiate data transfer when an update is made in the sensor database. In addition, semantic Web servlet application can register to sensor DB via the data layer server to instantly obtain data on adding, updating, or deleting a specific node by limiting the notification mechanism.

The structure of the sensor DB after insertions made by Environment Simulator over HTTP connection is shown in Figure 4.4. Only $actualData$, $sensorRawData$, $sensorNode$ and $fusedData$ vertexes of the sensor database are used by the semantic Web servlet application. $actualData$ vertex contains the entity's type, unique ID, position, and timestamp information. $sensorRawData$ includes measurement results from PIR, acoustic and vibratory sensors. PIR is a Boolean parameter and if set to true; the camera must be turned on by the simulated sensor system. The results of the acoustic and vibratory measurements are stored as float. Semantic Web servlet

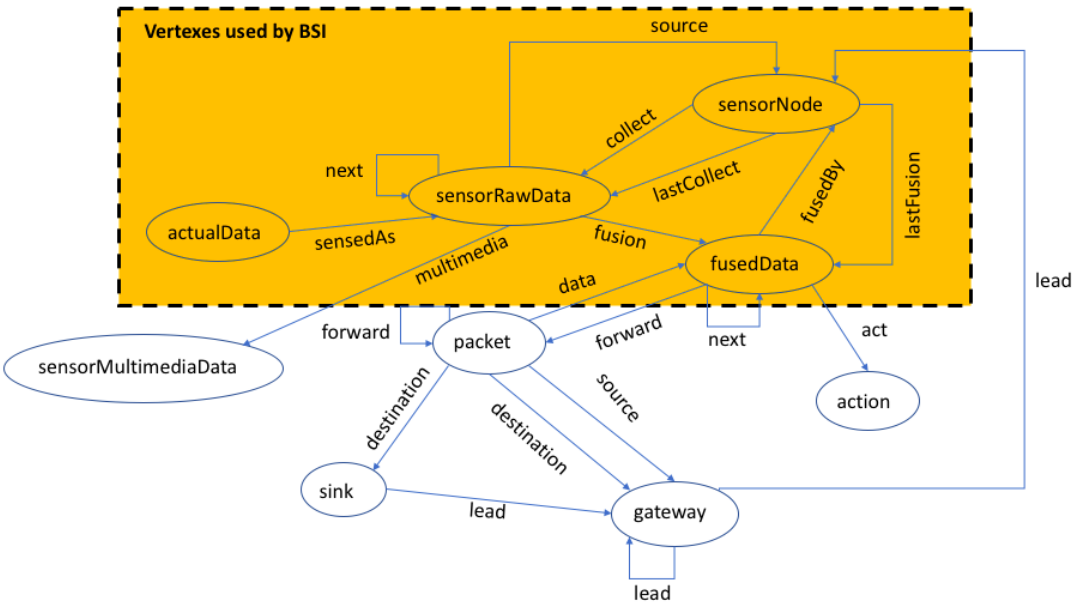


Figure 4.4: Sensor DB structure

application does not directly use the $sensorRawData$ vertex. The $sensorNode$ vertex contains the position information of all the sensors in WMSN. The $fusedData$ node is not used directly by semantic Web servlet application, like $sensorRawData$. The $sensorRawData$ and $fusedData$ vertices are used to access the $sensorNode$ vertices from $actualData$ vertices.

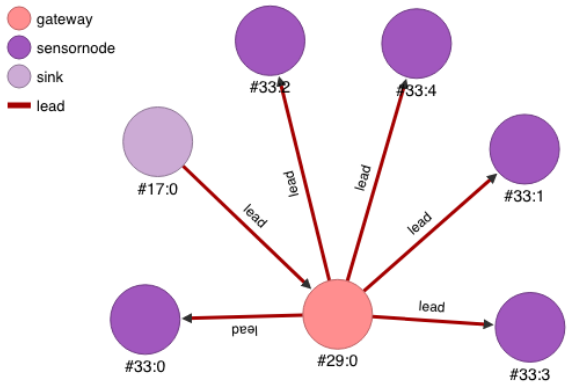


Figure 4.5: Sensor DB - After initialization step

For example, when WMSN is created for the first time with Environment Simulator, the structure in Figure 4.5 appears. In this structure, we first observe that the vertices created are $gateway$, $sensorNode$ and $sink$, their relation is represented by the $lead$

edge. According to this structure, all the sensor nodes are connected to a gateway and the gateway is connected to the sink.

Once initialization is complete, as soon as an entity is detected by the sensor nodes and the sink, Environment Simulator adds the $actualData$, $packet$, $sensorRawData$, $action$, $fusedData$, $sensorMultimediaData$ vertices for the detected entity. The resulting view of the sensor DB is shown in Figure 4.6. The semantic Web layer shares calculated results with the client after taking those input data stored in the vertices.

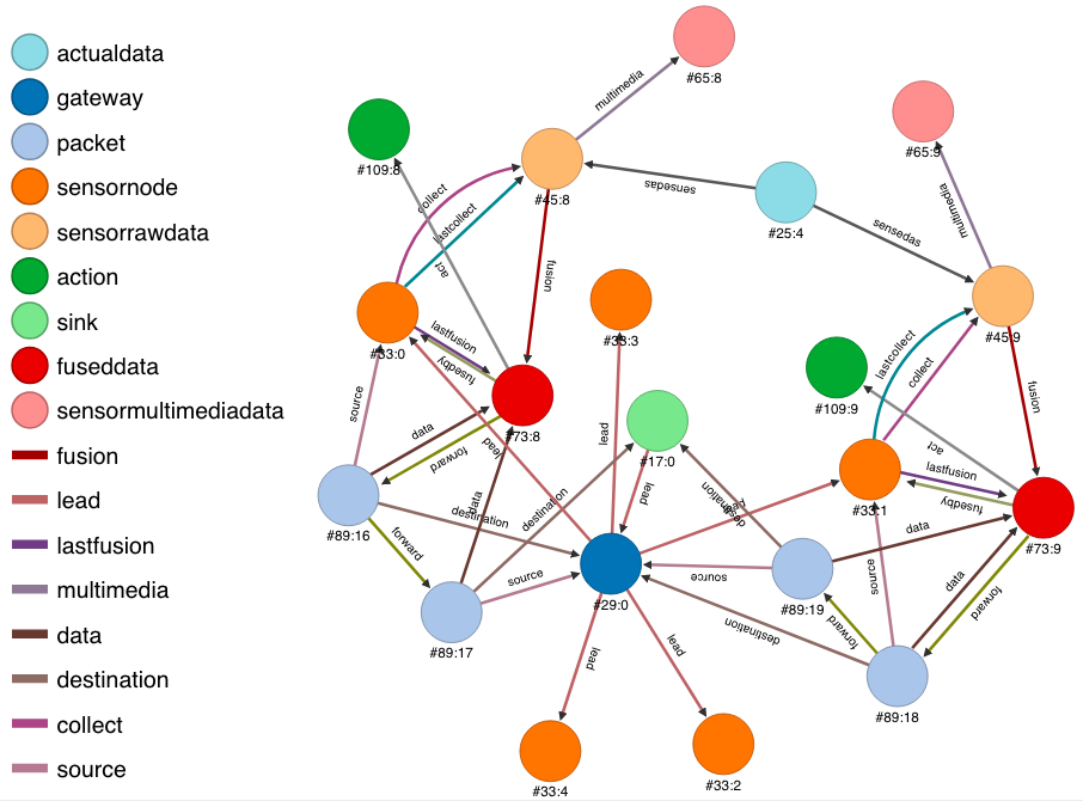
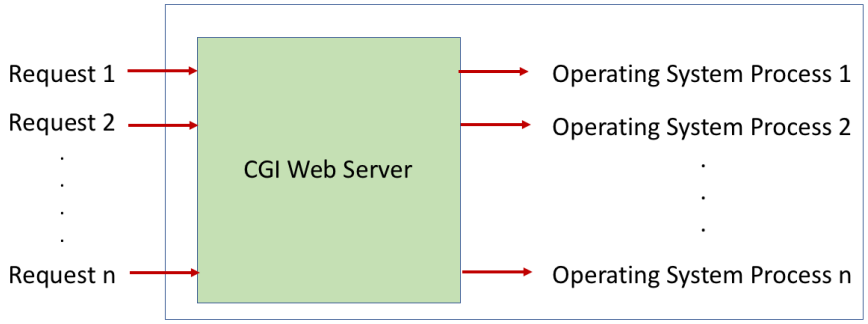


Figure 4.6: Sensor DB - After an entity is detected

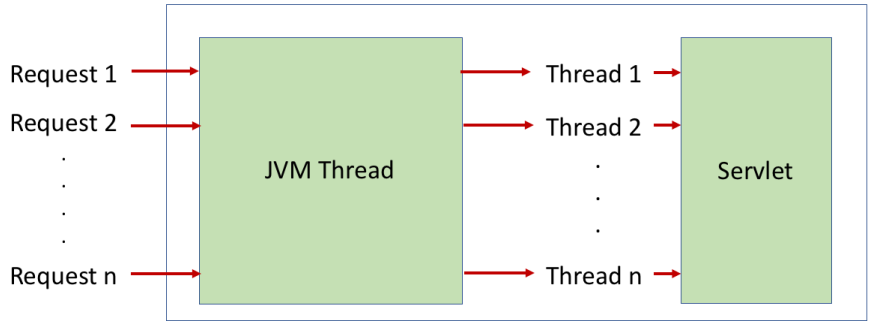
4.3 Semantic Web Layer

Semantic Web Layer consists of a semantic Web servlet application, a semantic WMSN KB, and a fuzzy reasoning engine by sharing the same ontology. Semantic Web servlet application running on semantic Web servlet application server is the core of the semantic Web layer. The semantic Web servlet application is a Web-

based HTTP servlet application that is implemented using Java Enterprise Edition. Servlets are a platform-independent, component-based way of creating Web applications, compared to the drawbacks of performance-related CGI programs [35]. Servlet is actually a Java class that increases the capabilities of a server by presenting applications as part of the request-response programming model. CGI provides dynamic content to the user by running a program on the server and accessing the database through this program. Figure 4.7 shows the CGI and servlet architectures. Since the servlet architecture has significant advantages over CGI, it is preferred in this study. The first is that servlet is portable while CGI is not. For each incoming request, while the servlet processes it by creating a separate Java thread, CGI creates a new operating system process. In addition, the servlet is cheaper than CGI. For example, HTML data is analyzed automatically in the servlet and a direct connection to the database can be established. For these reasons, it is best to create dynamic content in this layer and provide that content to clients.



(a) CGI.



(b) Servlet.

Figure 4.7: Possible server architecture of a Web application

On the front-end, the application is responsible for providing a graphical user interface to show users the environment in which the WMSN is established. Client-side of application has a responsive Web design and is compatible with mobile devices through the Bootstrap framework [36]. All animations of the application are implemented with support of HTML 5 SVG and Javascript. The semantic Web servlet application runs on Apache Tomcat 7.0 server.

Figure 4.8 illustrates all the data and command flows between components in our surveillance application. The synthetic and processed sensor data generated by Environment Simulator are inserted into the sensor DB via the data layer server endpoint. This data is extracted from sensor DB as offline or online and converted to semantic Web tuples (subject, predicate and object) according to the designed ontology shown in Figure 4.11.

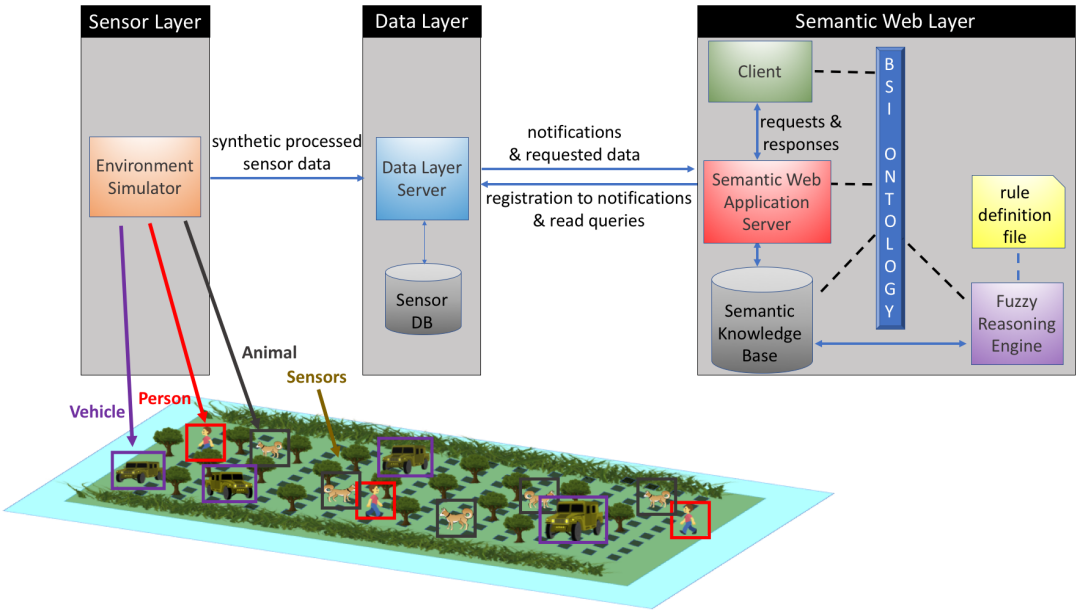
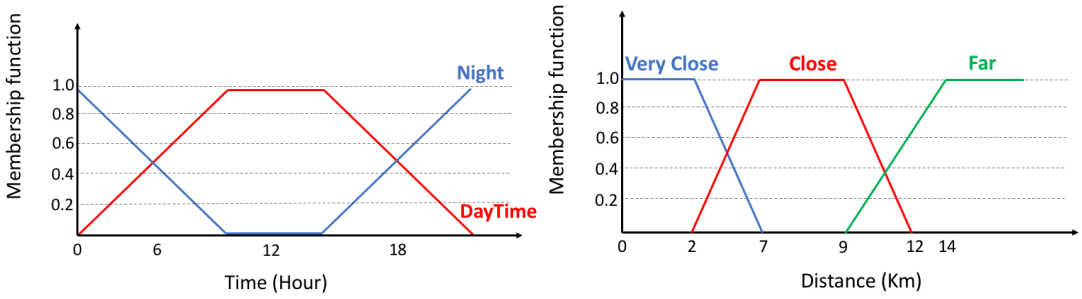


Figure 4.8: Data flow between components

As soon as the semantic web tuples are inserted into the semantic WMSN KB, the fuzzy reasoning engine is activated and initiates the fuzzification process and the subsequent inference process for each position data of each entity. To make reasoning on the Web with semantic techniques, Apache's Fuseki framework is used. Fuseki acts as a SPARQL server where select, insert, update, or delete operations can be performed on semantic web tuples over HTTP[37]. In order to perform the



(a) Time of day

(b) Closeness to target

Figure 4.9: Input membership functions

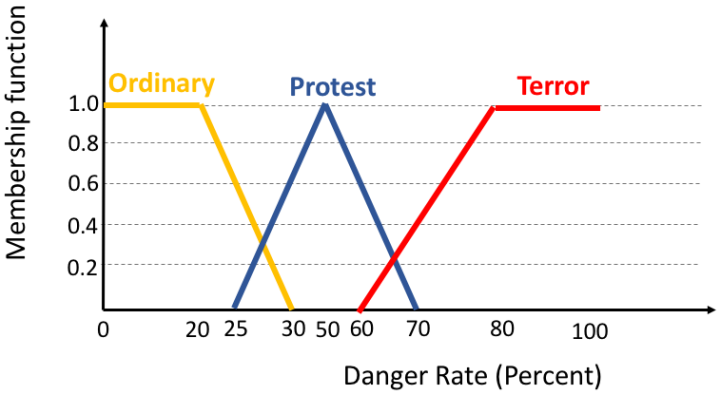


Figure 4.10: Output membership function

fuzzification, the implementations of the membership functions shown in Figure 4.9 must be assigned to the fuzzy reasoning engine via the input rule definition file provided during the initialization phase. The membership functions are applied to the $timeOfDay$ and $closeness$ input state variables. $timeOfDay$ can receive two linguistic label values with a certain degree of membership. $closeness$ can receive three linguistic labels with a certain membership degree. $timeOfDay$ receives $daytime$ and $night$ linguistic labels on the ontology and $closeness$ receives $veryclose$, $close$ and far linguistic labels. On the other hand, there are two crisp input state variables that do not have membership function. One of them is $movingDirection$ with two linguistic labels called $gettingCloser$ and $goingFar$. The other state variable is $detectedEntityType$ that have five linguistic labels: $person$, $animal$, $vehicle$, $grpOfPeople$, $grpOfPeopleAndVehicle$.

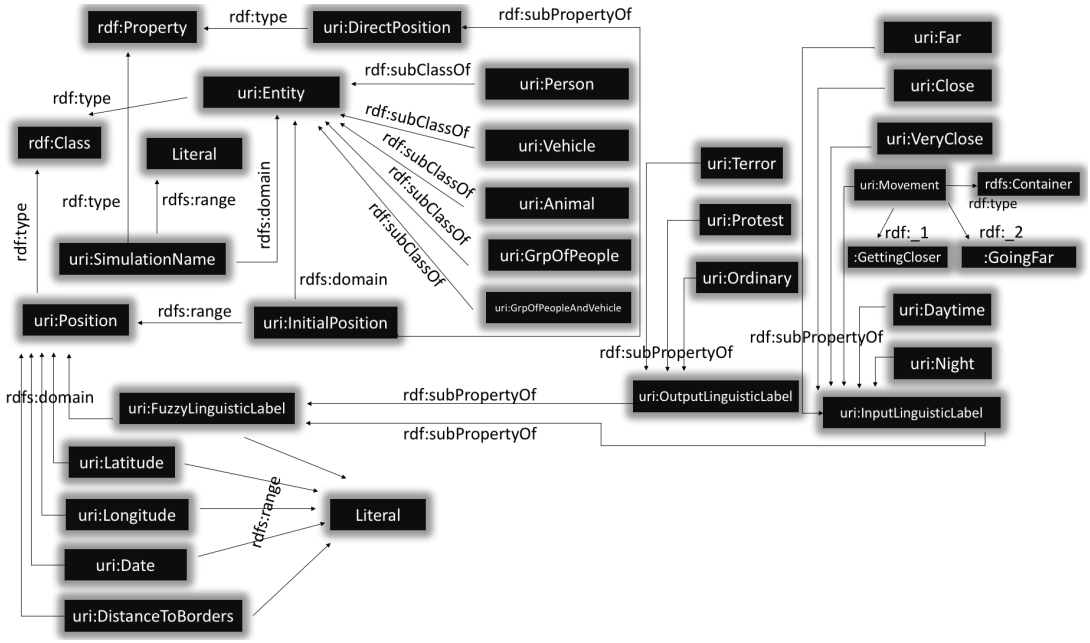


Figure 4.11: Terminological ontology of fuzzy semantic Web architecture

The membership degree values obtained as a result of fuzzification via membership functions in the reasoning engine are automatically inserted into Semantic WMSN KB in accordance with designed ontology. On the other hand, $entityType$ and $movingDirection$ crisp state variables with their linguistic labels which have membership degrees of 1.0, are directly inserted into semantic WMSN KB without applying a membership function. The fuzzy inferencing process, which started after fuzzification, is achieved by using the fuzzy rule base shown in Table 4.1 defined on reasoning engine, and in this way the linguistic labels of the $dangerRate$ output state variable and their respective degrees of membership are calculated. $dangerRate$ can take three optional linguistic label value called $ordinaryMobility$, $terrorAttack$ or $protest$. A total of $5x3x2x2 = 60$ different cases are evaluated for five $entityType$, three $closeness$, two $timeOfDay$ and two $movementDirection$ values defined in fuzzy reasoning engine's rule base. The calculation of the membership degree in the rules is performed by taking the minimum of membership degrees as defined in Equations 2.3 and 2.4 in Section 2.1.2.

Each rule in fuzzy rule base has a confidence coefficient received from a domain expert. The membership degrees of each output linguistic label obtained after fuzzy inferencing is multiplied by the confidence value of the related rule before defuzzi-

fication to increase the reliability of the system. This multiplication completes the work performed by fuzzy semantic reasoning engine.

On defuzzification stage, in other words, the stage of aggregation of triggered rules, in order to perform defuzzification with center of gravity formula, a flc file is created and desired defuzzification method with aggregation and accumulation properties is configured by using a fuzzy logic inference system language defined in study [38]. Then, this fuzzy logic controller file is given to a framework described in study [39] as input for generating our defuzzification method and this defuzzication method is adapted to semantic Web servlet application. Finally, output linguistic labels of the $dangerRate$ state variable and processed membership degrees with confidence coefficients are received from semantic WMSN KB by semantic Web servlet application. The application retrieves them and by using $dangerRate$ output membership function given in Figure 4.10, calculates a crisp $dangerRate$ data in percentage as formulated in Equation 2.7. In addition to percentage of danger rate, semantic Web servlet application then shares the position, entity type and timestamp information related to the source entity with client.

Table 4.1: Fuzzy rule base on fuzzy inferencing scheme

Rule	Entity Type	Closeness	Time of Day	Direction	Danger Type	Confidence
R_0	Person	Far	Night	Getting Closer	Protest	0.3
R_1	Person	Far	Night	Going Far	Ordinary	0.7
R_2	Person	Far	Daytime	Getting Closer	Protest	0.1
R_3	Person	Far	Daytime	Going Far	Ordinary	0.9
R_4	Person	Close	Night	Getting Closer	Protest	0.6
R_5	Person	Close	Night	Going Far	Protest	0.4
R_6	Person	Close	Daytime	Getting Closer	Ordinary	0.7
R_7	Person	Close	Daytime	Going Far	Protest	0.3
R_8	Person	Very Close	Night	Getting Closer	Terror	0.5
R_9	Person	Very Close	Night	Going Far	Protest	0.6
R_{10}	Person	Very Close	Daytime	Getting Closer	Terror	0.4
R_{11}	Person	Very Close	Daytime	Going Far	Protest	0.3
R_{12}	Animal	Far	Night	Getting Closer	Ordinary	1.0
R_{13}	Animal	Far	Night	Going Far	Ordinary	1.0
R_{14}	Animal	Far	Daytime	Getting Closer	Ordinary	1.0

Continued on next page

Table 4.1 – *Continued from previous page*

Rule	Entity Type	Closeness	Time of Day	Direction	Danger Type	Confidence
R_{15}	Animal	Far	Daytime	Going Far	Ordinary	1.0
R_{16}	Animal	Close	Night	Getting Closer	Ordinary	1.0
R_{17}	Animal	Close	Night	Going Far	Ordinary	1.0
R_{18}	Animal	Close	Daytime	Getting Closer	Ordinary	1.0
R_{19}	Animal	Close	Daytime	Going Far	Ordinary	1.0
R_{20}	Animal	Very Close	Night	Getting Closer	Ordinary	1.0
R_{21}	Animal	Very Close	Night	Going Far	Ordinary	1.0
R_{22}	Animal	Very Close	Daytime	Getting Closer	Ordinary	1.0
R_{23}	Animal	Very Close	Daytime	Going Far	Ordinary	1.0
R_{24}	Vehicle	Far	Night	Getting Closer	Ordinary	0.7
R_{25}	Vehicle	Far	Night	Going Far	Ordinary	0.8
R_{26}	Vehicle	Far	Daytime	Getting Closer	Terror	0.2
R_{27}	Vehicle	Far	Daytime	Going Far	Terror	0.1
R_{28}	Vehicle	Close	Night	Getting Closer	Ordinary	0.5
R_{29}	Vehicle	Close	Night	Going Far	Ordinary	0.6
R_{30}	Vehicle	Close	Daytime	Getting Closer	Terror	0.4
R_{31}	Vehicle	Close	Daytime	Going Far	Terror	0.3
R_{32}	Vehicle	Very Close	Night	Getting Closer	Terror	0.8
R_{33}	Vehicle	Very Close	Night	Going Far	Terror	0.7
R_{34}	Vehicle	Very Close	Daytime	Getting Closer	Terror	0.7
R_{35}	Vehicle	Very Close	Daytime	Going Far	Terror	0.6
R_{36}	GrpOfPeople	Far	Night	Getting Closer	Protest	0.3
R_{37}	GrpOfPeople	Far	Night	Going Far	Ordinary	0.8
R_{38}	GrpOfPeople	Far	Daytime	Getting Closer	Protest	0.2
R_{39}	GrpOfPeople	Far	Daytime	Going Far	Ordinary	0.6
R_{40}	GrpOfPeople	Close	Night	Getting Closer	Terror	0.3
R_{41}	GrpOfPeople	Close	Night	Going Far	Protest	0.4
R_{42}	GrpOfPeople	Close	Daytime	Getting Closer	Terror	0.2
R_{43}	GrpOfPeople	Close	Daytime	Going Far	Protest	0.6
R_{44}	GrpOfPeople	Very Close	Night	Getting Closer	Terror	0.9
R_{45}	GrpOfPeople	Very Close	Night	Going Far	Terror	0.7
R_{46}	GrpOfPeople	Very Close	Daytime	Getting Closer	Protest	0.8
R_{47}	GrpOfPeople	Very Close	Daytime	Going Far	Terror	0.5
R_{48}	GrpOfPeopleVehicle	Far	Night	Getting Closer	Protest	0.2
R_{49}	GrpOfPeopleVehicle	Far	Night	Going Far	Ordinary	0.7
R_{50}	GrpOfPeopleVehicle	Far	Daytime	Getting Closer	Ordinary	0.6

Continued on next page

Table 4.1 – *Continued from previous page*

Rule	Entity Type	Closeness	Time of Day	Direction	Danger Type	Confidence
R_{51}	GrpOfPeopleVehicle	Far	Daytime	Going Far	Terror	0.1
R_{52}	GrpOfPeopleVehicle	Close	Night	Getting Closer	Terror	0.5
R_{53}	GrpOfPeopleVehicle	Close	Night	Going Far	Ordinary	0.6
R_{54}	GrpOfPeopleVehicle	Close	Daytime	Getting Closer	Protest	0.4
R_{55}	GrpOfPeopleVehicle	Close	Daytime	Going Far	Terror	0.4
R_{56}	GrpOfPeopleVehicle	Very Close	Night	Getting Closer	Terror	0.9
R_{57}	GrpOfPeopleVehicle	Very Close	Night	Going Far	Protest	0.7
R_{58}	GrpOfPeopleVehicle	Very Close	Daytime	Getting Closer	Protest	0.9
R_{59}	GrpOfPeopleVehicle	Very Close	Daytime	Going Far	Terror	0.6

4.3.1 An Example Scenario

For example; suppose that the data to be read in the data layer via semantic Web servlet application and inserted into the semantic WMSN KB are as follows:

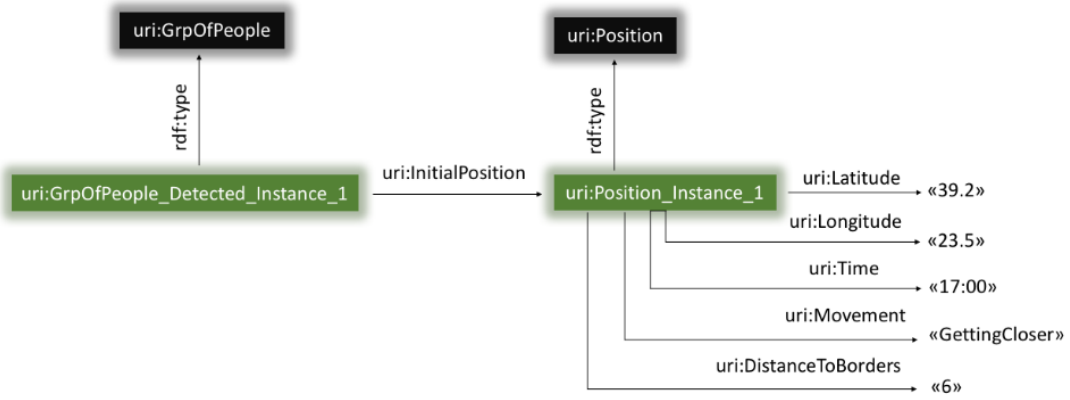


Figure 4.12: Assertional ontology of semantic WMSN KB

$Latitude = 39.2$, $Longitude = 23.5$, $Time = 17.00PM$, $DetectedEntity = GrpOfPeople$, $Movement = GettingCloser$, $Distance = 6km$. First of all, the assertional ontology of this data will be as shown in Figure 4.12. Then fuzzy reasoning engine is activated and the fuzzification process begins. Membership degrees are calculated through input membership functions shown in Figure 4.9, implemented by using the equations 3.1 and 3.2 in rule base.

Fuzzy input values are obtained as a result of fuzzification and final version of the assertional ontology on semantic WMSN KB is as in Figure 4.13.

$$\begin{aligned} Night(Time) &= ReverseTrapezoidal(17, 0, 11.5, 13.5, 24) \\ &= MIN(MAX(\tfrac{11.5-17}{11.5-0}, 0, \tfrac{17-13.5}{24-13.5}), 1) \\ &= 0.33 \end{aligned}$$

$$\begin{aligned} DayTime(Time) &= Trapezoidal(17, 0, 11.5, 13.5, 24) \\ &= MAX(MIN(\tfrac{17-0}{11.5-0}, 1, \tfrac{24-17.0}{24-13.5}), 0) \\ &= 0.66 \end{aligned}$$

$$\begin{aligned} VeryClose(Distance) &= Trapezoidal(6, 0, 0, 2, 7) \\ &= MAX(MIN(\tfrac{6-0}{0-0}, 1, \tfrac{7-6}{7-2}), 0) \\ &= 0.2 \end{aligned}$$

$$\begin{aligned} Close(Distance) &= Trapezoidal(6, 2, 7, 9, 12) \\ &= MAX(MIN(\tfrac{6-2}{7-2}, 1, \tfrac{12-6}{12-9}), 0) \\ &= 0.8 \end{aligned}$$

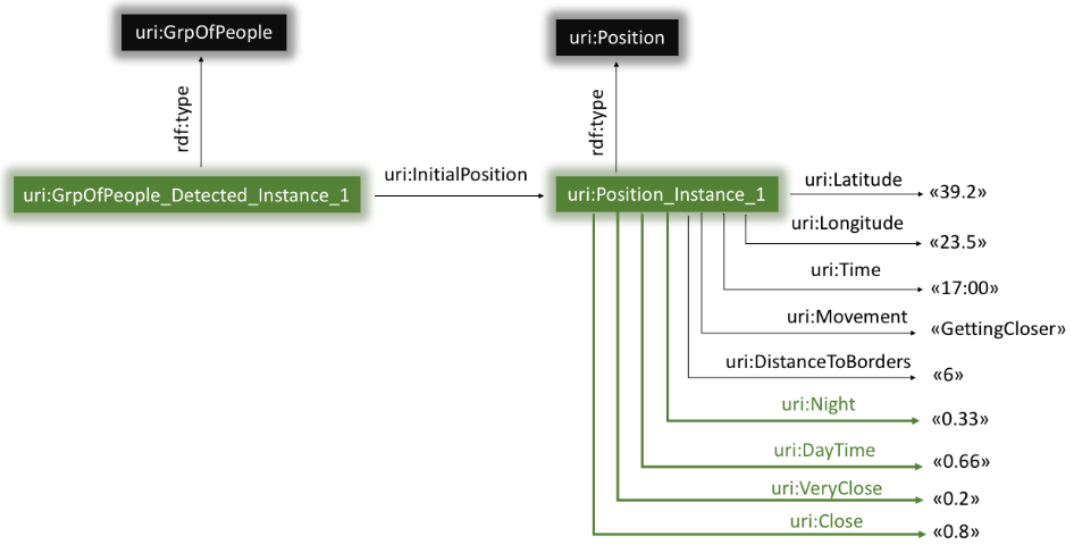


Figure 4.13: Assertional ontology of semantic WMSN KB - After fuzzification

After combining the resulting fuzzy input values($veryClose$, $close$, $daytime$, $night$) with crisp input values($entityType$, $movementDirection$) that do not need fuzzification, fuzzy inferencing is started on fuzzy reasoning engine. Accordingly, R_{40}, R_{42}, R_{44}, R_{46} rules, expressed in Table 4.1, are triggered. As a result of triggering of

the rules,output membership degrees are calculated by applying Equation 2.3 to input membership degrees of danger types and the values obtained after this calculation is multiplied by the confidence coefficients of related rules:

$$eval(R_{40}) : GrpOfPeople(EntityType)\ AND\ Close(Distance)$$
$$AND\ Night(TimeOfDay)\ AND\ GettingCloser(Direction)$$
$$\rightarrow Terror(DangerType)$$

$$Terror(DangerType) = MIN(GrpOfPeople(EntityType),\ Close(Distance),$$
$$= Night(TimeOfDay),\ GettingCloser(Direction))$$
$$= MIN(1,\ 0.8,\ 0.33,\ 1.0)\ *\ 0.3\ = 0.099$$

$$eval(R_{42}) : GrpOfPeople(EntityType)\ AND\ Close(Distance)$$
$$AND\ DayTime(TimeOfDay)\ AND\ GettingCloser(Direction)$$
$$\rightarrow Terror(DangerType)$$

$$Terror(DangerType) = MIN(GrpOfPeople(EntityType),$$
$$= Close(Distance),\ DayTime(TimeOfDay),$$
$$= GettingCloser(Direction))$$
$$= MIN(1,\ 0.8,\ 0.66,\ 1.0)\ *\ 0.2\ = 0.132$$

$$eval(R_{44}) : GrpOfPeople(EntityType)\ AND\ VeryClose(Distance)$$
$$AND\ Night(TimeOfDay)\ AND\ GettingCloser(Direction)$$
$$\rightarrow Terror(DangerType)$$

$$Terror(DangerType) = MIN(GrpOfPeople(EntityType),$$
$$= VeryClose(Distance),\ Night(TimeOfDay),$$
$$= GettingCloser(Direction))$$
$$= MIN(1,\ 0.2,\ 0.33,\ 1.0)\ *\ 0.9 = 0.18$$

$$eval(R_{46}) : GrpOfPeople(EntityType)\ AND\ VeryClose(Distance)$$
$$AND\ DayTime(TimeOfDay)\ AND\ GettingCloser(Direction)$$
$$\rightarrow Protest(DangerType)$$

$$
\begin{aligned}
Protest(DangerType) &= MIN(GrpOfPeople(EntityType), \\
&= VeryClose(Distance),\ DayTime(TimeOfDay), \\
&= GettingCloser(Direction)) \\
&= MIN(1,\ 0.2,\ 0.66,\ 1.0)\ *\ 0.8\ = 0.16
\end{aligned}
$$

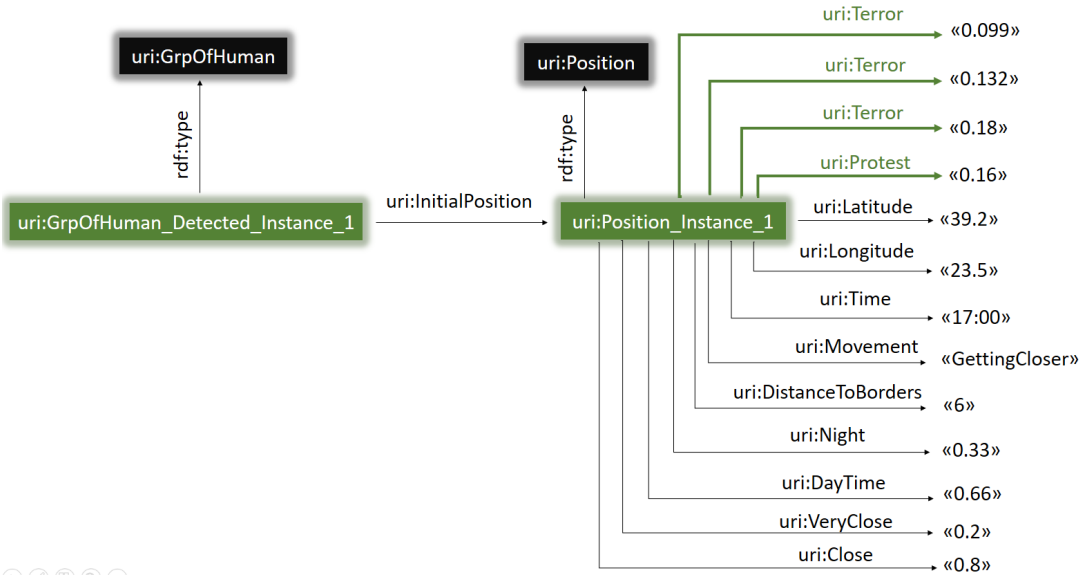


Figure 4.14: Assertional ontology of semantic WMSN KB - After fuzzy inferencing

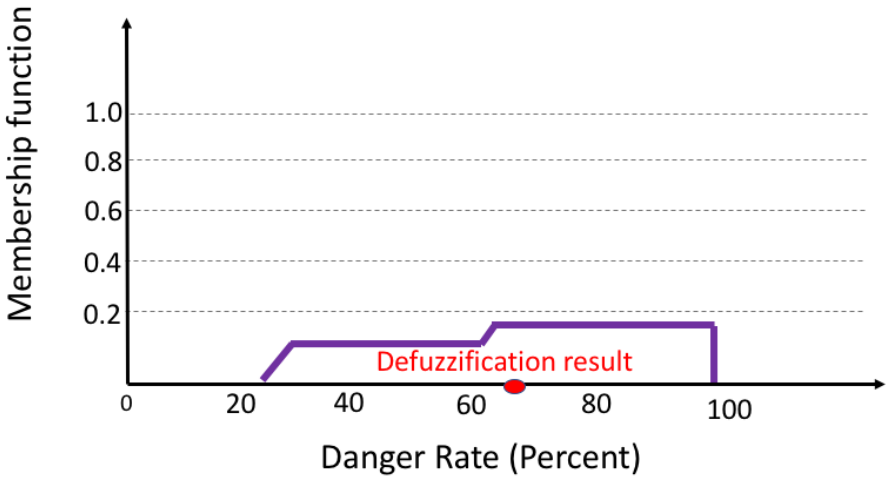


Figure 4.15: Defuzzification output

Prior to defuzzification performed by semantic Web servlet application, value of the output membership degrees are finalized. The final version of semantic WMSN Knowledge Base after fuzzy inferencing is as shown in Figure 4.14. After this stage,

the output membership degrees obtained from four triggered rules are defuzzified and a crisp danger rate is calculated. The defuzzification calculation is performed as specified in Equation 2.7:

$$COG = \frac{25*0+(30+35+40+45+50+55+60)*0.16 \ + \ (65+70+75+80+85+90+95+100)*0.18}{0+0.16+0.16+0.16+0.16+0.16+0.16+0.16+0.18+0.18+0.18+0.18+0.18+0.18+0.18+0.18} = 66.09$$

While the semantic Web servlet application shares this danger rate data with client, it also sends instant positions of sensors and entities to the client, allowing the data to be displayed to client with animation. Semantic Web servlet application has two main modes of operation: online and offline monitoring. For both modes of operation, reasoning processes in semantic Web layer are identical to those described in this section, but the way in which data is transmitted to semantic Web layer varies.

4.3.2 Offline Monitoring

Offline monitoring mode is used to watch previous WMSN activities generated by Environment Simulator. When offline mode is turned on, semantic Web servlet application connects to sensor DB graph store via data layer server to get processed sensor data of all days. First, $actualData$ list containing latitude, longitude, entity type, timestamp, and ID information is retrieved from sensor DB. Sensor node list in the WMSN is then taken from sensor DB together with the position information. The positions of sensor nodes are used to display this on the client screen. Secondly, the $sensorNode$ vertex is used indirectly via $actualData$ vertex. Finally, the $actualData$ and $sensorData$ vertexes fetched from sensor DB are converted to semantic Web tuples. To achieve this, an HTTP connection is established from semantic Web servlet application to semantic WMSN KB through Apache Jena Fuseki endpoint, and all semantic tuples are inserted into semantic WMSN KB. Then, these semantic tuples are first fuzzified, then fuzzy inference is applied by fuzzy reasoning engine. Semantic Web servlet application performs a defuzzification process on semantic Web tuples as soon as it detects of new fuzzy inference output tuples in the checks it performs periodically, and then calculates the danger rate for terrorist activity, protest or ordinary mobility and transmits the result to the client side. The reasoning server rules are defined on initialization phase, so these rules do not directly

affect operating mechanism of semantic Web servlet application.

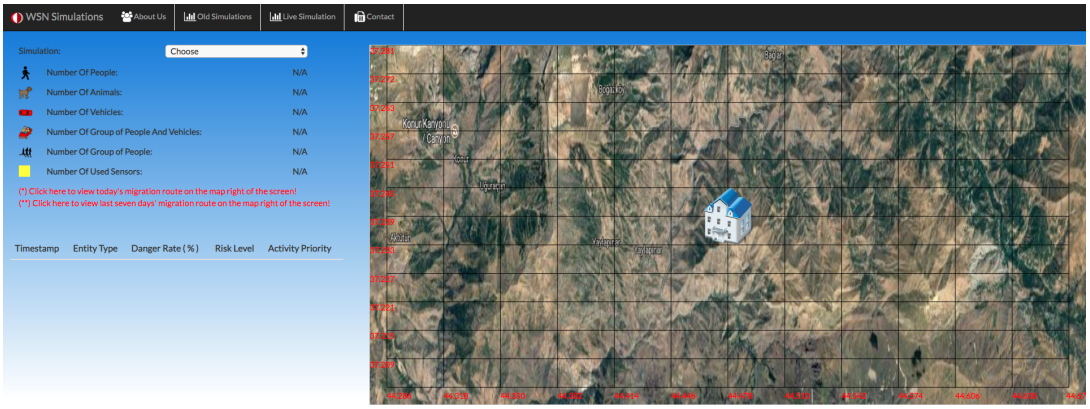


Figure 4.16: Offline mode in semantic Web servlet application (no simulation is selected)

The screen shown to client in offline monitoring mode is as in Figure 4.16. As it appears in Figure 4.16, the user must first select a simulation to display in the application. The simulation name is the same as the activity date entered in Environment Simulator. When the activity date is selected, the number of each entity detected by sensor nodes and total number of sensors detecting the entity are displayed to the user for the selected date. In addition to entity count, sensors that detect the changed entity positions are shown to client as blinking with the SVG animation feature of HTML 5. Furthermore, if a danger rate detection is performed by semantic Web servlet application for the selected date, this information is shown to the user as in Figure 4.17.

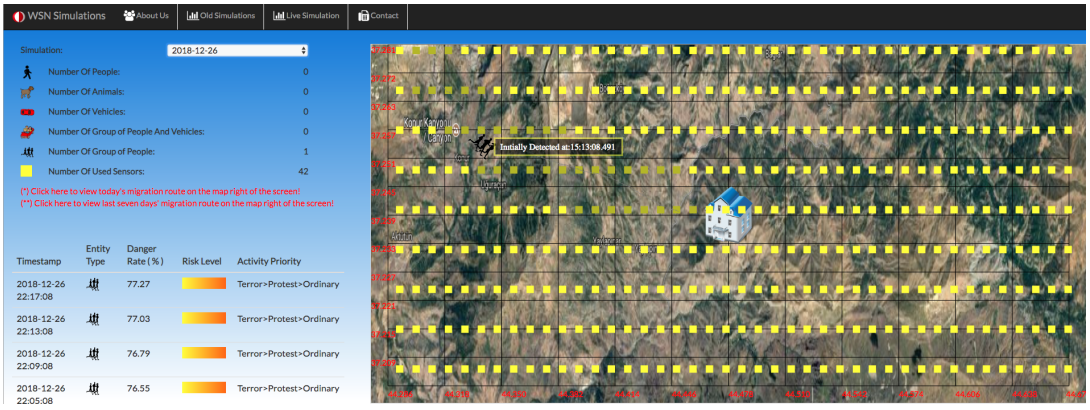


Figure 4.17: Offline mode in semantic Web servlet application (simulation replay)

In addition to its animation and reasoning capabilities, semantic Web servlet applica-

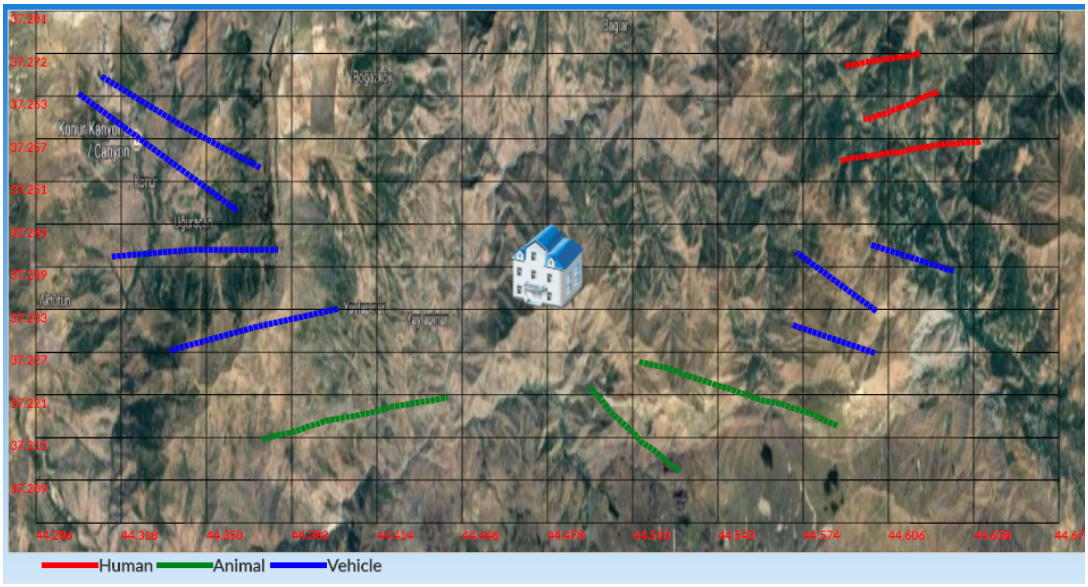


Figure 4.18: Daily migration route

tion in offline mode shows the daily migration route and the migration route of the last 7 days to the user. In the WMSN environment, seven vehicles, three animals and three people were detected throughout the day, as shown in the Figure 4.18.

4.3.3 Online Monitoring

Online monitoring mode provides instant tracking of environment in which the synthetic WMSN is established. As soon as this mode is turned on, semantic Web servlet application first connects to sensor DB via data layer server and subscribes to the $actualData$ and $sensorNode$ vertexes. The subscription enables semantic Web servlet application to be notified when new processed sensor data as $actualData$ vertex is inserted into sensor DB graph store via Environment Simulator or $sensorNode$ vertex is updated. Insertion of $actualData$ means that sensor nodes sense an entity with a new position, type, ID and time stamp; while updates of $sensorNode$ provide information about which sensor nodes sense that entity.

When semantic Web servlet application is notified by data layer server as soon as an $actualData$ insertion or $sensorNode$ update is available; LiveSim module of semantic Web servlet application takes whatever is updated or inserted and waits for three

seconds in a separate thread to receive a new notification. This trick is important for integrity of the simulation. Because Environment Simulator simulates each entity in turn. Therefore, in order to show that the entities are moving in parallel, the server side of semantic Web servlet application does not send new animation data as soon as it receives the first update notification. It waits for three-second periods throughout the entire simulation and sends all the animation data to the client side at the end of the first three-second waiting period without a notification from data layer server. In this way, it is ensured that the entity data, which are added or updated in the order, are displayed on the client side as if the entities are moving in parallel in the same time interval.

The processed data obtained just before the client screen is updated with that new data is converted into semantic tuples and inserted into semantic WMSN KB from the relevant endpoint. The semantic tuples are then processed by fuzzy reasoning engine, as in offline mode. If a new danger rate detection is performed, this information is shared with the client.

The screen shown to the user in online monitoring mode is almost the same as in offline monitoring mode. One difference between them is that the migration route option is not in online tracking mode. Because it is designed as an offline tracking capability. However, displaying the number of entities detected and the number of sensor nodes that detect entities are the same as the offline monitoring mode. In addition, animations shown to the client were again provided with HTML 5 SVG feature and fuzzification and fuzzy inferencing mechanism in danger rate detection process were also provided by fuzzy reasoning engine. Another difference between online and offline monitoring mode is the type of communication. Messages sent from client to server are provided with HTML server sent events. This mechanism automatically asks the server if there is a need to update anything and if an update is required, the client only updates related part of the page by taking this update from the server.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

The main idea put forward in this study is that activity detection can be done with the semantic Web. The study presented in this thesis differs from other studies in this area for two reasons. There are two types of studies in the literature. The first type of study makes sensor raw data available for other software, end-users or other sensor networks via the semantic Web. These studies do this by defining a new sensor ontology or by updating existing ontologies. However, instead of converting raw sensor data, our study focuses on publishing processed sensor data acquired from the sink of WMSN, in accordance with the ontology. Moreover, fuzzy logic is not used in those studies. The second type of study processes wireless sensor networks data to detect activity using fuzzy set theory as in our study, but the semantic Web infrastructure making processed sensor available in a well-defined format for any other systems to consume this data does not exist in this type of studies. On the other hand, the central point of this thesis is not only semantic Web technology, but also the theory of fuzzy sets when detecting activity with processed multimedia sensor networks data.

The architecture described in Chapter 3 is implemented for an example surveillance application. This application, called BSI, is a security system that instantly calculates and shares the current danger ratio with soldiers at the border station to alert them to activities such as a terrorist attack or protest in the controlled area. In order to achieve this, a data simulator[34] used in our work group in the university is used as Environment Simulator to simulate the wireless multimedia sensor network environment

and produce processed sensor data instead of sensor raw data as if this data comes from the sink of WMSN. The study implements danger ratio detection calculations using the rules defined on fuzzy reasoning engine. These rules are used in performing fuzzification and fuzzy inferencing processes of fuzzy set theory. There are also uncertainty values defined for rules, for example when a night-time activity can be more dangerous than daytime. In this way, our approach gives semantic web reasoning mechanism the ability to think humanely.

5.2 Future Work

The study presented in this thesis focuses on the usage of semantic Web reasoning to detect activities with processed WMSN data. Moreover, fuzzy set theory is used in conjunction with semantic Web reasoning. In addition, thanks to the usage of semantic Web with a designed ontology, processed sensor data acquired from the sink and fuzzy reasoning results are shared with other third party applications in a well-defined, machine-processable format. On the other hand, standardization of different types of sensor raw data and thus communication of different WMSNs are not focused on this study. In the future; if semantic part of our approach is improved in order to standardize not only processed sensor data but also sensor raw data; this study can be transformed into a high-potential product that can be preferred among environmental monitoring applications in sectors such as health and defense because of being transformed into a more extensible product.

REFERENCES

[1] A. Zafeiropoulos, D.-E. Spanos, S. Arkoulis, N. Konstantinou, and N. Mitrou, "Data management in sensor networks using semantic web technologies," 2009.

[2] A. Moraru and D. Mladenić, "A framework for semantic enrichment of sensor data," *Journal of computing and information technology*, vol. 20, no. 3, pp. 167–173, 2012.

[3] G. R. Berkenbrock, C. M. Hirata, F. G. Á. de Oliveira Júnior, and J. M. P. de Oliveira, "Applying semantic web services and wireless sensor networks for system integration," in *International Conference on IT Revolutions*, pp. 161–170, Springer, 2008.

[4] F. L. Lewis, "Wireless sensor networks," *Smart environments: technologies, protocols, and applications*, pp. 11–46, 2004.

[5] S.-H. Choi, B.-K. Kim, J. Park, C.-H. Kang, and D.-S. Eom, "An implementation of wireless sensor network," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 236–244, 2004.

[6] A. Milenković, C. Otto, and E. Jovanov, "Wireless sensor networks for personal health monitoring: Issues and an implementation," *Computer communications*, vol. 29, no. 13-14, pp. 2521–2533, 2006.

[7] S. Nagaraj and R. V. Biradar, "Applications of wireless sensor networks in the real-time ambient air pollution monitoring and air quality in metropolitan cities—a survey," in *Smart Technologies For Smart Nation (SmartTechCon), 2017 International Conference On*, pp. 1393–1398, IEEE, 2017.

[8] S. Sridharan, "Water quality monitoring system using wireless sensor network," *International Journal of Electronic Communications Engineering Advanced Research*, vol. 3, pp. 399–402, 2014.

[9] X. Sun and E. J. Coyle, "Low-complexity algorithms for event detection in wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 7, 2010.

[10] A. Sundaresan, P. K. Varshney, and N. S. Rao, "Distributed detection of a nuclear radioactive source using fusion of correlated decisions," in *Information Fusion, 2007 10th International Conference on*, pp. 1–7, IEEE, 2007.

[11] H. Liu and A. Gegov, "Rule based systems and networks: Deterministic and fuzzy approaches," in *Intelligent Systems (IS), 2016 IEEE 8th International Conference on*, pp. 316–321, IEEE, 2016.

[12] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific american*, vol. 284, no. 5, pp. 34–43, 2001.

[13] G. J. Klir and B. Yuan, *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A. Zadeh*. World Scientific Publishing Co., Inc., 1996.

[14] N. Werro, *Fuzzy classification of online customers*. Springer, 2015.

[15] E. Brands, "Integration of wsn in semantic web," Master's thesis, Universitat Politècnica de Catalunya, 2014.

[16] K. Kapitanova, S. H. Son, and K.-D. Kang, "Using fuzzy logic for robust event detection in wireless sensor networks," *Ad Hoc Networks*, vol. 10, no. 4, pp. 709–722, 2012.

[17] N. Shadbolt, T. Berners-Lee, and W. Hall, "The semantic web revisited," *IEEE intelligent systems*, vol. 21, no. 3, pp. 96–101, 2006.

[18] P. Hitzler, M. Krotzsch, and S. Rudolph, *Foundations of semantic web technologies*. Chapman and Hall/CRC, 2009.

[19] O. Lassila and R. R. Swick, "Resource description framework (rdf) model and syntax specification," 1999.

[20] D. Brickley and L. Miller, "Foaf vocabulary specification 0.91," 2010.

[21] L. Muduli, P. K. Jana, and D. P. Mishra, "Wireless sensor network based fire monitoring in underground coal mines: A fuzzy logic approach," *Process Safety and Environmental Protection*, vol. 113, pp. 435–447, 2018.

[22] G. Klir and B. Yuan, *Fuzzy sets and fuzzy logic*, vol. 4. Prentice hall New Jersey, 1995.

[23] M. M. Gupta and J. Qi, "Theory of t-norms and fuzzy inference methods," *Fuzzy sets and systems*, vol. 40, no. 3, pp. 431–450, 1991.

[24] T. J. Ross, *Fuzzy logic with engineering applications*. John Wiley & Sons, 2005.

[25] U. Arora and A. Kumar, "Architecture for raw data processing at sensor nodes in semantic wireless sensor networks," *International Journal of Computer Applications*, vol. 55, no. 12, 2012.

[26] Y. J. Lee, J. Trevathan, I. Atkinson, and W. Read, "The integration, analysis and visualization of sensor data from dispersed wireless sensor network systems using the swe framework," *Journal of Telecommunications and Information Technology*, 2015.

[27] K. A. Bispo, N. S. Rosa, and P. R. Cunha, "A semantic solution for saving energy in wireless sensor networks," in *Computers and Communications (ISCC), 2012 IEEE Symposium on*, pp. 000492–000499, IEEE, 2012.

[28] L. Gao, M. Bruenig, and J. Hunter, "Estimating fire weather indices via semantic reasoning over wireless sensor network data streams," *arXiv preprint arXiv:1411.2186*, 2014.

[29] H. S. Hossain, M. A. A. H. Khan, and N. Roy, "Active learning enabled activity recognition," *Pervasive and Mobile Computing*, vol. 38, pp. 312–330, 2017.

[30] N. Cho and E.-K. Kim, "Enhanced voice activity detection using acoustic event detection and classification," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 1, 2011.

[31] G. Civitarese, C. Bettini, T. Sztyler, D. Riboni, and H. Stuckenschmidt, "Nectar: Knowledge-based collaborative active learning for activity recognition," 2018.

[32] V. Huang and M. K. Javed, "Semantic sensor information description and processing," in *Sensor Technologies and Applications, 2008. SENSORCOMM'08. Second International Conference on*, pp. 456–461, IEEE, 2008.

[33] "Rdf concepts." https://www.w3.org/TR/rdf-concepts/. last accessed at 07/10/2018.

[34] C. Küçükkeçeci and A. Yazıcı, "Big data model simulation on a graph database for surveillance in wireless multimedia sensor networks," *Big data research*, vol. 11, pp. 33–43, 2018.

[35] "Servlets tutorial." https://www.tutorialspoint.com/servlets/index.htm. last accessed at 07/10/2018.

[36] J. Spurlock, *Bootstrap: Responsive Web Development*. " O'Reilly Media, Inc.", 2013.

[37] "Apache jena." https://www.jena.apache.org/documentation/serving_data/. last accessed at 07/10/2018.

[38] P. Cingolani and J. Alcala-Fdez, "jfuzzylogic: a robust and flexible fuzzy-logic inference system language implementation," in *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, pp. 1–8, IEEE, 2012.

[39] P. Cingolani and J. Alcalá-Fdez, "jfuzzylogic: a java library to design fuzzy logic controllers according to the standard for fuzzy control programming," *International Journal of Computational Intelligence Systems*, vol. 6, no. sup1, pp. 61–75, 2013.

APPENDIX A

APPENDIX 1

A.1 Software for developing and using Border Safety Informer

- Eclipse for Java Enterprise Edition 4.4.2
- Apache Tomcat Server 7.0
- OrientDB Graph Store 2.2.17
- RabbitMQ 3.6.12
- Apache Jena Fuseki Server 2.4.0
- Synthetic Data Simulator
- Apache Jena RDF API
- JFuzzyLogic Framework

A.2 BSI Fuzzy Reasoning Engine Input Membership Function Rules

```
@prefix rdf:  http://www.w3.org/1999/02/22rdf syntax ns   .
@prefix rdfs:  http://www.w3.org/2000/01/rdf schema   .
@prefix metu:  http://www.example.org/   .
```

```
[ decideTimeOfDayDayTime:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?pos1 metu:Date ?second1 ) ,

difference(?second1, 0.0, ?diff1),
difference(41400.0, 0.0, ?diff2),
quotient( ?diff1, ?diff2, ?param1 ),
min(?param1, 1.0, ?tmp1),

difference(86400.0, ?second1, ?diff3),
difference(86400.0, 48600.0, ?diff4),
quotient( ?diff3, ?diff4, ?param2 ),
min(?tmp1, ?param2, ?tmp2),

max( ?tmp2, 0.0, ?membershipDegree ),
notEqual( ?membershipDegree, 0.0 )

→ ( ?pos1 metu:DayTime ?membershipDegree )
]
```

```
[ decideTimeOfDayNight:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?pos1 metu:Date ?second1 ) ,

difference(41400.0, ?second1, ?diff1),
difference(41400.0, 0.0, ?diff2),
quotient( ?diff1, ?diff2, ?param1 ),
max(?param1, 0.0, ?tmp1),

difference(?second1, 48600.0, ?diff3),
difference(86400.0, 48600.0, ?diff4),
quotient( ?diff3, ?diff4, ?param2 ),
max(?tmp1, ?param2, ?tmp2),

min( ?tmp2, 1.0, ?membershipDegree ),
notEqual( ?membershipDegree, 0.0 )

→ ( ?pos1 metu:Night ?membershipDegree )
]
```

```
[ decideMovementGettingCloser:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?conceptId1 metu:DirectPosition ?pos2 ) ,
( ?pos1 metu:Distance ?dist1 ) ,
( ?pos2 metu:Distance ?dist2 ) ,
( ?pos1 metu:Date ?second1 ) ,
( ?pos2 metu:Date ?second2 ) ,
notEqual( ?pos1, ?pos2 ),

sum(?second1, 240, ?second3) ,
equal( ?second2, ?second3 ),

difference(?dist2, ?dist1, ?distanceDifference),
le( ?distanceDifference, 0 ),

→ ( ?pos2 metu:Movement metu:GettingCloser)
]
```

```
[ decideMovementGoingFar:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?conceptId1 metu:DirectPosition ?pos2 ) ,
( ?pos1 metu:Distance ?dist1 ) ,
( ?pos2 metu:Distance ?dist2 ) ,
( ?pos1 metu:Date ?second1 ) ,
( ?pos2 metu:Date ?second2 ) ,
notEqual( ?pos1, ?pos2 ),

sum(?second1, 240, ?second3) ,
equal( ?second2, ?second3 ),

difference(?dist2, ?dist1, ?distanceDifference),
greaterThan( ?distanceDifference, 0 ),

→ ( ?pos2 metu:Movement metu:GoingFar )
]
```

```
[ decideClosenessVeryClose:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?pos1 metu:Distance ?dist1 ) ,

difference(7.0, ?dist1, ?diff3),
difference(7.0, 2.0, ?diff4),
quotient( ?diff3, ?diff4, ?param2 ),
min(1.0, ?param2, ?tmp2),

max( ?tmp2, 0.0, ?membershipDegree ),
notEqual( ?membershipDegree, 0.0 )

→ ( ?pos1 metu:VeryClose ?membershipDegree )
]
```

```
[ decideClosenessClose:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?pos1 metu:Distance ?dist1 ) ,

difference(?dist1, 2.0, ?diff1),
difference(7.0, 2.0, ?diff2),
quotient( ?diff1, ?diff2, ?param1 ),
min(?param1, 1.0, ?tmp1),

difference(12.0, ?dist1, ?diff3),
difference(12.0, 9.0, ?diff4),
quotient( ?diff3, ?diff4, ?param2 ),
min(?tmp1, ?param2, ?tmp2),

max( ?tmp2, 0.0, ?membershipDegree ),
notEqual( ?membershipDegree, 0.0 )

→ ( ?pos1 metu:Close ?membershipDegree )
]
```

```
[ decideClosenessFar:
( ?conceptId1 metu:DirectPosition ?pos1 ) ,
( ?pos1 metu:Distance ?dist1 ) ,

difference(?dist1, 9.0, ?diff1),
difference(14.0, 9.0, ?diff2),
quotient( ?diff1, ?diff2, ?param1 ),
min(?param1, 1.0, ?tmp1),

max( ?tmp1, 0.0, ?membershipDegree ),
notEqual( ?membershipDegree, 0.0 )

→ ( ?pos1 metu:Far ?membershipDegree )
]
```