UNIVERSITETI I EVROPËS JUGLINDORE
УНИВЕРЗИТЕТ НА ЈУГОИСТОЧНА ЕВРОПА
SOUTH EAST EUROPEAN UNIVERSITY

FAKULTETI I SHKENCAVE DHE TEKNOLOGJIVE BASHKËKOHORE
ФАКУЛТЕТ ЗА СОВРЕМЕНИ НАУКИ И ТЕХНОЛОГИИ
FACULTY OF CONTEMPORARY SCIENCES AND TECHNOLOGIES

THIRD CYCLE OF ACADEMIC STUDIES – DOCTORAL STUDIES

DOCTORAL DISSERTATION TOPIC:

“A real-time integration of semantics into heterogeneous sensor stream data with context in the Internet of Things”

Candidate:
MSc. Besmir Sejdiu

Mentor:
Assoc. Prof. Dr. Florije Ismaili
Co-Mentor: Prof. Dr. Lule Ahmedi

Tetovo, February 2022

Abstract (EN)

Because of its importance for the development of many areas, such as environmental monitoring (observing of air quality, tracking of weather alerts, monitoring of water quality, and so forth), cities, healthcare, homes, energy systems, traffic control, and industry, the Internet of Things (IoT) is a dynamic area of study. Sensors are one of the fundamental elements that enable IoT, as they generate an ongoing sensor stream and send it to a central server, and their processing necessitates a unique method due to their huge volume. Additionally, extracting the context-specific data required for situational awareness from sensor stream data is exceptionally hard, even more so when real-time computation and interpretation are required. Furthermore, the discovery, access, and control of all different sensors and sensor stream observations through the internet are enabled by Sensor Web (SW), which incorporates the technologies of Semantic Web to form the Semantic Sensor Web (SSW). The interpretation and comprehension of sensor stream data and metadata are facilitated by annotating sensor stream data with semantic containing domain-specific concept definitions (e.g., ontologies). *The term "non-real-time semantic annotation"* refers to the process of storing sensor data in a repository (data store) as static data and then integrating it with semantics, whereas "*real-time semantic annotation*" refers to the process of integrating sensor stream data (as dynamic data) with semantics, which is the goal of this study. Recently, industry standards such as Sensor Web Enablement (SWE) were proposed by institutions including the World Wide Web Consortium (W3C) as well as the Open Geospatial Consortium (OGC).

This dissertation begins by conducting an in-depth examination of the incorporation of semantic information into the heterogeneous sensor data for application domains of IoT . The performed review analyzes the primary options for trying to add semantic comments to sensor data streams, the norms that facilitate all kinds of sensor information to be viewed on the internet, existing models of sensor data stream annotations, and IoT pattern domains that employ semantic annotations.

Then, the advanced annotation techniques for integration and interpretation of the semantic annotations in real-time into heterogeneous sensor observation data and metadata with context in the IoT has been introduced. Spark Streaming, Apache Kafka, and the Apache

Cassandra, as well as norms as SWE Sensor Observations Service (SOS), are used in this context. Next, an integrated system called IoTSAS (IoT Semantic Annotations System) is developed to evaluate the proposed techniques. It examines observed sensor data by integrating and interpreting semantic annotations in real-time. Finally, by extending the SWE standards, correspondingly the SOS standards, IoTSAS system testing is done in the IoT domains of air quality, weather warnings, and water quality monitoring. This dissertation also includes the findings of the system's performance when processing 1,000,000 observed sensor stream in real-time at a time.

Abstrakt (SQ)

Interneti i Gjërave (ang. Internet of Things - IoT) është një fushë aktive e kërkimeve shkencore për shkak të rëndësisë së saj në zhvillimet e shumë fushave, duke përfshi monitorimin e mjedisit (monitorimin e kualitetit të ajrit, monitorimi i sinjalizimit të motit, monitorimi cilësisë së ujit, etj.), shëndetësinë, qytetet, sistemet e energjisë, kontrollin e trafikut, industrinë, etj. Rrjetet e sensorëve pa tela (ang. Wireless Sensor Networks - WSNs) janë një nga teknologjitë kryesore që mundësojnë IoT-në, të cilat prodhojnë vazhdimisht të dhëna rreke të sensorëve dhe i transmetojnë këto të dhëna në një server të centralizuar, dhe si rezultat i vëllimit të madh të të dhënave, procesimi i tyre kërkonë një trajtim të veçantë. Gjithashtu, nxjerrja e informacionit kontekstual thelbësor për njohuritë e situatës nga të dhënat rreke të sensorit është shumë e vështirë, veçanërisht kur procesimi dhe interpretimi i këtyre të dhënave kërkohet që të bëhet në kohë reale. Për më tepër, të dhënat rreke të sensorit mundësohen në ueb përmes Sensor Ueb-it (ang. Sensor Web - SW), i cili duke inkorporuar teknologjitë e Uebit Semantik (ang. Semantic Web) krijon Uebin e Sensorëve Semantikë (ang. Semantic Sensor Web – SSW). Prandaj, duke shtuar anotime semantike në të dhënat rreke të sensorit me definimet e koncepteve nga domeni i njohurive (p.sh. ontologjitë), mundësohet interpretimi dhe kuptimi i të dhënave të sensorit dhe meta të dhënat e tij. Të dhënat rreke të sensorit që paraprakisht janë ruajtur në një depo të të dhënave, si të dhëna statike, dhe pastaj integrohen me semantik është definuar si *anotim semantik në kohë jo reale* (ang. *non-real-time semantic annotation*), ndërsa integrimi me semantikë në kohë reale i të dhënave rreke të sensorit, si të dhanë dinamike, është definuar si *anotim semantik në kohë reale* (ang. *real-time semantic annotation*) e që është edhe fokusi i këtij studimi. Së fundit organizatat si World Wide Web Consortium (W3C) dhe Open Geospatial Consortium (OGC) dhe kanë propozuar standarde të industrisë të tilla si Sensor Web Enablement (SWE), që ka për qëllim sigurimin e standardeve të unifikuara.

Në këtë disertacion, fillimisht është paraqitur një përmbledhje sistematike e literaturës rreth integrimit të semantikës në të dhënat rreke të sensorit për IoT. Rishikimi i literaturë është përqendruar në analizimin e zgjidhjeve kryesore që janë bërë në shtimin e anotimeve semantike në të dhënat rreke të sensoreve, standardet që mundësojnë të gjitha llojet e të dhënave të sensorëve të qasen nga uebi, modelet ekzistuese të anotimeve në të dhënat rreke të sensorëve dhe trendët e domeneve të IoT-së që përdorin semantiken.

Pastaj, janë paraqitur teknikat e avancuara për integrim dhe interpretim të anotimeve semantike në kohë reale në të dhënat e observuara heterogjene të sensorëve dhe meta të dhënat e tyre me kontekst në IoT. Në këtë kontekst janë utilizuar teknologjitë e tilla si Apache Kafka, Spark Streaming dhe Apache Cassandra (si bazë e të dhënave), si dhe standardet SWE Shërbimet e Observimeve të Sensorit (ang. Sensor Observations Service - SOS). Për t'i validuar teknikat e propozuara, është zhvilluar një sistem i integruar i quajtur IoTSAS (IoT Semantic Annotations System), i cili proceson në kohë reale të dhënat rreke të sensorëve duke i integruar me anotimet semantike dhe duke i interpretuar ato. Në fund është bërë testimi i sistemit IoTSAS ne domene të IoT-së, si në monitorimin e kualitetit të ajrit, monitorimin e sinjalizimeve të motit dhe monitorimin e cilësisë së ujit, duke i zgjeruar standardet SWE, respektivisht standardet Sensor Observations Service (SOS). Gjithashtu, rezultatet e performancës së sistemit duke procesuar 1,000,000 të dhëna rreke të sensorëve në të njëjtë kohë, janë paraqitur në këtë disertacion.

Апстракт

Интернетот на нештата (ИОТ) е активна област на научно истражување поради неговата важност за развојот на многу области, вклучително и мониторинг на животната средина (мониторинг на квалитетот на воздухот, следење на временските сигнали, следење на квалитетот на водата, итн.), Здравствена заштита, градови, домови , енергетски системи, контрола на сообраќајот, индустрија итн. Безжичните сензорски мрежи (WSN) се една од главните технологии што овозможуваат IoT, кои произведуваат континуирани податоци за проток на сензори и ги пренесуваат овие податоци до централизиран сервер, и како резултат на нивниот голем волумен , обработката бара посебен пристап. Исто така, извлекувањето на контекстуалните информации од суштинско значење за ситуационото знаење од податоците од сензорот е многу тешко, особено кога обработката и толкувањето на овие податоци е потребно во реално време. Понатаму, податоците од сензорот се овозможуваат на веб преку сензорската мрежа (SW), која со вградување на технологии на семантичка мрежа создава семантичка сензорска мрежа (SSW). Затоа, со додавање на семантички прибелешки на податоците од сензорот со концептни дефиниции од знаење на доменот (на пример, онтологии), се овозможува толкување и разбирање на податоците и мета протокот на сензорот. Податоците за сензорски струи кои се складираат во складиштето (складиште на податоци) како статични податоци, а потоа се интегрираат со семантика се дефинираат како семантички прибелешки во реално време, додека интеграцијата во реално време на сензорските податоци како динамични податоци со семантика е дефинирана како реална -временска семантичка прибелешка која е целта на оваа студија. Неодамна организации како World Wide Web Consortium (W3C) и Конзорциум за отворен геопростор (OGC) предложија индустриски стандарди како што е Sensor Web Enablement (SWE), кои се насочени кон обезбедување унифицирани стандарди.

Во оваа дисертација, првично е обезбеден систематски преглед на интеграцијата на семантиката во податоците за сензорот за IoT. Спроведениот преглед е фокусиран на анализирање на главните решенија за додавање на семантички прибелешки на податоците од сензорот, стандарди што овозможуваат прегледување на веб од сите типови на сензорски податоци, излегување од модели на прибелешки за податоци од сензори, и домени на тренд на ИОТ кои користат семантика.

Потоа, воведени се напредните техники на прибелешка за интеграција и толкување на семантичките прибелешки во реално време во хетерогени податоци за набervationудување на сензори и метаподатоци со контекст во ИОТ. Во овој контекст, се користат технологии како што се Apache Kafka, Spark Streaming и Apache Cassandra база на податоци, како и стандарди како SWE Sensor Observations Service (SOS). Следно, за да се потврдат предложените техники, се развива интегриран систем наречен IoTSAS (IoT Semantic Annotations System), кој ги обработува податоците од сензорот во реално време со интегрирање на семантички прибелешки и ги толкува. Конечно, тестирањето на системот IoTSAS се врши во мониторинг на квалитетот на воздухот, мониторинг на временски сигнали и мониторинг на квалитетот на водата во домените на ИоТ, со проширување на стандардите SWE, соодветно на стандардите за Сензор за набудување на сензори (СОС). Исто така, резултатите од перформансите на системот со обработка на податоци од 1.000.000 сензори во реално време во исто време, се претставени во оваа дисертација.

Declaration

I hereby declare that my Dissertation

“A real-time integration of semantics into heterogeneous sensor stream data with context in the Internet of Things”

has been written entirely by myself. This work never was presented for a diploma or degree at a university before

The research was carried out at the SEEU under the supervision of Assoc. Prof. Dr. Florije Ismaili, and co-mentor Prof. Dr. Lule Ahmedi.

MSc. Besmir Sejdiu

____

Proofreading declaration

I hereby declare that I have proofread PhD thesis with the title

“A real-time integration of semantics into heterogeneous sensor stream data with context in the Internet of Things”

to the best of my ability, and as such, it meets the criteria for being defended and published.

Respectfully,

Acknowledgments

To begin, I would like to express my gratitude to my advisors, Prof. Dr. Lule Ahmedi and Assoc. Prof. Dr. Florije Ismaili, without whom the success and completion of this dissertation would have been more difficult.

I thank my friends for the support they have given me at every moment.

Finally, but certainly not least, I would like to convey my heartfelt appreciation to my family, to whom I owe a great deal for the successful commencement and completion of this challenging and beautiful dissertation.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AAL	Ambient Assisted Living
AHA	Active and Healthy Aging
API	Application Programming Interface
AQI	Air Quality Index
BIB	Bibliography Document
CSV	Comma Separated Values
DStream	Discretized Stream
EPA	Environmental Protection Agency
EU	European Union
HDFS	Hadoop Distributed File System
HMIK	Hydrometeorological Institute of Kosovo
HTML	HyperText Markup Language
IDS	Invalid Data Streams
IoT	Internet of Things
IoTSAS	IoT Semantic Annotations System
JSON	JavaScript Object Notation
O&M	Observation & Measurement
OGC	Geospatial Consortium
OWL	Web Ontology Language
PDS	Processor Data Streams
RDD	Resilient Distributed Dataset
RDF	Resource Description Framework

RDFa	Resource Description Framework in attributes
REST	Representational State Transfer
RTISA	Real-Time Interpreting Semantically Annotated
RTSA	Real-Time Semantic Annotation
SAS	Sensor Alert Service
SAWSDL	Semantic Annotations for WSDL and XML Schema
SEL	Semantic Enablement Layer
SensorML	Sensor Model Language
SML	Sensor Model Language
SOAP	Simple Object Access Protocol
SOS	Sensor Observation Service
SOSA	Sensor Observation Sampler Actuator
SPS	Sensor Planning Service
SSN-XG	Semantic Sensor Networks Incubator Group
SW	Sensor Web
SWE	Sensor Web Enablement
TML	Transducer Model Language
UNECE	United Nations Economic Commission for Europe
URN	Uniform Resource Name
USN	Ubiquitous Sensor Network
UUID	Universally Unique Identifier
W3C	World Wide Web Consortium
WDS	Working Data Streams

WDSA	Working Data Stream Annotations
WFD	Water Framework Directive
WMD	WSNs Metadata
WNS	Web Notification Services
WPS	Web Processing Service
WSDL	Web Services Description Language
WSN	Web Notification Services
WSNs	Wireless Sensor Networks
XLink	XML Linking Language
XML	Extensible Markup Language
XPath	XML Path Language
XHTML	Extensible HyperText Markup Language

List of equations

List of Terms and Definitions

The next section contains a glossary of significant terminology used for this thesis. These terms are defined as they are used in this context.

Internet of Things	The IoT is a network of items or "things" which are integrated in electronics, software, and sensors that allow them to sense their surroundings and collect/exchange data through network infrastructure.
Sensor	The sensor is a monitoring device that converts physical phenomena such as heat, light, motion, vibration, sound, pressure, and other similar phenomena into an electrical signal that can be read by an instrument or an observer, and then sends the data collected.
Sensor stream data or sensor observation data	Sensor stream data or sensor observation data is generated by all sensor types and transmitted to a remote server in a continuous time-stamped format.
Sensor metadata	Sensor metadata is the data that describes the sensors, their devices, and the site allocation data that goes with them.
Static sensor	The static sensor is stationed in a fixed position to conduct monitoring operations in the target area.
Mobile sensor	The mobile sensor is used to monitor various ad-hoc sites of interest.
Homogeneous sensors	Homogeneous sensors can monitor for a certain type of event, such as carbon monoxide.
Heterogeneous sensors	Heterogeneous sensors collect data from multiple types of phenomena, such as humidity, ozone, carbon monoxide, and so on.

Sensor Web	The Sensor Web enables the discovery, access, and control of all different sensors and sensor stream information through the internet.
Semantic Sensor Web	The SSW is made by combining SW and Semantic Web technologies to provide better meaning for sensor stream data and enabling situation awareness.
OGC standard	The OGC defines the SW as a set of standards that enable the use of WSNs connected to a communication network.
Fixed sliding window	The fixed sliding window shows only the most recent data or only the most recent data based on a timeframe or a fixed window length.
Outlier sensor stream data	Outlier sensor stream information is a kind of anomalous sensor stream information that does not follow the expected pattern, which can be noise or data with mistakes.
Phenomenon or parameter	Temperature, carbon monoxide, humidity, ozone, pressure, nitrogen dioxide, sulfur dioxide, and other physical properties are examples of phenomena or parameters that can be sensed using sensors.
Sensing node	A sensing node, sometimes known as a mote, is a low-powered device with sensors connected. The sensing node transmits the data obtained from sensors to the gateway node.
Gateway node	A gateway node is a critical component of a wireless network system that collects data from sensing nodes and sends it to a central monitoring node, such as a remote server.

Central monitoring node	The gateway node sends all of the sensors' observed data to the central monitoring node, which processes it.
Deployment site	The deployment location is the area where the sensor nodes are distributed.

Part I.

Fundamentals and Related Work

1

Chapter

1. Introduction

1.1. Context

Smart Infrastructure systems in cities, healthcare, homes, water networks, grids, and intelligent transportation are today increasingly diverse and rich than we ever anticipated. The IoT has been typically associated with a more traditional view of such systems (Atzori, 2010). The IoT is a network of devices or "things" that are equipped with integrated technology (electronics, intelligent sensors, and software) and are capable of collecting data. The Internet of Things (IoT) enables remote sensing and control of physical objects via a network infrastructure, enabling a more direct integration of the physical world and computers. To put it another way, the IoT has resulted in automation in all industries (Santhi, 2016), (Begum, 2016), (Bera, 2016).

Due to the fact that the notion of the IoT was developed concurrently with the creation of Wireless Sensor Network (WSN), WSNs are the fundamental elements that enable IoT. A WSN is a collection of self-contained, geographically distributed devices that employ sensors to observe environmental or physical factors' (Yinbiao, 2014), (Lazarescu, 2017).

A wide range of environmental conditions, including humidity, pressure, temperature, vehicle movement, lightning condition, soil composition and noise levels, are monitored by WSNs in the army for earth observation, emergency management, fire alarm sensors, sensors planted underground for precision farming and intrusion detection (Akyildiz, 2010), (Bakaraniya, 2012).

While WSNs are commonly implemented with fixed sensor nodes to perform surveillance operations in a defined area, they can also be placed with mobile nodes to do surveillance in multiple places. In nature, WSNs are heterogeneous or homogeneous. While heterogeneous sensors transmit a variety of data types (e.g. carbon monoxide and nitrogen dioxyde), homogeneous WSNs broadcast only a single observation data (for example, the water humidity). Every one of these WSNs transmits observed information to the server via a sensor data stream. Sensor metadata is information on the WSN, its instruments, and the associated location data.

The SW makes sensor data available to the internet. SSW is produced by incorporating semantic web technology. As a result, "a sensor data stream can be *annotated with semantics* (for example, domain knowledge) by providing machine-interpretable descriptions of what the data represents, where it comes from, how it can be related to its surroundings, who is providing it, and what quality, technical, and non-technical attributes it has" (Barnaghi, 2012). "*Non-real-time semantic annotation"* is defined as the storage of sensor data as static data in a repository (data store) and then integration with semantics, whereas "*Real-time semantic annotation"* is described as the real-time integration of semantics into sensor stream data as dynamic data. Sensor data standards have been proposed by groups such as the W3C[1] and the OGC[2], which are covered in the following sections.

1.2. Problem description

Sensors are a critical component of the Internet of Things. Numerous sensors continuously generate a variety of perceived data kinds in the sensor stream data. Given the absence of norms, extracting helpful info from the large volumes of information accessible is difficult. The OGC suggested SWE and related standards to characterize the SW (Bröring, 2012). Since the publication of these standards are entirely syntactic, they fall short of enabling knowledge-based reasoning and discovery, as they do not adequately represent data essence relationships (Ji, 2014).

[1] https://www.w3.org
[2] http://www.opengeospatial.org

The researchers propose the Semantic Sensor Web (SSW) to gain additional information and knowledge by integrating meaningful annotation with idea definition from domain knowledge (eg. ontologies), that enables the understanding and comprehension of sensor technologies data and metadata (Sheth, 2008).

The W3C provided numerous mechanisms for tagging observation data, notably:

- Xlink - XML Linking Language
- SAWSDL - Semantic Annotations for WSDL and XML Schema
- RDFa - Resource Description Framework in Attributes

However, the subject of how to develop strategies for real-time incorporation and understanding of semantic annotations remains open (W3C, 2010), (Henson, 2009a), (Sheth, 2008), (Pu, 2016), (Ji, 2014). The primary objective of this research is to investigate semantic annotation strategies in this setting.

1.3. Hypothesis

The hypothesis for this proposal are:

NULL. Annotation techniques can be advanced for integration and interpretation of the semantic annotations in real-time into heterogeneous sensor observation data and metadata with context in the Internet of Things.

I. The model of real-time data stream processing can be extended for supporting techniques of real-time integration and interpretation of semantics into heterogeneous sensor observation data and sensor metadata with context in the Internet of Things.

II. The real-time semantic annotation can improve usability and performance of the Internet of Things applications.

1.4. Research objectives

The following are the key objectives of this research:

Objective 1. Devise annotation techniques for real-time integration of semantics into heterogeneous sensor observation data and sensor metadata with context in the Internet of Things.

Objective 2. Devise techniques to enable interpreting semantically annotated of the context, mentioned above.

Objective 3. Develop a prototype application that demonstrates the utility of proposed research idea, which will be tested in a certain(s) of the following IoT domains such as: Air Quality Monitoring and Smart Water Monitoring.

1.5. The importance of this study

According to experts, by 2030, there will be 500 billion connected devices/things. (Cisco, 2016). That means that compatibility between "Things" over the Internet of Things is a prerequisite for tracking, object addressing, finding, as well as the representation, storage, and interchange of information (Barnaghi, 2012). One of the primary difficulties that should be addressed in the future is achieving absolute uniformity and generalization. Different applications have their own set of knowledge that is incompatible with that of others. The granularity of the descriptions would differ even if the seen item was the same. Industry standards such as SWE have been proposed by organizations such as OGC and W3C, with the goal of creating unified standards (Shi, 2018). (Sheth, 2008) and the W3C offered a number of strategies for annotating observed data, including XLink, RDFa, and SAWSDL .

It is feasible to improve interoperability and give contextual information necessary for situational understanding by combining recognized standards with semantic expression forms. As a result, the sensors will reveal more information than they detect.

In this study, it was discovered that the majority of offered solutions used the RDFa annotation technique to semantically annotate in stream data (Sheth, 2008), (Henson, 2009a), (Compton, 2009), (Babitski, 2009), (W3C, 2010), (Vera, 2014), (Pradilla, 2016), (Bytyçi, 2017). Since the data from the sensors recorded as static data in a data storage and is later merged with semantics, the offered solutions required non-real-time semantic annotation. However, given the rapid growth of the IoT and its incorporation with advanced analytics, it is necessary to enhance techniques for real-time semantic annotation incorporation and understanding (Ji, 2014), (Sheth, 2008), (Henson, 2009a), (W3C, 2010), (Pu, 2016). As a result, this research is crucial for creating ways for real time integrating semantics into heterogeneous observed sensor stream in the IoT.

1.6. Research methodology

The procedure defined in (Petersen, 2008), is used to conduct the systematic literature review (SLR) presented in this paper. The research objectives are created first, followed by a technique for finding relevant publications in significant digital libraries in computer science, and finally, the criteria for admission and disqualification.

1.6.1. Research Questions

Initially as an important step in this research is the translation of the review's goal into survey questions. Table 1 encapsulates the questions and their reasons, together with the semantic sensor that was used. Web technologies and major solutions for annotating sensor data with semantics, sensor web standards, stream processing models with semantic integration, and semantic Internet of Things trend domains.

Table 1. Research questions and incentives

#	Research Question	Incentive
RQ1	Which Semantic Sensor Web technologies are most frequently used, as well as the most common methods for adding semantic annotations to sensor data?	This query serves as a jumping-off point for a discussion on the Semantic Sensor Web. The answer to this question summarizes recent research and answers.
RQ2	What are the standards that enable the discovery, access, and use of all different sensors and sensor data sources over the Internet?	The response summarizes current standards that define a defined web service interface that enables clients to obtain descriptions of linked sensors and their collected data.
RQ3	What models are capable of analyzing data streams in real time and integrating semantics?	There are a variety of models that support real-time data stream processing, and the answer to this question can help us determine which models are most suited to this task depending on their performance.
RQ4	What are the semantics-based IoT trend domains?	IoT trends employing semantic annotation can be identified by answering this question.

1.6.2. Search process

The search procedure is divided into four steps, as indicated in Figure 1, to find relevant and helpful material.

The following sections detail each phase.

Phase 1

The research topic's search phrases are defined first, including the primary keywords from the survey questions and their equivalents. The following table contains sample search phrases for the keywords SW standards, SSW, sensor stream data semantic models, and Internet of Things semantic trend developments. In order to generate a more expressive query, the `OR` operator is used to connect alternate phrases and the `AND` operator is used to join the major components, as illustrated in Table 2.

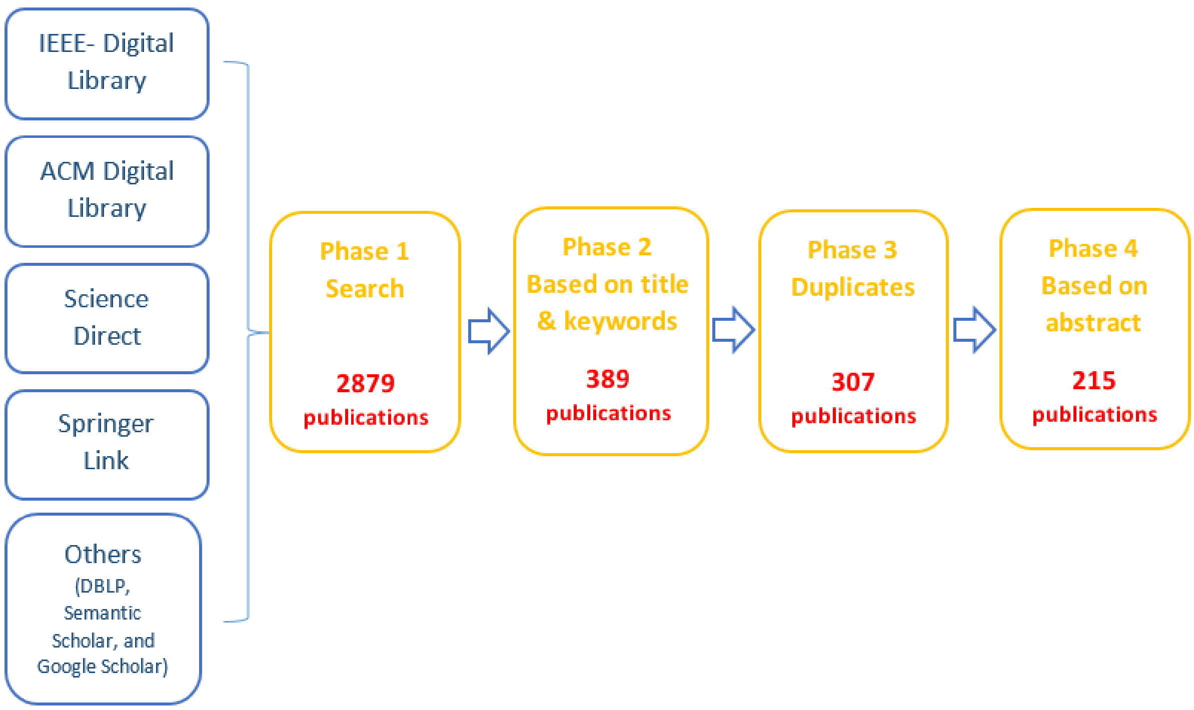


Figure 1. The process of articles' selection

Table 2. Sample search strings

Search term	Search string
Semantic Sensor Web	(("Semantic" AND ("web" OR "OWL" OR "Ontology")) OR ("SSW" OR "SWRL" OR "SSN" OR "SPARQL")) AND ("Data Stream" OR "Sensor Data")
Standards of Sensor Web	("Standard" OR "Sensor Observation Service" OR "SOS" OR "Sensor Web Enablement" OR "SWE" OR "Observations & Measurements" OR "O&M") AND ("Sensor" OR "Wireless Sensor Network" OR "WSN")
Data Stream Models	("Data Stream Model") AND ("Real-Time Processing") AND ("Sensor Data")
Semantic IoT trends	("Internet of Things" OR "IoT") AND ("Semantic" OR "OWL" OR "Ontology" OR "SSW" OR "SWRL" OR "SSN" OR "SPARQL")

In December 2021, an article search was conducted. To provide a thorough review of this academic subject, journals, conferences, books, and technical reports from various important digital computer science libraries have been involved in the review's search approach.

- **IEEE Digital Library[3],**
- **ACM Digital Library[4],**
- **Science Direct[5],**
- **Springer Link[6],**
- **DBLP computer science bibliography[7].**
- **Semantic Scholar[8],**
- **Google Scholar[9]**

Using the sample search phrases in Table 2, an advanced search of the aforementioned digital libraries returned 2,879 articles. At first, a scan of the Springer Link digital library uncovered 4,288 publications. Only 486 of these publications were chosen for additional investigation based on their titles.

It's worth noting that no filter was applied to the published year of publications during this phase. Because these include the publications of other digital libraries mentioned above, the publications discovered in the IEEE, ACM, Science Direct, and Springer Link digital libraries have been excluded from the search results of DBLP, Semantic Scholar, and Google Scholar (IEEE, ACM, Science Direct, and Springer Link).

The ability of digital libraries to export search results in files such as *csv* (IEEE, ACM, and Springer Link), *bib* (Science Direct), and *xml* (DBLP) has been used to finish this phase.

The Publish or Perish software is used to export Google Scholar results. As illustrated in Figure 2, all files downloaded from computer science libraries are migrated into a MS SQL Server database using a developed custom-built program. The outcomes of categorizing, grouping, filtering, sorting, and other operations can be simply modified with this tool.

[3] https://ieeexplore.ieee.org/Xplore/home.jsp
[4] https://dl.acm.org
[5] https://www.sciencedirect.com
[6] https://link.springer.com
[7] http://dblp.uni-trier.de
[8] https://www.semanticscholar.org
[9] https://scholar.google.com/

In the following phases, the publication list for final analysis is selected using inclusion and exclusion criteria.

Phase 2

389 papers were chosen on the basis on the titles and phrases associated with the topic of study.

Phase 3

There were 82 duplicate publications found in multiple digital libraries, which were removed, leaving 307 papers for the next step.

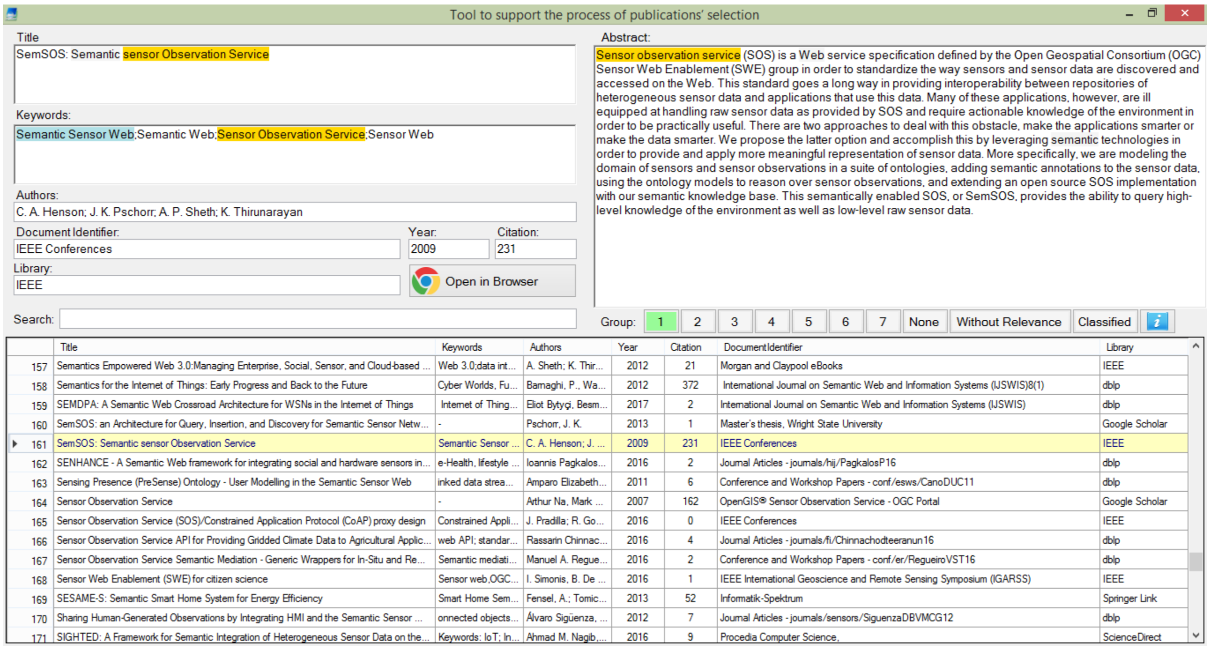


Figure 2. The tool that assists in the selection of articles

Phase 4

Finally, the relevant papers were chosen based on the abstracts. Furthermore, the intro, section headings, and conclusion analysis of the publications were required for proper selection. There are a total of 215 publications considered in this analysis. The whole list of selected papers is available for download at the following link[10].

1.6.3. Inclusion and exclusion criteria

Publications that are irrelevant to the review's research topics are excluded using inclusion and exclusion criteria. The papers selected using the process outlined above were evaluated and categorized as per the research subjects specified in this research for further investigation. The list consists of the inclusion criteria:

- Research on the Semantic Sensor Web, suggested in particular IoT systems that combined semantics with observed sensor data.
- Research into the use of standards as a mechanism for allowing applications to access IoT data.
- Research into models that enables real-time observed sensor data processing with semantics.

Exclusion criteria consists of: Exclusion criteria consists of:

- 2597 papers which aren’t immediately apparent to the study's subject.
- 11 publishing of brief essays (fewer than three pages).
- 6 articles were replicated (papers presented at conferences that were subsequently published in the journal).
- 43 articles in the form of tutorials or demonstrations.
- 7 articles written in languages other than English.

1.7. Challenges

On the internet, real-time dynamic data collected via numerous sensors is now accessible. The primary impediments to real-time semantic annotation include intricacy, versatility, standardisation, and generalisation, as well as the large amount of unorganized sensor data streams. Additionally, as a result of diverse, scattered, inadequate information representation

[10] http://luleahmedi.uni-pr.edu/students/bsejdiu/selectedreviewpapers.xlsx

and non-standard infrastructure, numerous sensor data streams have already been locked within private applications, rendering them unavailable to the broader public.

The Sensor Web's application of semantics raises five difficulties (Corcho, 2010). One of the very first concerns is the abstraction required to gather, analyze, and manage sensor data in general. The next is the demand for data quality management that is appropriate. The third issue is the incorporation and integration of information from diverse and independently dispersed sensor networks. The fourth issue is identifying and localizing critical sensor-based data sources. Finally, rapid software based on a variety of sources of data is a challenge.

Semantic integration of diverse sensor data should result in improved comprehension and more relevant representations, enabling IoT potential uses to become significantly more intelligent (Shi, 2018).With advancements in technology, researchers are paying more attention to obstacles and issues in order to close gaps and handle them effectively in the future.

1.8. Organisation of this Dissertation

Following the introductory chapter, the structure of the dissertation is organized as follows:

Chapter 2 - explains the fundamentals of IoT, IoT data transmission models, IoT applications, Wireless Sensor Networks (WSNs), sensor stream data, and semantic annotations.

Chapter 3 - provides a literature review of the research questions. The study focuses on semantic annotation methodologies, the primary solutions for annotating sensor data with semantics, the web standards that support all types of sensors, the current stream models, and the semantic Internet of Things trend areas.

Chapter 4 - presents the selected technologies and standards of the real-time integrating and interpreting of semantic annotations into the observed WSN data, system architecture, main components and data modeling.

Chapter 5 - presents the proposed system's implementation modules, which include the real-time integrating and interpreting of semantic annotations into the observed WSN data module, data modelling module, the module for managing meta data, monitoring module for air quality, monitoring module for weather warnings, water quality monitoring module, and the module for external systems - RESTful APIs. Additionally, this chapter discusses the system's network architecture and the simulator for sensor stream data.

Chapter 6 - presents the IoT Semantic Annotations System (IoTSAS) system's testing results.

Chapter 7 - concludes the thesis by assessing and addressing the findings of the preceding chapters and identifying some areas for future research.

1.9. List of publications

The research papers that have been published during this dissertation are:

1. Besmir Sejdiu, Florije Ismaili, and Lule Ahmedi. (2020). "**Integration of semantics into sensor data for the IoT - A Systematic Literature Review"**. *International Journal on Semantic Web and Information Systems (IJSWIS).* Volume 16, Issue 4, Article 1.

   Web of Science Impact Factor 1.5

   Indexed In: Web of Science Science Citation Index Expanded (SCIE), SCOPUS, Compendex (Elsevier Engineering Index), INSPEC, DBLP, ACM Digital Library, etc.
   Editor-in-Chief: Amit Sheth, Wright State University, United States

2. Besmir Sejdiu, Florije Ismaili, and Lule Ahmedi. (2020). "**A real-time integration of semantics into heterogeneous sensor stream data with context in the Internet of Things"**. The 15th International Conference on Software Technologies (ICSOFT 2020). July 07 - 09, 2020, Lieusaint - Paris, France.

   Indexed In: DBLP, Thomson Reuters Conference Proceedings Citation Index, EI, SCOPUS, Semantic Scholar, Google Scholar and Microsoft Academic.

3. Besmir Sejdiu, Florije Ismaili, and Lule Ahmedi. (2020). "**A management model of integrated semantic annotations to the sensor stream data for the IoT"**. The *16th International Conference on Web Information Systems and Technologies (WEBIST 2020),* November 03 - 05, 2020, Budapest, Hungary.

   Indexed In: SCOPUS, DBLP, Thomson Reuters Conference Proceedings Citation Index, Semantic Scholar, Google Scholar and Microsoft Academic.

4. Besmir Sejdiu, Florije Ismaili, and Lule Ahmedi. (2021). "**A Real-time Integration of Semantic Annotations into Air Quality Monitoring Sensor Data"**. Communications in Computer and Information Science (CCIS), Software Technologies, Springer ICSOFT 2020, *Book Chapter,* Springer Nature, Switzerland.

   Indexed In: Scopus, DBLP, EI-Compendex, Mathematical Reviews, SCImago, Google Scholar, ISI Proceedings.

5. Besmir Sejdiu, Florije Ismaili, and Lule Ahmedi. (2021). “**IoTSAS: An integrated system for real-time semantic annotation and interpretation of IoT sensor stream data”**. *Computers 2021, Multidisciplinary Digital Publishing Institute (MDPI)*.

   Citescore 3.3 Scopus

   Indexed In: Scopus, ESCI (Web of Science), DBLP, Inspec, SCimago, etc.

6. Besmir Sejdiu, Florije Ismaili, and Lule Ahmedi. (2021). “**A real-time semantic annotations to the sensor stream data for the water quality monitoring”**. *SN Computer Science, Springer Link*. (*Accepted*)

   Indexed In: ACM Digital Library, DBLP, CNK, Dimensions, EBSCO Discovery Service, Japanese Science and Technology Agency (JST), etc.

2

Chapter

2. Fundamentals

2.1. Internet of Things (IoT)

The Internet of Things (IoT) refers to a wide range of devices or "things" which have never before been linked with the Internet, but that have now been given an identity and connected to the Internet via the IoT. Utility meters, Thermostats, Bluetooth headphones, pumps for irrigation, and sensors or electric vehicle motor control circuits are among the items included. Advances in sensor network capabilities, mobile devices, wireless communications, networking and cloud technologies have sparked a new revolution in the capabilities of Internet-connected endpoints.

According to Cisco predictions, by 2030 will be connected 500 billion objects and devices to the Internet (Cisco, 2016). As a result, companies are encouraged by the prospect of investing in the Internet of Things industry for their products. Products can consist of hardware or software as components of the Internet of Things (Bahga, 2014).

There is a lot more to the Internet of Things (IoT) than simply connecting things (appliances, machines, devices) to the Internet; there is a possibility that IoT devices will exchange data (control and information that may contain personal data about users) and perform useful tasks in pursuit of a common goal shared by users or machines. Before it can be turned into valuable information, raw data must be contextualized and processed. As part of the Internet of Things (IoT), software applications filter, analyse, categorize, condense, and contextualize data in order to produce and generate new information. After obtaining the necessary data, the information is organized and formatted to gain insight into the system and its users, as well as the environment in which the system operates, and the progress made toward the intended objectives. For instance, consider a stream of raw sensor values ((16.6, 42); (17.2, 49)) produced by a weather monitoring station that have no meaning or context on their own. Each tuple contains a piece of data that represents the temperature and humidity of the environment at a certain time interval. The measured data tuples have significance (or information) thanks to this added context. Classifying, compressing, or otherwise manipulating this data might provide further insights. Temperature and humidity values are averaged to provide an average of the past five minutes' worth of data tuples. Organizing and understanding the links between bits of information is the next stage in the process of gaining actionable knowledge. For instance, if the last five minutes' average temperature was greater than 48 degrees Celisus, an alert is raised and this notification can also be based on the user's geographical location as well (Bahga, 2014).

According to the (Suoa, 2012), the IoT has recently gained a lot of traction thanks to a few notable applications (e. g., meter reading, smart electric greenhouse monitoring, telemedicine monitoring, and intelligent transportation). Sensors, heterogeneous access, information processing, applications and services, as well as additional components such as security and privacy, make up the four major components of the Internet of Things (IoT).

2.2. IoT Data Transmission Models

With the Internet of Things, various gadgets and sensors will be linked together. IoT devices and sensors are connected and transmit data in different architecture models. In 2015, was created a guide by the Internet Architecture Board group about IoT networking and data transmission from IoT devices. This guide presents four types of the IoT device connected and data transmission models (Rose, 2015), (Ali, 2016), (Kaushik, 2016):

Device:

1. to Device,
2. to Cloud,
3. to Gateway, and
4. Back-End Communication.

2.2.1. Device-to-Device data transmission model

Device-to-device data transmission model provides connection and direct communication between two or more IoT devices without the interference of any server application. The communication can be done using different types of protocols such as ZigBee, Bluetooth, and Z-Wave, as shown in Figure 3. This type of model is commonly used in systems that require a small data packets, such as smart homes and IoT wearable devices that monitor human health related to smartwatch, where is not necessary to share information with other people. This type of data transmission model illustrates many of the interoperability challenges (Ali, 2016), (Kaushik, 2016).

In terms of security, each model of communication and data transmission has its own characteristics, but with the device-to-device data transmission model, security is specifically simplified because of the short-range technology (ZigBee, Bluetooth, Z-Wave) that they use. Communication between sensors and node devices is usually done through the ZigBee protocol.

2.2.2. Device to Cloud data transmission model

At device to cloud data transmission type, the Internet of Things devices are connected directly to the Internet through traditional communications such as Wi-Fi or wired Ethernet to communicate with cloud services or application service providers. This is shown in Figure 4.

This model is used by mobile sensors which directly send data to could services and application service providers. Sensors to send data to the server use protocols such as HTTP, IP, TCP, UDP, TLS, etc.

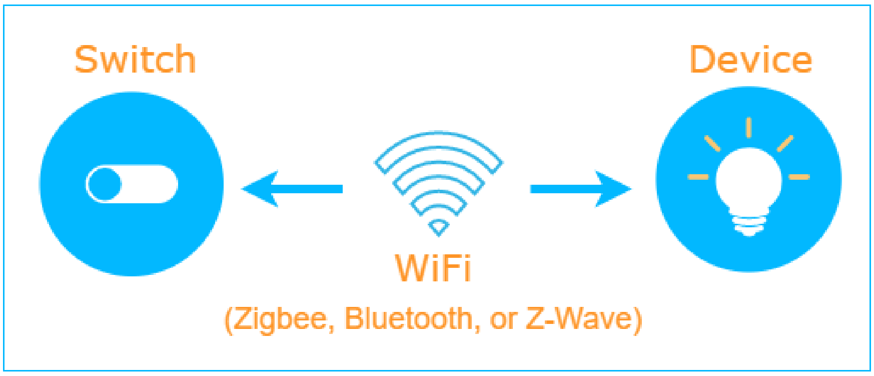


Figure 3. Device to Device data transmission type

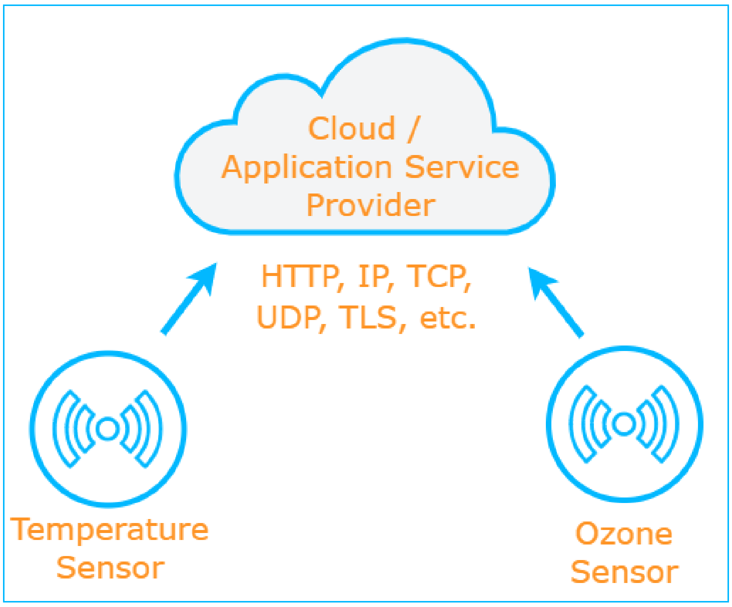


Figure 4. Device to Cloud data transmission type

2.2.3. Device to Gateway data transmission

In the device to gateway data transmission type, the Internet of Things sensors send data to the Gateway which is used in the capacity of a go-between sensors & the cloud services or application service providers, as shown in Figure 5. Communication and data transmission between IoT sensors and Gateway is enabled via Wi-Fi, ZigBee, Z-Wave, Bluetooth, HTTP, IP, TCP, UDP, TLS, etc. protocols, while the communication between Gateway and cloud services or application service providers is enabled through IP protocol. This model is found in many consumer devices. Smartphone app software serves as a middleware gateway for devices like personal fitness trackers, which can't communicate directly with cloud services, thus they often link via smartphone app software.

The device-to-gateway data transmission model, also is applied in smart home, by enabling the IoT devices to connect with the cloud services, allowing the users to control home devices using smartphone application.

2.2.4. Backend Data Sharing model

Smart object data from a could service can be exported and combined with data from different inputs using the backend data sharing paradigm. With this architecture, users are able to access sensor stream data that has been uploaded and stored in the cloud, as illustrated in Figure 6.

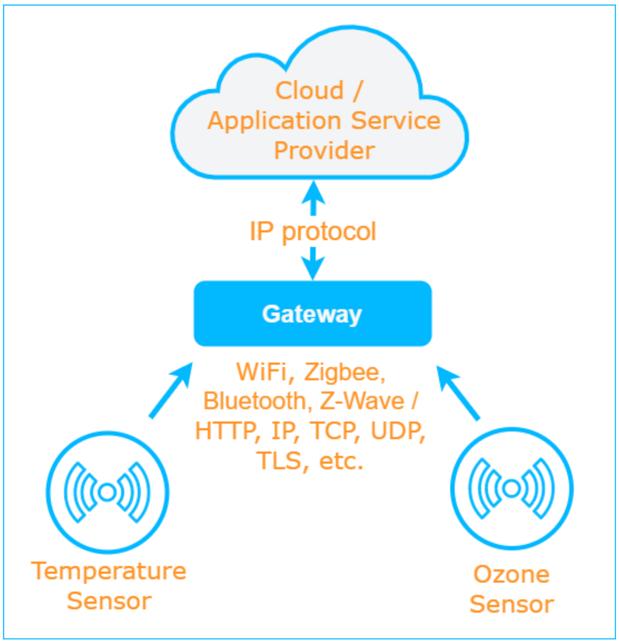


Figure 5. Device to Gateway data transmission type

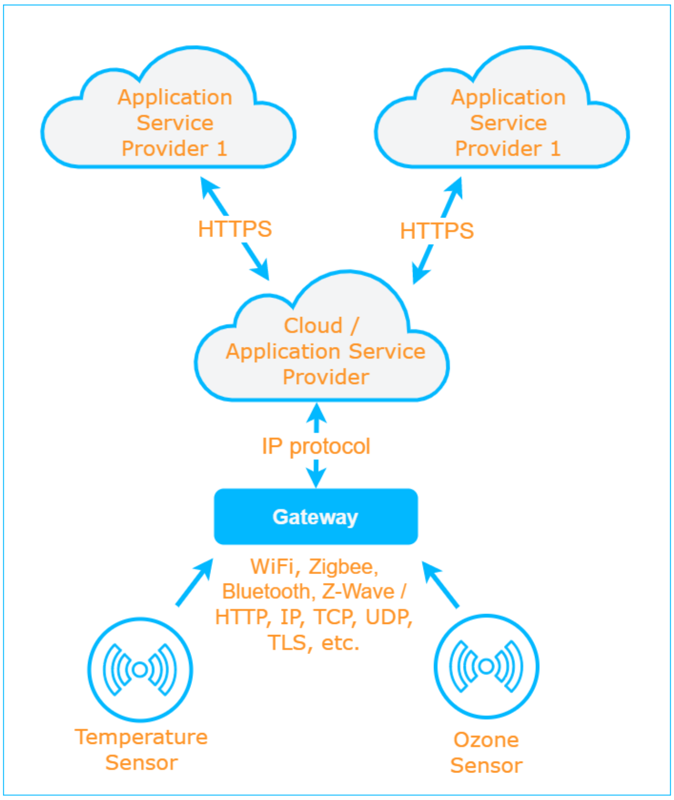


Figure 6. Backend data sharing type

To connect and transmit data from IoT sensors to gateway, the Wi-Fi, Bluetooth, ZigBee, Z-Wave, HTTP, IP TCP, etc. protocols are used, while the communication between gateway and application service providers is enables by IP protocol, respectively HTTPs protocol.

2.3. IoT Applications

As computer technology has progressed, little sensors and processors have been able to be integrated into common things. The desire of a smart environment is becoming a reality today thanks to advancements in fields such as WSNs, mobile appliances, Internet Protocol version 6, omnipresent computing, decision-making based on machine learning, mobile communications, agent technologies and human computer interactions. Connected sensor-enabled gadgets work together to make people's lives more pleasant in a smart environment. This means that in order to have a smart environment, it must be capable to learn and adapt responding to shifting needs of its inhabitants.

In Figure 7, are presented some IoT-based smart environments and described each of the domains.

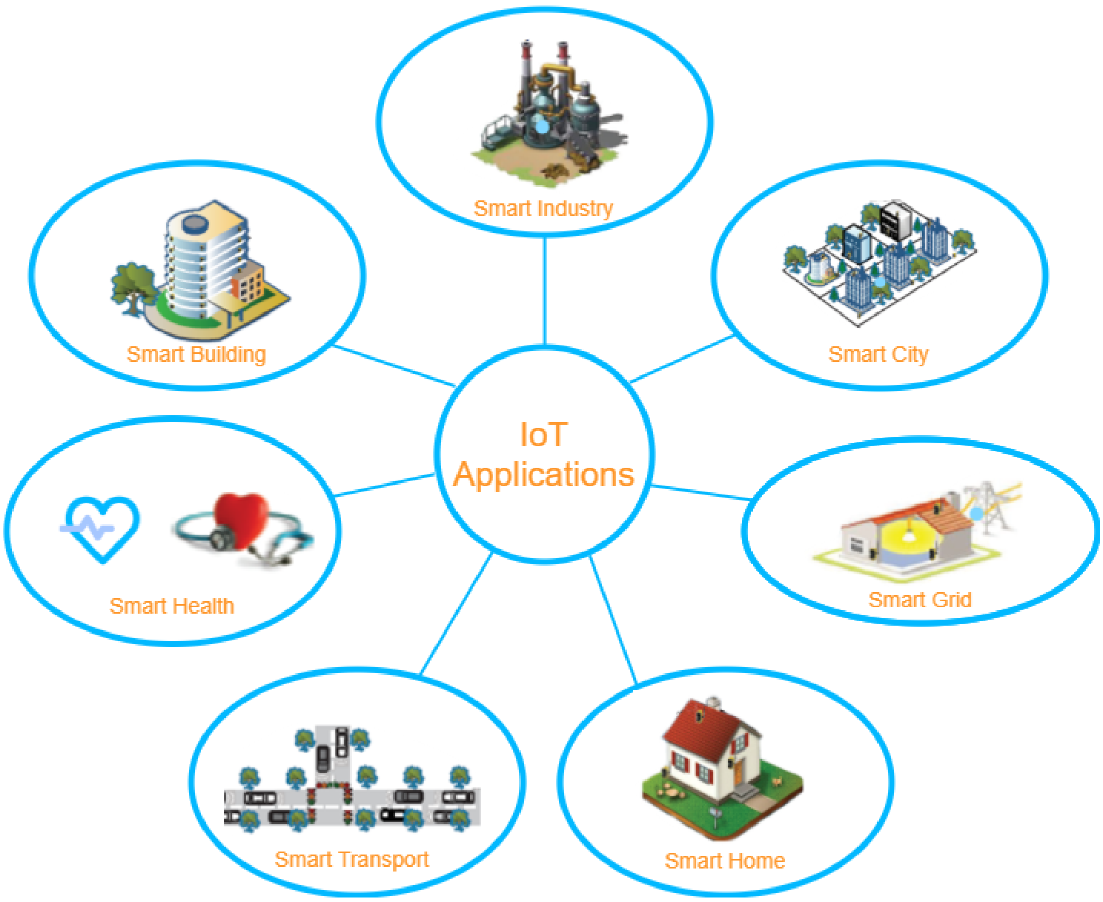


Figure 7. IoT Applications

Examples of IoT applications in several fields are included in the following list, demonstrating why the IoT is a key technology trend for the foreseeable future. (Vermesan, 2014):

- Smart Food / Water Monitoring
    - Distribution Network Control, On-Site Monitoring, Quality Of water, Water Leaks, River Foods, Water Management, and more.
- Health Care
    - Temprature Monitoring, Electrocardiogram (ECG) Monitoring, Asthma Monitoring, Fall Detection, Oxygen saturation Monitoring, Sleep Control, Glucose Level Monitoring, Physical Activity Monitoring, Blood Pressure (BP) Monitoring, and more.
- Smart Living
    - Apps for Smart Shopping, Gas Monitoring, Remote Control Devices, Water, Energy, Consumption and Energy and Water Use, and more.
- Smart Environment Monitoring
    - Air Pollution Monitoring, Deforestation prevention, Water Quality Monitoring, Flood Monitoring, Smart Agricultural Monitoring, Earthquake Monitoring, Forest Fire Detection, and more.
- Smart Manufacturing
    - o Intelligent Product Management, Tracking of Animals, Composting, Offspring Care, Toxic Gas Levels, Production Line, Telework, and more.
- Smart Energy
    - Flow Of water, Intelligent Grid, Solar and Wind Turbine Installations, Radiation Levels, and Power Supply Controllers, and more.
- Transport and Mobility
    - Intilligent Transport, Smart Lighting, Fleet Tracking, Smart Parking, Smart NFC Payment, Smart Traffic Lights, Electric Vehicle Charging Stations, Electric Mobility, Road Pricing, Green mobility, Vehicle Auto-diagnosis, Driving Safety, Sharing and Urban Mobility, and more.
- Smart Industry

- Smart Factory, Smart Construction, Aquaculture Industry Monitoring, M2M Applications, Smart Railway, Automotive Industry, Ozone Presence, Oil, Gas and Mining, and more.

- Smart City
    - Smart Parking, City Lighting, Smart Tourism, City Transit, Smart Buildings, Controlling Air Pollution, Intelligent Transportation Systems, Smart Meters & Billing, Safe City, and more.

2.4. Wireless Sensor Networks (WSNs)

These days, sensors may be found everywhere. Our cars, cellphones, factories, and even vineyard soil are all equipped with sensors to keep tabs on CO2 emission levels which we sometimes ordinarily assume. According to (Yinbiao, 2014), academic on WSNs started in 1980s, and only since 2001 have WSNs garnered increasing interest from both research and industrial perspectives. This is because inexpensive, low-power tiny components such as computers, radios, and sensors are frequently constructed on a single chip (system on a chip (SoC)).

The concept of the IoT has evolved in parallel with WSNs. Originally conceived by Kevin Ashton in 1999, the phrase "Internet of Things" refers to items that may be uniquely identified and their virtual representations in a framework that resembles the Internet. From big structures, industrial facilities, aircraft, and automobiles to small pieces of a larger system, these items may be anything. They can be anything from human beings, animals, and plants to individual bodily parts. WSN, in particular, will flourish in a wide range of applications and sectors regardless of the fact that the Internet of Things does not automatically imply a specific communication technology. IoT can be brought to even the tiniest things deployed in any setting, at a reasonable cost, thanks to WSN sensors that are compact, affordable, and low-power. These are combined in entities into the Internet of Things will be a significant evolution. an important step forward for WSN.

WSNs are networks of nodes that work together to perceive and manage the environment, allowing humans or computers to interact with their surroundings. In reality, a cross-layer design approach is usually required to incorporate distributed data processing, security control, and protocols of communication together when detecting has a finite quantity of energy.

By synthesizing current WSN applications into an infrastructure network, new apps may be found and built to match future market trends and technologies. There are a number of WSN technology applications that create a lot of data that may be used for various reasons, such as smart grid, smart water, intelligent transportation systems, and smart homes, for example. Many legal issues must be addressed as the IoT enters an age of WSNs in the contemporary world, and they will become clearer over time. The ownership and use of data gathered, collated, linked, and mined for extra value is one of the most important challenges.

The cost of WSN equipment has decreased dramatically due to the development of associated technologies, as well as various useage are increasingly spreading from army to commercial sectors. WSN equipment and standards for Wireless Sensor Networks technology are fully established such as Wireless Hart, ZigBee, Wireless Networking for Industrial Automation - Process Automation (WIAPA), etc. In addition, with the emergence of new WSN application modes in industrial automation and home applications, the overall size of the WSN applications market will continue to grow rapidly (Yinbiao, 2014).

Sensor, *Sensor Node*, and *Sensor Network* are the three key components of the WSN (Sohraby, 2007):

- *Sensor* - Is a transducer that transforms physical phenomena such as heat, light, motion, vibration, sound, and pressure into electrical signals. Sensors are frequently linked to a sensor node.

- *Sensor Node* - In a sensor network, a Sensor Node has embedded sensors, a processor, memory, a transceiver, and a power supply. Sensor nodes are occasionally connected to other sensor nodes, but most of the time they are connected via a wireless link to a gateway node. This type of network is known as a sensor network.

- *Sensor Network* - A sensor network is made up of several different sensor nodes. Nodes are placed inside or extremely close to the phenomenon being observed. The gateway sends the collected data via the internet to a distant server, guaranteeing that maintenance, database storage, and data processing (e.g. analytics) are all possible. This makes it easier for users to operate with a wonderful interface and ubiquitous connectivity (Arockiam, 2016).

Wireless Sensor Networks (WSNs) are made up of the gateway node, sensor nodes, and sensors, as depicted in Figure 8.

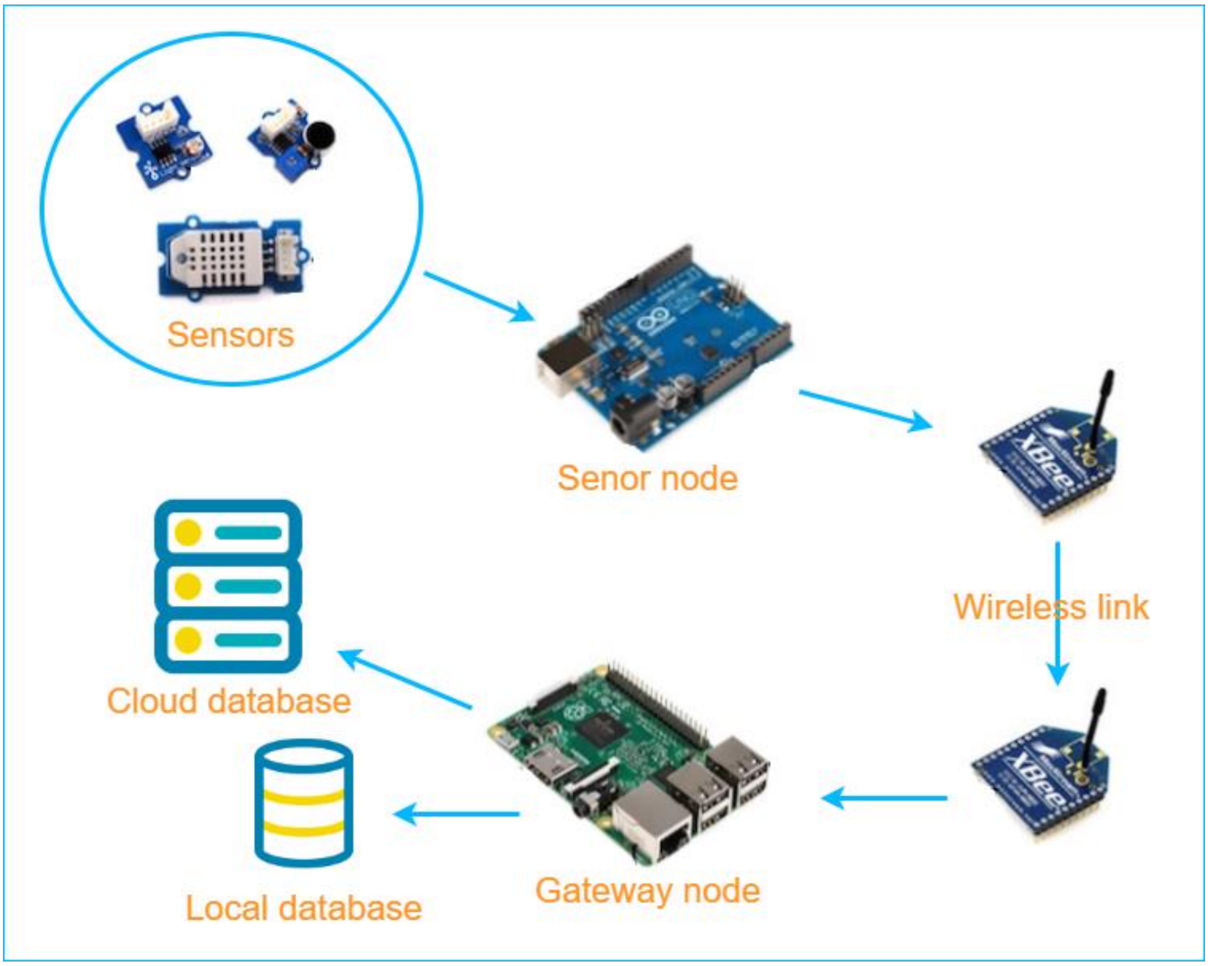


Figure 8. Wireless Sensor Networks (WSNs)

2.5. Sensor Stream Data

Sensors take a physical measurement and turn it into a signal that may be represented as a single value throughout time. Depending on the application and system requirements, the sensor sampling rate can range from milliseconds to hours.

A sensor data stream is a set of values having a timestamp associated with them. A timestamp in IoT refers to the time when a measurement was taken (Kenda, 2019).

In order to distinguish between sensor data streams and traditional data streams, it's significant to understand the distinctions between the two types of data streams (Gama, 2007), (Elnahrawy, 2003), and (Aquino, 2006):

- Sensor observed streams represent just a subset of the overall population, whereas typical streaming such as network streams, online log data, stock market data, and so on reflect the entire data population.
- In the context of typical streaming data, sensor observed streams are characterized as noisy. In the classical meaning, traditional data streams is accurate and error-free. Ever, the environment's influence on WSNs installed might have a detrimental effect on the data they collect. When compared to sensor network data, web clickstreams and web logs are regarded reliable.
- Large amounts of data from traditional streams are stored and processed in traditional streams, but sensor data streams are much smaller.

As previously stated, we will present a novel sensor data stream management paradigm, because sensor data transmission rates are low, and sensor data streams are typically smaller (e.g., sensors for water quality monitoring, air quality monitoring, etc.) than standard streaming data.

2.6. Semantic annotations

Heterogeneous sensors enable IoT applications data from sensors is sent to an offsite server, where it is processed and stored. Unless appropriately annotated, raw sensor stream data is meaningless. As a result, the researchers developed the SSW, which is a hybrid of SW and technologies of Semantic Web. According to our study "*Integration of semantics into sensor* *data for the IoT - A Systematic Literature Review*” a large number of studies accept suggest industry standards such as Sensor Web Enablement, as well as sensor data annotation approaches proposed by organizations like as OGC, such as RDFa, Xlink and SAWSDL. (Compton, 2012). But the difficulty remains: how can real-time semantic annotations be better integrated into strategies?

3

Chapter

3. Related Work

Following the extraction of pertinent publications, the selected primary sources are examined and assessed in accordance with our research questions. The following are the findings from the systematic review of the literature.

3.1. Which Semantic Senor Web technologies are most frequently used, as well as the most common methods for adding semantic annotations to sensor data? (RQ1)

The SW enables wireless sensor networks, hence giving solutions for web-enabled WSNs (Udayakumar, 2012), (Rouached, 2012). The SW idea exemplifies this form of sharing architecture, locating, in addition incorporating sensors and related information into a spectrum of uses (Bröring, 2011). It shields solutions that are developed on top of the heterogeneous sensor hardware and connection protocols. As a result, the SW acts as a middleware layer amongst detectors & applications (Auger, 2017).

To facilitate interchange and provide contextual information important for situational awareness, sensor readings might be tagged with semantic information in the form of *ontologies* – this is referred as the SSW. Additionally, they serve as a bridge between the SWE and the Semantic Web's RDF/OWL - grounded metadata standards, providing more meaningful descriptions of sensor data and better access to it (Sheth, 2008), (Calbimonte, 2013). Interoperability and analysis of heterogeneous multimodal sensor data rely heavily on ontologies and semantic annotation in SSW (Simonis, 2016), (Henson, 2014), (Le-Phuoc,2011), (Gray, 2011), (Corcho, 2012).

XLink, RDFa, and SAWSDL are examples of semantic annotation approaches (W3C, 2010). XML Linking Language is a World Wide Web Consortium recommendation for producing and associating metadata with hyperlinks in XML documents (Lefort, 2009). References to the others knowledge base controlled with URNs are regularly included in OGC standards (Uniform Resource Name). RDFa is a World Wide Web Consortium recommendation for embedding rich metadata in Web publications by adding a group of attribute-level additions to Extensible HyperText Markup Language. To give semantic annotations for sensor data, Resource Description Framework in Attributes can be placed to OGC O&M files. SAWSDL is a group of WSDL and XML Schema extension characteristics which enable for the characterization of added semantics of the WSDL files.

It's required to design and employ rules – known as reasoning – to infer new knowledge from sensor observed semantic annotations (Sheth, 2008).

The following is a list of the most important RQ1-related papers. Because they utilised OGC standards, several of them are also related to RQ2.

- According to (Sheth, 2008), Semantic Sensor Web is defined as a combination of observed sensor information and semantics. In order to give meaning to sensor data, the “Semantic Web Activity” of the OGC and the W3C created SSW. Under the SWE architecture, OGC has established and maintains a number of essential services, including SML, SOS, O&M, SPS, and SOS.
- According to the SWE standards, it is possible to construct Semantic Sensor Monitoring (SemSOS) (Henson, 2009a). Ontology modeling of the sensors and observed WSN data, and ontology-based reasoning over observed WSN data are some of the ways in which an ontology-based SWE implementation can be improved, they created a semantic knowledge base Open-source SOS is improved by them over 52North's version.

- Semantically annotating streams with IoT-Streams is described in (Elsaleh, 2020). Ontology IoT-Stream, as well as an expansion of the well-known SSN ontology, which provides a minimal semantic framework for stream annotations. As a result, IoT applications that deal with streaming sensory input are made easier to design.

- Semantic Web technologies are illustrated in (Patni, 2011) as a framework for integrating and analyzing heterogeneous sensor feeds. The goal of this framework was on creating meaningful abstractions or features in real-time from sensor stream data, and publishes these streams as linked data. The raw sensor stream data was transformed to an Observation and Measurements (O&M) format, and then it was converted to RDF stream.
- A methodology a confluence of data of series of heterogeneous sensor data streams is given in (Kenda, 2019), which supplements IoT sensor stream data with contextual and historical information important to understanding underpinning actions.
- As part of the SWE criteria, sensor data semantic annotation is incorporated in (W3C, 2010). The basic semantic annotation approaches are investigated and then it is suggested that RDFa with Sensor Observation Service, SAWSDL, and XLink be used.
- A semantic tagging and integrating framework for sensor services that are OGC-compliant is provided in (Babitski, 2009). The method is built on the SWE program, and it uses annotations to enable semantic discovery of sensor services.
- A system architecture is presented in (Liefde, 2016), which employs the semantic web technologies to enhance the sensor observation data fusion and aggregation from many sources. Two web processes are presented in this conceptual system architecture: (1) “gathering and harmonizing SOS data connected in a semantic knowledge base”, and (2) “processing observation data by translating logical queries into SOS requests. Both processes use a semantic knowledge base linked to linked sensor metadata for harvesting and harmonizing SOS data”.
- An ontology SmartOntoSensor is presented in (Ali, 2017), which is constructed utilizing NeOn methodology and the CPs pattern. Protégé was used to create SmartOntoSensor, which was then tested utilizing SPARQL, OntoQA, and an experimental investigation. In addition, the ontology is put to the test by incorporating it into the ModeChanger app, which uses SmartOntoSensor to change smartphone modes automatically based on context.

- This article explains how to build a semantic information model that combines information from many sources into a single framework (Calcaterra, 2016). At the sensor level (which is responsible for detecting potentially hazardous phenomena), ontologies have been developed to characterize risks in terms of their potential to cause damage or harm to people and property.

- In a previous paper (Bytyçi, 2017), we developed SEMDPA, a novel lightweight IoT architecture that facilitates connecting sensors and other technologies, and also persons, through a web using the DPA crossing ontology. A Java prototype system uses the OGC's Sensor Observation Service to encode data collected by observed devices. Users sends a query to a website to see the results of the observations, perhaps choosing different filters in the process. The request is encoded and sent to the SOS server as an SOS request. The request is then converted into an SPAQL request that is run against the ontology. The XML output of the SPARQL query is encoded into the OGC's O&M type, which is then shown to the user in HTML table format on the web portal.

- Using a (REST)ful service as a front for Sensor Observation Service is described in (Pschorr, 2013). SOS requests, data collection, and RDF results are all handled by the Semantic Enablement Layer (SEL) when a specific URI is accessed. On-the-fly, the sensor observed stream is transformed to Resource Description Framework. Thus, the data may be interpreted by humans and robots alike.

- A general "Stimulus-Sensor-Observation" ontology design model is proposed in (Janowicz, 2010) for the build of the SSW and the linking of sensor observed stream. Class and connection examples are used to demonstrate each of the primary concepts in the course.

- A SW framework called INWS is developed in the INWATERSENSE project (Ahmedi, 2013). It is based on the SSN model for water monitoring, which models sensor observed data for water quality monitoring in order to simplify classification of water quality based on various regulatory bodies such as the Water Framework Directive.

- It has been proposed that the INWS ontology be used to monitor water quality. INWS sensor data can be analyzed using either a Jess production rule system (Jajaga, 2017a) or C-SWRL (Jajaga, 2017b).

- Consumers who don't adhere to SWE requirements can access data through the use of Linked Sensor Data (Keßle, 2010). Despite the fact that it complicates search,, it nevertheless makes what meta-data describes obvious by allowing annotations with timestamps and locations.

- A model of Semantic Sensor Network for IoT is proposed in (Rezvan, 2015), which can be applied in the processing of data of any kind of sensors. The model uses semantic web and machine learning technologies to transform observed sensor stream to higher-order abstractions that can be recognized by humans and machines.

- Sensor data is modelled using the SSN and SWEET ontologies, which enable a federated request program to be implemented in (Calbimonte, 2011a).

- This is a more comprehensive ontology New notions including communication, data flow, and state are introduced by WSSN to the already existing SSN (Bendadouche, 2012).

3.2 What are the standards that enable the discovery, access, and use of all different sensors and sensor data sources over the Internet? (RQ2)

Sensor observed stream archives are now available or useable through the Web thanks to efforts like SSN-XG[11] and SWE[12]. However, with this newfound flexibility comes a slew of additional obstacles, including the following (W3C, 2010):

- What is the best way to find, search, and access observed sensor stream on the Internet?
- How do you combine observed sensor stream from several sites?
- How can naive users and Web apps make sense of raw sensor stream?

The OGC defines SWE as a specification for the Sensor Web. It is possible to link sensing devices and groups of them to a network of communication using SWE's specifications and infrastructure. Based on the concept of a " Sensor Web", SWE was created with the goal of

[11] https://w3.org/2005/Incubator/ssn/XGR-20110628/
[12] http://opengeospatial.org/ogc/swe

making them available through app interfaces and protocols (Pradilla, 2016), (Echterhoff, 2011), (Botts, 2007). Sensor Web Enablement is separated in two sections:

a. *SWE Information model*

   This model is composed of compression algorithms for theoretical data languages that allow observed WSN data to be visible on the Internet. The following specifications are included in the SWE information model:

   - *Transducer Model Language (TML)*

     - Describes transducers and facilitates actual data streaming between sensor systems and transducers.

   - *Sensor Model Language (SensorML)*

     - Specifies sensors and the associated operations.

   - *Observations & Measurements (O&M)*

     - XML Schemas and standard models provides both archival and real-time storage of sensor observations and measurements.

b. *SWE Service model*

   Through the use of this collection of Web Services, the customer can look for and receive the data they need. Specifications for SWE Service model are as follows:

   - Sensor Planning Service (SPS)

     - A web service interface for user-defined acquisitions and observed WSN data. SPS gives information about a sensor's capabilities as well as how to task it.

   - Sensor Observations Service (SOS)

     - A web service platform for seeking, sorting, and retrieving observations and data from sensor systems. This is the near-real-time connection between an user and an observation archive or a sensor channel.

   - Sensor Alert Service (SAS)

     - Is an interface for web services in sending & receiving sensor alerts. SAS outlines how alarm or alert circumstances are specified, detected, and communicated to users.

   - *Web Notification Services (WNS)*

- A interface to web services for delivering messages asynchronously. WNS supports two types of communication:

a) one-way communication - send a message to a client without waiting for a response.

b) asynchronous two-way communication - deliver the message to the client and wait for a response.

Web-accessible sensors, instruments, and imaging equipment, as depicted in Figure 9, are the primary goal of SWE, as shown in the figure. Web-based sensor networks that are "plug-and-play" are the goal of this effort. When it comes to geospatial standards, the position of a sensor on the Web is often an important consideration (Botts, 2007), (Simonis, 2016).

When sensor data management requires interoperability, the Sensor Observations Service (SOS) standard is ideal (Bröring, 2012). In order to manage sensors that have been deployed and to have access to sensor data, specifically "observation" data, SOS provides an API for that purpose. Today's geospatial systems rely heavily on sensor measurements from both static (such as radar) and in-situ (such as satellite imagery) sensors (Na, 2007). There are several types of deployed sensors (Sn) that can be arranged into constellations (Cn) in Figure 10, and these constellations can be accessed via services, e.g. Sensor Observations Service (Ahn, 2014).

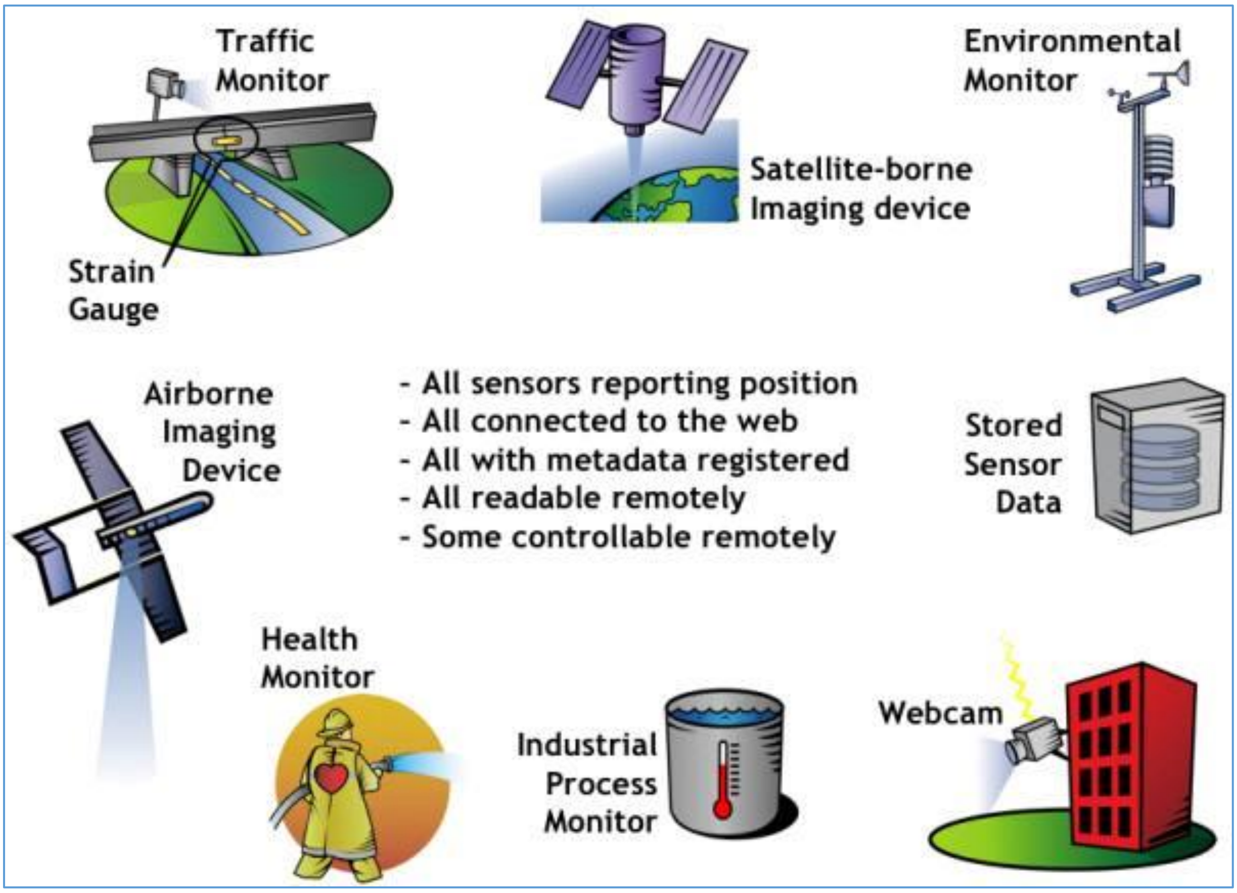


Figure 9. Sensor Web Enablement Concept (Botts, 2007)

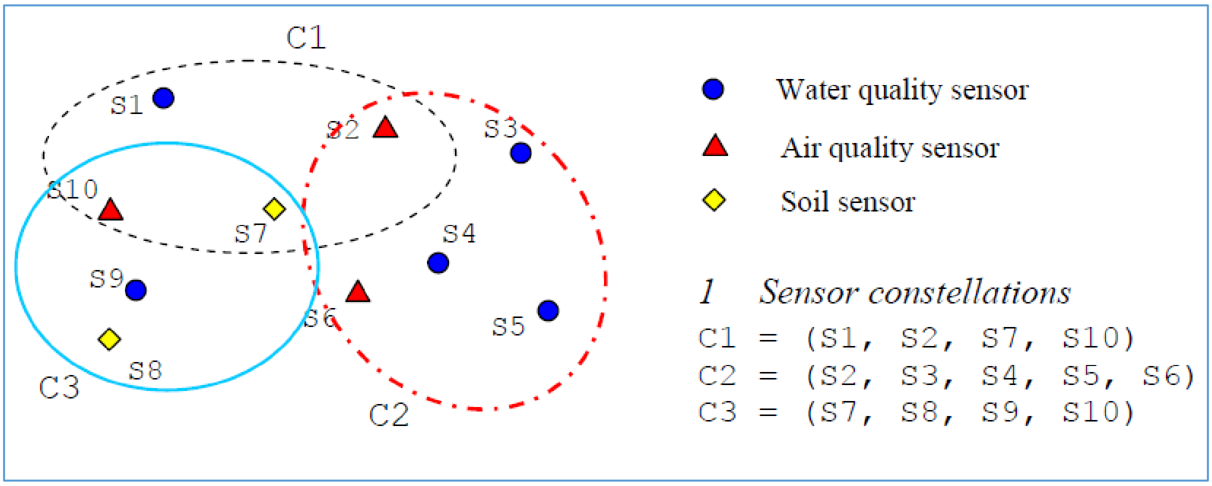


Figure 10. General Case for In-Situ Sensors (Na, 2007)

According to Na (2007), SOS must do three "core" operations:

- *GetObservation* – "data from sensors can be retrieved using a spatial-temporal query that can be filtered by a variety of factors, including phenomena".

- *DescribeSensor* – “allows an SOS server to query metadata about sensors and sensor systems”.
- *GetCapabilities* – “gives you access to metadata and extensive information about an SOS”.

Figure 11 illustrates a GetObservation document request encoded as a Sensor Observation Service query, which contains locating all observations sensed on the sites “Mitrovica” or “Plemetin” for occurrences such as “Electrical conductivity” and “Temperature” using sensors “Sensor1 Temp”, “Sensor2 Cond”, “Sensor3 Temp”, or “Sensor4 Cond” between “2016-01-19 14:00“ and “2016-01-19 14:05“ (Bytyçi, 2017).

In (Cox, 2011) and (Bröring, 2012), are specfied the following attributes of the supplied GetObservation request (or its SOS query):

- temporalFilter – “specifies a time property filter for the requested data”.
- *featureOfInterest* – “identifier for a feature of interest for which observations are sought”.

```
<sos:GetObservation xmlns:sos=... service="SOS" version="2.0.0"...>
  <sos:temporalFilter>
    <fes:During>
      <fes:ValueReference>phenomenonTime</fes:ValueReference>
      <gml:TimePeriod gml:id="tp_1">
        <gml:beginPosition>2016-01-19T14:00:00.000+01:00</gml:beginPosition>
        <gml:endPosition>2016-01-19T14:05:00.000+01:00</gml:endPosition>
      </gml:TimePeriod>
    </fes:During>
  </sos:temporalFilter>
<sos:featureOfInterest>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Plemetin</sos:featureOfInterest>
<sos:featureOfInterest>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Mitrovica</sos:featureOfInterest>
<sos:observedProperty>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Temperature</sos:observedProperty>
<sos:observedProperty>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Conductivity</sos:observedProperty>
<sos:procedure>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Sensor1_Temp</sos:procedure>
<sos:procedure>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Sensor2_Cond</sos:procedure>
<sos:procedure>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Sensor3_Temp</sos:procedure>
<sos:procedure>http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Sensor4_Cond</sos:procedure>
  <sos:responseFormat>http://www.opengis.net/om/2.0</sos:responseFormat>
</sos:GetObservation>
```

Figure 11. Example GetObservation Request as SOS query (Bytyçi, 2017)

```xml
<sos:GetObservationResponse ...>
  <observationData>
    <om:OM_Observation gml:id="o1">
      <om:type xlink:href="http://www.opengis.net/def/observationType/OGC-OM/2.0/OM_Measurement"/>
      <om:phenomenonTime>
        <gml:TimeInstant gml:id="phenomenonTime_1">
          <gml:timePosition>2016-01-19T14:00:00.000+01:00</gml:timePosition>
        </gml:TimeInstant>
      </om:phenomenonTime>
      <om:resultTime xlink:href="#phenomenonTime_1"/>
      <om:procedure xlink:href="http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Sensor1_Temp"/>
      <om:observedProperty xlink:href="http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Temperature"/>
      <om:featureOfInterest xlink:href="http://inwatersense.uni-pr.edu/ontologies/inws-
core.owl#Mitrovica"/>
      <om:result xsi:type="gml:MeasureType" uom="C">15.3</om:result>
    </om:OM_Observation>
  </observationData>
    ...
</sos:GetObservationResponse>
```

Figure 12. Example GetObservation response (Bytyçi, 2017)

- *observedProperty* – a reference to something observed The property for which observations are sough.
- *procedure* – observations are solicited using this technique. For each observation, it describes a filter to apply on the procedure property.
- *responseFormat* – desired responseFormat identifier for the demanded observational data, use this format.

Figure 12 illustrates an example of an O&M Observation from the GetObservation response. Different solutions are available that use these standards to enable all kinds of sensing devices and their observed data archives accessible, discoverable, and usable via the SW, such as (Pradilla, 2016), (Henson, 2009a), (Ikechukwu, 2018), (Pschorr, 2013), (Bytyçi, 2017), (Regueiro, 2017), (Gonzalez, 2017), (Pu, 2016), (Chinnachodteeranun, 2016), (Janowicz, 2010), (Stasch, 2018). Since they have incorprated semantics into OGC SWE standards, several of these methods are covered in section 3.1. As a result, the additional options are listed below:

- In (Pradilla, 2016), An Sensor Observation Service application is proposed, which is suitable for tiny sensor network contexts and does not necessitate particularly robust aplicatoins to work, resulting in a standard and agile program. This application of the Sensor Observation Service enables autonomy from manufacturers and different WSNs by conveying observed sensing data in a standard format and across well-defined apps interfaces.

- In (Haller, 2018), sensors and actuators are described, as well as observations, the processes employed, the subjects and attributes of samples and the sampling process, being seen or acted upon, and so on. SOSA is a stand-alone core which is augmented by SNS and several components to enhance articulacy and diversity. Large-scale scientific monitoring, community science, satellite imagery, social media, industrial and home infrastructures, and the IoT are all possible applications and use cases for the SOSA/SSN ontologies.
- Improved sensor data access through the SWE framework is discussed here (Lee, 2015). This web appliaction provides a visualization of data archived in the SOS by combining free technologies like the API of WEKA.
- OGC SOS interfaces provide a platform for mediating environmental observation data semantically (Regueiro, 2017). Incorporation of sematics in an Sensor Observation Service application is one of the primary features of the suggested method.
- To illustrate OGC SWE's potential to support observation and forecasting, the "Sensor Management for Applied Research Technologies - SMART" project was created (Conover, 2008). Algorithms for anomaly detection rely heavily on the Phenomena Extraction Technique (PEA).
- In (Gonzalez, 2017), an open OGC-compliant AAL approach to provide the AHA is described. SOS is utilized as a major component for collecting and managing sensor data from heterogeneous sensor networks.
- Using a mix of OGC WPS and SOS, a water level monitoring at dam system named TaMIS is introduced in (Stasch, 2018).
- A common web service, SOS API, wraps and exposes a source of climatological data on a grid, AmGSD (Chinnachodteeranun, 2016).
- In (Pu, 2016), a concentrated framework to incorporate different types of sensors into Sensor Observation Service is presented, which allow to exchange and access different environment surveillance sensors.

The Semantic Sensor Network ontology, implemented by the W3C SSN-XG, shows sensors and sensor network resources (Compton, 2012). Observation and measurement data as well as sensor attributes essential to processing are all described in the ontology (Barnaghi, 2012).

3.3. Which models are capable of analyzing data streams in real time and integrating semantics? (RQ3)

There is a series of tuples in the data stream. It's like a record in a database, tuples have attributes. Because greater data rates are more difficult to manage and interpret, the rate at which stream data is received is crucial for processing (Rajaraman, 2014), (Golab, 2003). Sensors of agricultural and water quality monitoring have slow data transfer norms. Since they are able to be saved or archived, their data management and processing differs from those with high data transmission rates.

Data Stream Models - Stream data transmission and saving can be modeled in numerous ways based on characteristics:

- *Real-time data stream*
- *Stream items*
- *Window models*

Each of them is detailed further below.

Real-time Data Stream - is a collection of data which has been arranged in a particular manner that arrive in order and/or in preprocessed ways, resulting in a possible list of models (Golab, 2003):

- *Unordered cash register*: data from multiple sources that do not come in a predetermined order or with pre-processing.
- *Ordered cash register*: Pre-processing is not done on any of the individual data from various domains, but they attain a well-known order in some way.
- *Unordered aggregate*: pre-processed data from the same domain that only one data from the domain comes without being arranged in any way.
- *Ordered aggregate*: pre-processed individual data from the same domain, with only one data arriving in a well-known order.

Stream Items - Data can be represented as a sequence of elements in a list since it is received in a stream, which can be relational rows or object instances (Motwani, 2003). Data are represented as rows in relational based models and saved in virtual relations, however in object based models, types of sources and data are portrayed as hierarchical data types with nearby procedures.

Window Models - In many circumstances, just a part of the sensor streams is of importance at any one time, which encourages the use of window models, which may be characterized using three criteria:

- *Fixed sliding window*: includes only the most recent data or displays only the most recent data based on the timeframe.
- *Landmark window*: a time reference point is established, and data are derived from that period. Because of the increased data quantity within the frame, this criterion is less frequently utilized.
- *Adaptive window*: the window changes dynamically based on the input data and user-specified values.

Streaming data is a continuous and real-time data sequence. Inquiries on stream data are conducted constantly throughout time and gradually yield results as new data comes; thus, these queries are referred to as continuous queries (Chen, 2000).

Many *stream processing engines* are built to deal with stream data. They are primarily concerned with memory stream processing and run continuous queries on streams, such as:

- *Aurora* (Abadi, 2003) is a process system that makes it possible to create query plans through the arrangement of boxes (operators) and arrows (data flow among agents). Aurora is centered on effective planning, high-quality service, and the improvement framework. In the unified processing engine, the system enables continuous inquiries, ad-hoc requests, and sliding windows. Aurora is developed as a distributed streaming paradigm which may be automatically reconfigured as network conditions change.

- *STREAM* (Motwani, 2003) is a general-purpose relational database system that focuses on memory management and approximate query responding. STREAM is a solution that allows you to run continuous queries across numerous continuous data streams.

- TelegraphCQ (Chandrasekaran, 2003) is a query execution system that prioritizes shared query evaluation and adaptive query processing. The fundamental concept would be to analyze accurate inquiry replies for item sets that can be dealt with inside the confines of a given time-frame, in conclusion, features whichever surplus, item sets that the query engine does not have the processing power for capacity. An important part of summarization is the use of processing techniques to reduce the number of inputs to a manageable size.

- Data stream systems like COUGAR (Demers, 2003) use object-relational or object-oriented data models in which sensor nodes are represented by abstract data types (ADTs). For queries that take a long time to complete, COUGAR provides well-defined meanings. A database system's dispersed context was changed to allow for set-oriented execution techniques.

- *NiagaraCQ* (Chen, 2000) is a continuous query system that enables for the execution of continuous XML-QL queries over dynamic Web material.

Other engines rely on a stream processing engine and relational database management systems to work together in harmony (DBMS), for example:

- *Harmonica* (Kitagawa, 2007) is a data stream management system that utilizes an architecture that combines a stream processing engine called StreamSpinner with relational DBMSs. Harmonica satisfies the needs for continuous persistence of streaming data as well as searches over data streams, selecting, joining, projecting, and custom functions.

- *Nile* is a data stream query processing engine. Nile enhances the PREDATOR object-relational database management system's query processor engine to handle continuous queries over data streams. Nile includes improved SQL operators that support sliding-window execution as a way to limit the size of stored information in operators such as join (Hammad, 2004).

Distributed batch processing technologies such as MapReduce (Dean, 2008) and Hadoop (White, 2015) continue to be critical for processing static and historical data collections. There is an increasing demand for stream processing systems, as real-time applications like as intrusion detection and web analytics become more common. Massive streams of dynamic data should be processed on-the-fly by streaming processing systems, and conclusions should be delivered to prospective (potential) customers with the least amount of latency possible (Hanif, 2017).

There are a number of stream processing systems that offer “real-time” and “near-real-time” analytics on data stream, such as (Akidau, 2015), (Zaharia, 2013), (Feng, 2015), (Carbone, 2016), (Toshniwal, 2014):

- Spark Streaming
- Flink
- Storm
- Google Data Flow
- Samza

Modern big data applications rely on distributed data processing platforms to manage both batch and real-time analytics.

These systems are discussed in detail in the following paragraphs:

- A system known as *MapReduce* (Dean, 2008) involves two main steps: Map and Reduce. Batch processing is used to sort and shuffle data.

- The *Hadoop* architecture allows for parallel computing of enormous data volumes over multiple machines using basic algorithms (White, 2015).

- Since its inception, Twitter has been working on a distributed stream data processing system called *Storm* (Toshniwal, 2014).

- It was developed at LinkedIn to address the problem of continuous data processing and is known as *Samza* (Feng, 2015).

- For real-time stream data transformation and enrichment, *Google Data Flow* is a completely - service management (Akidau 2015), (Akidau 2015).

- As a result, *Flink* is a system which can execute both stream and batch analytics (Carbone, 2016).

- For high-performance, fault-tolerant stream processing, *Spark Streaming* is an elongation of the basic Spark API (Ivanov, 2018).

Spark Streaming[13] has been used to compute in real-time the sensor stream in a number of recent research publications, including (Parveen, 2017), (Ge, 2016), (Chen, 2015), (Zaharia, 2013), (Nair, 2018), and (Zhou, 2018). Using powerful methods such as *reduce, window*, *join*, and *map,* stream may be consumed from a variety of origin, counting Kinesis, TCP connections, Flume, and Kafka. When all is said and done, a variety of protocols are available for delivering processed data to filesystems, databases, and even real-time dashboards. Machine learning and graph processing techniques from Spark may be used to analyze data streams as well. Inside, it's a little more complicated. Receives real-time data streams, divides them into batches, and processes them using the ApacheSpark mechanism to produce a final batch of results in batches.

Spark *DStream (Discretized Stream)*, is a powerful abstraction that represents a data stream of *Resilient Distributed Datasets* (*RDDs*) given by the SparkStreaming API (Nabi, 2016), (Spark Apache, 2018). Both Kafka, Flume, and Kinesis-based data streams and high-level operations can be used to build new DStreams (Karau, 2017).

3.4. What are the semantics-based IoT trend domains? (RQ4)

Because of the advancement of the IoT, an increasing number of sensors, motors, as well as mobile devices are being integrated into our everyday life. Consequently, massive amounts of data are generated, and it is imperative that the knowledge concealed inside these massive amounts of data be unearthed. However, sensor and device data from several modes has a wide range of structures, fields, and classes, making it difficult for machines to understand and evaluate the data they produce. As a result, the incorporation of semantics into the Internet of Things has become an overwhelming trend (Shi, 2018). Semantic alerts can be

[13]www.spark.apache.org/docs/latest/streaming-programming-guide.html

used to undertake advanced knowledge discovery and data interpretation, especially in the areas of acknowledging activities, detecting trends, as well as making decisions (Barnaghi, 2012).

Semantic annotation can be found in the following IoT trend domains, according to this study's findings: (but not limited to):

- *Smart Water Monitoring* (Arrieta, 2021), (Jajaga, 2015), (Ahmedi, 2015), (Bytyçi, 2017)
- *Smart Cities* (Ghazal, 2021), (Petrolo, 2017), (Gyrard, 2015), (Djenouri, 2021), (Soldatos, 2015), (Puiu, 2016)
- *Smart Energy Management* (Ploennigs, 2014), (Fensel, 2013)
- *Smart Homes* (Fensel, 2013) , (Vlachostergiou, 2016), (Chen, 2009), (Huang, 2016)
- *Smart Health* (Vannieuwenborg, 2014) (Krummenacher, 2007), (Lee-H, 2015)

The following is a list of other options:

- It is claimed in (Rubi, 2020) that an IoT-based platform for exchanging environmental data in smart cities, focused on semantic web standards and a source of an OWL for environmental indicators, provides interoperability from data collection through knowledge extraction and visualization (Rubi, 2020).

- In (Muppavarapu, 2021), an ontology for the smart home and smart building IoT domains is developed discovering and extracting the most common concepts from sixteen prominent ontologies automatically based on semantic similarities, which reduces the effort required to generate a domain ontology (Muppavarapu, 2021). Experts in the field examine the findings and conclude that they are enough for describing smart homes and intelligent buildings.

- The use of semantics and machine learning to integrate sensor stream data into healthcare platforms is described in (Balakrishna, 2020). A detailed description of the proposed framework's procedures and algorithms is provided, including the collection of raw data, annotation of concepts, extraction of resources data, semantic reasoning, and clustering.

- An application for measurement of water quality in real-time is given by (Sowmya, 2017) based on WSNs. The observed data are continuously transmitted to the coordinator by the wireless sensor nodes. The data is collected at the data center and sent to the database server (Oracle database) where the data is logged in the sensor data table. As a outcome of the data center's connection to the internet, users can access and monitor the data.

- As part of the ongoing work to construct the Museum Energy-Saving Ontology (MESO), we provide in (Zachila, 2021) an ontology connected to energy-efficient cultural settings.

3.5. Systematic review – a summary

Table 3 highlights the findings from the investigated papers and groups them into categories based on their relevance to our research. The table attributes are described below:

- *Reference* –in the APA style, signifies the reference cited (author's surname(s) or organization and the year of the source was published),
- *Type of publication* – Indicates if the work is a conference publication, a journal publication, a report, or something else.
- *Study Type* – determines if the study is practical, theoretical, or evaluative.
- *Domain* – reflects the domain of contribution, such as smart city, water monitoring, and so forth.
- *Research questions (RQ1, RQ2, RQ3, RQ4) - classifies the article according to its relevance to the study's designated research questions.*
- *Main Contributions* – a succinct summary of the article's key contribution.

Table 3. Key studies

Reference	Type of publication	Study Type	Domain	Related to research questions				Main Contributions
				RQ1	RQ2	RQ3	RQ4	
(Sheth, 2008)	Journal	Theory	General	✓	✓	✗	✗	An approach to the sensor web that incorporates semantic web technologies is proposed. The SSW is based on OGC and W3C standards to enhance sensor data descriptions and meaning. RDFa with SOS (O&M-OWL), XLink, and

								SAWSDL are just a few examples of annotation methodologies to consider.
(Henson, 2009a)	Conf.	Applicative	Weather monitoring	✓	✓	✗	✓	The Semantic Sensor Observation Service (SemSOS) is being developed in accordance with the SWE requirements. It adds functionality to 52North SOS, an open-source software developed by the OGC. The SOS (O&M-OWL) annotation approach is utilized.
(Patni, 2011)	Conf.	Applicative	Weather monitoring	✓	✓	✗	✓	An framework to transform and publish the sensor stream data from the weather monitoring domain and to make them publicly available in the cloud, by extracting features from the RDF.
(Elsaleh, 2020)	Journal	Applicative	Smart cities and Smart living	✗	✗	✓	✓	To assist the creation of IoT applications that handle sensor stream data, a novel semantic model for stream annotations is presented, as well as a framework for implementing the model effectively.
(Babitski, 2009)	Conf.	Applicative	Disaster management	✓	✓	✗	✓	SOS: (O&M-OWL) Light-weight semantic annotations are utilized to provide an architecture for the flexible discovery and integration of SOS services.
(Rhayem, 2020)	Journal	Theory	General	✓	✓	✓	✓	A systematic literature review to investigate and examine the approaches to semantic web

								technologies. in the IoT domain is performed.
(Compton, 2009)	Conf.	Theory	General	✓	✓	✗	✗	A thorough literature review is conducted to investigate and analyze a set of the most recent and relevant techniques in the IoT sector that deal with semantic web technologies, such as XLink, and RDFa.
(Honti, 2019)	Journal	Theory	General	✗	✗	✓	✓	An overview of sensor ontologies is offered based on the semantic requirements of the layers of IoT solutions. According to this study, greater standardization is required to allow more flexible connectivity, interoperability, and rapid application-oriented development.
(Lefort, 2009)	Conf.	Theory	General	✓	✓	✗	✗	The article discusses the linking and annotation strategies that can be used to construct geographic mashups services. Four fundamental semantic enablement techniques are identified: a) embedding distant RDF (or OWL) resources in XML via XLink, b) annotating web services with SAWSDL, c) annotating RESTful web services via hRESTs, and d) embedding remote RDF (or OWL) resources in HTML via RDFa.
(Haller, 2018)	Journal	Theory	General	✓	✓	✗	✓	The SOSA (Sensor, Observation, Sampler, and Acuator) lightweight core module is introduced. It is

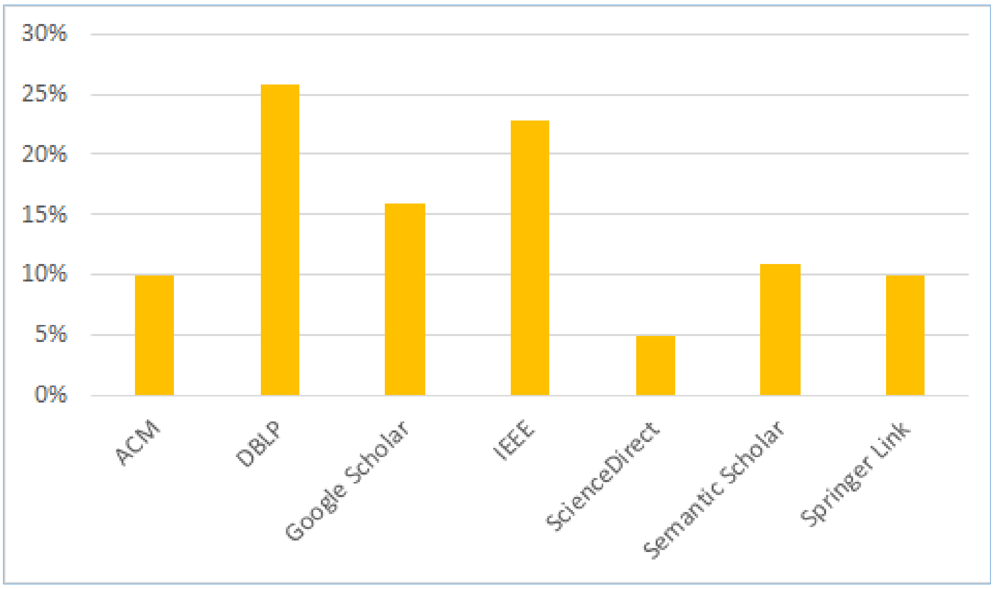
								available at: https://www.w3.org/ns/sosa.
(Chen, 2009)	Journal	Applicative	Smart Homes	✓	✗	✗	✓	Protégé OWL plugin is used to design a conceptual system architecture for semantic smart homes. There are no SWE standards in use at all.
(W3C, 2010)	Report	Theory	General	✓	✓	✗	✗	The fundamentals of annotation are discussed. Some techniques that have been proposed for usage in this context are RDFa with SOS (O&M-OWL), XLink, and SAWSDL.
(Ahmedi, 2013)	Conf.	Applicative	Water Quality Management	✓	✗	✗	✓	For water quality monitoring, the INWS ontology has been proposed using the SSN ontology (Jajaga, 2017a) and (Jajaga, 2017b) show how to evaluate INWS stream sensor data using a Jess production rule system or a Semantic Web rule language C-SWRL.
(Ji, 2014)	Journal	Theory	General	✓	✓	✗	✗	Basic concepts, features, and fundamental technologies of SSW are described in this paper to give an overview of the current status of SSW technology.
(Vera, 2014)	Journal	Applicative	Smart car (smart safety assistance)	✓	✓	✗	✓	OGC Sensor Web services are embedded in an experimental Telco Ubiquitous Sensor Network (USN) Platform. Smart device observations are semantically annotated using SOS (O&M-OWL).

(Khan, 2015)	Conf.	Applicative	Fire monitoring	✓	✗	✗	✓	Virtualized heterogeneous WSNs can be used for semantic applications. SWE has yet to implement any semantic annotation standards or methods.
(Gyrard, 2015)	Conf.	Theory	Smart Cities	✓	✗	✗	✓	It is planned to develop a semantic engine for IoT and smart cities. There is no reference to SWE standards or methodologies for semantic annotation.
(Pu, 2016)	Journal	Applicative	Environmental monitoring	✓	✓	✗	✗	Integration of diverse sensor data into SOS via a centralized framework.
(Pradilla, 2016)	Conf.	Applicative	eHealth	✗	✓	✗	✓	A small sensor network implementation of SOS is proposed that does not necessitate the use of extremely robust equipment, thus providing a standardized and adaptable platform. Semantic web capabilities are not supported by the SOS implementation.'
(Ge, 2016)	Conf.	Applicative	General	✗	✗	✓	✓	Streaming-based processing infrastructure is presented to provide IoT real-time analytics services with high throughput and low latency. An adaptive method for integrating heterogeneous data streams is also discussed in this chapter.
(Osta, 2017)	Conf.	Applicative	Gas Detection	✓	✗	✓	✓	For IoT gateway data annotation, it is suggested that a semantic web method be used. A RDF file is the output of the data annotation module.

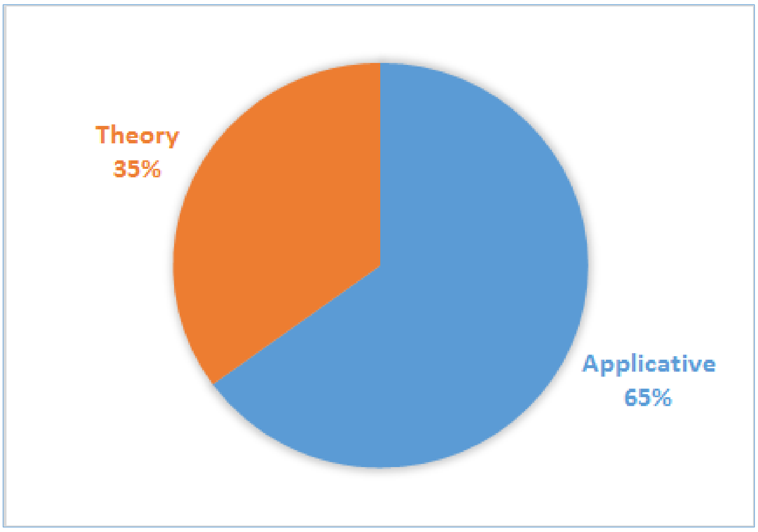
								There is no use of SWE standards.
(Bytyçi, 2017)	Journal	Applicative	Water monitoring	✓	✓	✗	✓	In this paper, a unique lightweight IoT SEMDPA architecture is presented that enables a triangle ontological link between device, person, and activity. New developing technologies, such as context-aware data mining techniques, semantic-enriched sensor web enablement standards, and linked data methodologies, are used to link the previous under the banner of IoT using web semantics as the central architecture.' Since modeling and non-real-time semantic annotation are supported by the SEMDPA framework,
(Rio, 2017)	Journal	Applicative	Smart Oceanographic	✓	✓	✗	✓	For ocean and coastal data management, the "NeXOS" project developed a geographic "sensor web" architecture., and is based on spatial data infrastructure principles and the OGC's SWE framework.
(Parveen, 2017)	Conf.	Applicative	General	✗	✗	✓	✗	A spark streaming-based feature selection technique is presented that can analyze input data in batches and extract the desired feature from a live stream of input data.
(Shi, 2018)	Journal	Theory	Smart Homes, E-Health, & Smart Cities	✓	✓	✗	✓	Data somatization in the Internet of Things is described in generic terms., which

								includes related concepts, generic designs, important methodologies, applications, and obstacles.
(Nair, 2018)	Journal	Applicative	Health monitoring	✗	✗	✓	✓	Predicting health condition in real time is the goal of this project. Open-source big data processing engine Apache Spark is used to build this system. In the cloud, a machine learning model is used to test and build it.

Figure 13 shows the number of papers picked by digital libraries and publication category.



a) digital libraries



b) publication type

Figure 13. Papers chosen based on the following criteria

Part II.

Annotation techniques for real-time integration and interpretation of semantics into sensor stream data

4

Chapter

4. IoT Sensor Data Semantics Integration and Interpretation in Real-Time

Billions of IoT devices are now transmitting continuous streams of sensed data to a central server. The exponential expansion of streaming data has increased the importance and complexity of real-time processing and integration of semantic and sensor data streams. When it comes to IoT, it is critical to identify the optimal strategy for incorporating contextual information into WSNs streaming data and metadata.

4.1. Selected technologies and standards

Spark Streaming[14], Apache Kafka[15], Cassandra database[16], as well as SWE standards are all used in the presented system which enables to real-time incorporate semantic annotation into the WSNs streaming data, that will be explored further in the next sections.

4.1.1. Apache Spark Streaming

Numerous technologies for handling sensor observations have arisen to offer real-time analysis of observations sets, such as Google Data Flow, Apache Spark Streaming, Apache Storm, and Apache Flink (Karimov, 2018). The majority of research find that when the input volume is large, Apache Spark Streaming performed better with high throughput (Gorasiya,

[14] spark.apache.org/streaming
[15] kafka.apache.org/intro
[16] cassandra.apache.org/_/cassandra-basics.html

2019). As a result, we picked Apache Spark Streaming to build our system for real-time integrating semantics into the observed data of different WSN types.

An Apache Spark addon, Spark Streaming, may be used for creating scalable, fault-tolerant IoT systems that analyze sensor stream data. Flume, Kinesis, HDFS, and Twitter are just a few of the many data sources that it can access and handle. A last option is to distribute the streamed data in real time through IoT applications or to databases or file systems. Figure 14 depicts the Spark Streaming methodology.

Figure 14. Spark Streaming workflow

4.1.2. Kafka

Kafka is a framework for the distributed stream-handling similar to a message queue or commercial messaging system that can subscribe as well as publish to streams of data, make it possible to store and process data streams in a fault-tolerant and permanent manner. Data pipelines based on Kafka's real-time streaming capabilities are the most prevalent use cases. In our system, it serves as a bridge in the middle WSNs stream and Apache Spark Streaming.

4.1.3. Cassandra database

As an open-source and free distributed repository platform when data is in a standardized format, Apache Cassandra is ideal for storing mission-critical data that has to be scaled up quickly. Designed to manage a lot of data across many number of servers, while maintaining high scalable and avoiding single points of down. Spark Streaming with this Apache database are a good match. As a result, our system's Cassandra database will house the WSNs observations and integrated semantics handled by Apache SparkStreaming.

4.1.4. OGC standards

With the OGC SOS standard, which is defined in Section 3.2, all sensors, in-situ as well as fixed and mobile, may share their observations in a uniform way that is consistent across all types of sensors. Search results from SOS are returned in Observation and Measurement

(O&M) standard format. These SOS O&M standards will be utilized to encode semantic annotations and WSNs observation.

4.2. An overview of system architecture

Figure 15 illustrates the architecture of an IoT system for real-time integrating/interpreting of semantic annotations into the observed WSN data of different sensor types with context. SparkStreaming, Kafka, Cassandra, as well as O&M standards are used in the developed real-time semantic annotation techniques.

As "producer" for the Kafka server, the IoT-based sensor device wirelessly transmits its heterogeneous sensor stream data. Sending data streams to the Kafka “topics”, which are distributed amongst one or more of the clustered servers (named nodes) referred to as "brokers”, is done via the "producer" client. The data streams from Kafka are subsequently treated in parallel and in real-time using Apache Spark Streaming.

Many types of sensor data streams are received by the Kafka server (e.g., text, binary, JSON, XML, and so on)and turn them into a format that Spark Streaming can process. The real-time detection of outliers will be relayed by the altered sensor data stream, that's alsoperformed inside Spark Streaming. If a data stream object does not behave as predicted, it is classified as an outlier, which could be due to noise or abnormality (Tran, 2016). Outliers can occur for a variety of causes, including mechanical failures, system modifications, fraudulent conduct, technical fault, or human mistake (Koupaie, 2013).

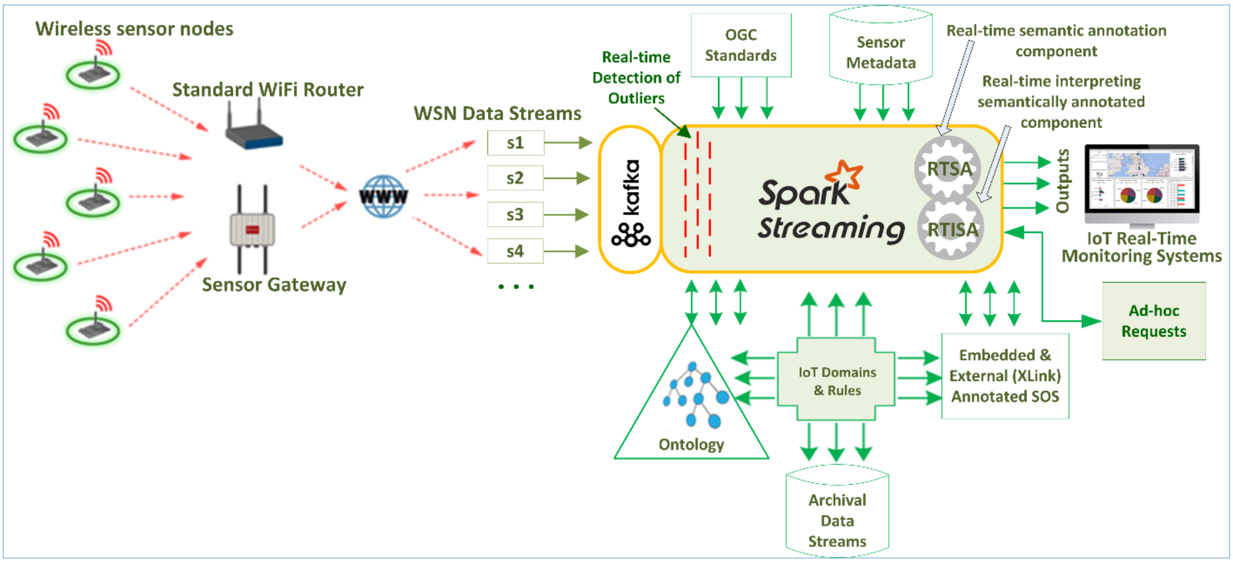


Figure 15. System architecture

WSNs observed data that no marked as outliers and require additional semantic annotation analysis are forwarded to the Spark Streaming component *Real-Time Semantic Annotation (RTSA).* The semantic annotations are subsequently interpreted by another component in Spark Streaming named *Real-Time Interpreting Semantically Annotated (RTISA)*. This section will discuss these two components in further detail:

- *Real-Time Semantic Annotation* – is defined as follows: "*RTSA* enables a real-time integration of semantics into heterogeneous sensor stream data with context in the Internet of Things. RTSA use sensor metadata, archival data streams, and mining data streams for adding se-mantic annotations with concept definitions from ontologies or other semantic sources, which allows the understanding of senor data and metadata elements. The semantic annotations are implemented into SOS O&M by using stakes, such as *External* XML Linking Language (XLink) or *Embedded* to add annotations in XML files. External annotations can point to extra sources of information (e.g. a file), or to Uniform Resource Name (URN), while Embedded annotations are only a single value-scalar of semantic annotation".
- *Real-Time Interpreting Semantically Annotated* – is defined as follows: "*RTISA* enables real-time interpretation of semantics from heterogeneous sensor observation data and sensor metadata with context in the Internet of Things. In other words, it executes and interprets stake an-notated expressions, such as *External* (XLink) or *Embedded*".

The results of the semantic annotations on the enriched WSNs observed data are saved in a database (Cassandra database or relational database) and shown in the real-time IoT applications. It should be noted that Apache SparkStreaming will convert semantic annotations and observed data received by WSNS in the format of SWE standards like SOS (Bröring, 2012). Our system also allows ad-hoc queries, as seen in Figure 15. An ad-hoc query is an inquiry regarding the present condition of a stream or streams that is asked only once.

4.3. Main components and Data modelling

Wireless sensor networks (WSNs) are a critical part of the IoT. They generate a constant observation data in the streaming form, which they send to a central server. The management and use of streaming data has grown increasingly critical as the volume of data has increased dramatically. The inclusion of semantic annotations into the observed data of WSNs is also expected to improve comprehension and describe IoT application areas in a more relevant manner, allowing them to become significantly more intelligent. As a result, figuring out how to include semantic annotations into the observed data of the WSNs and make them machine-interpretable is crucial.

OGC SOS standards can be extended to real-time integrate semantic annotations into the observed data of the different WSN types with context in the Internet of Things.

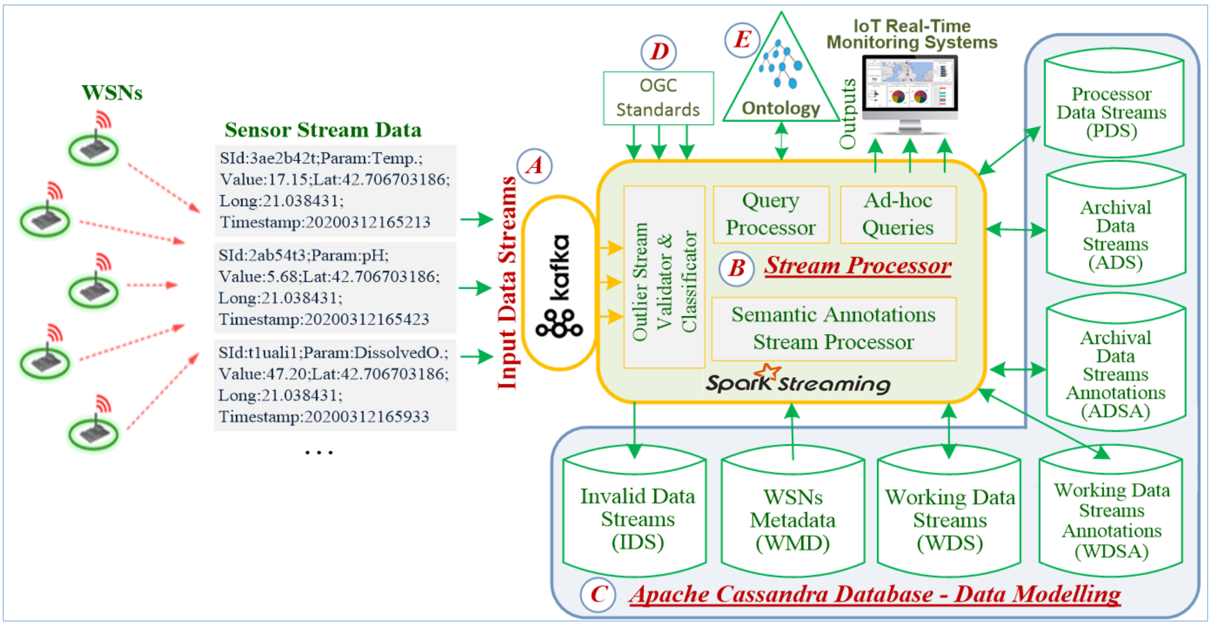


Figure 16. The architecture of the data modeling

Different real-time IoT monitoring applications, like air quality, weather quality, health care monitoring, weather alerts monitoring, etc., can benefit from a model architecture depicted in Figure 16. In a variety of areas, WSNs can be found. A steady stream of data is generated by them, which they send to Kafka in several formats (for example, XML, JSON, text, and others). A special format supported by Spark Streaming's real-time and parallel capabilities is created using Kafka. It is possible to incorporate semantics into observed WSN data using Spark Streaming by incorporating archival sensor observed data, sensor metadata, and IoT domain rules with notion definitions from ontologies or other sources of semantic information, leading to better knowledge and more purposeful descriptions of IoT application fields. As illustrated in Figure 16, the developed model of managing stream data enables "the real-time integrating/interpreting semantic annotations into the observed WSN data, continuous queries on streaming data, outlier validation of streaming data, ad hoc queries, and archive stream data with semantic annotations for applications that require responses from the archival store (persistent data stored)". The primary components of the proposed model are as follows:

A. Input Data Stream,
B. Stream Processor,
C. Data Modelling,
D. OGC standards, and
E. Ontology.

The following is a detailed description of each model component:

A. *Input Data Stream* – it is an Kafka implementation that receives in real-time observed data from sensors in streaming form. As with an *un-ordered cash register*, unordered streams arrive without any form of preparation. This means that each stream can supply items on its own timetable, and they don't have to be of the same kind or have the same data rates.

B. *Stream Processor* – Spark Streaming *Stream Processor* features include *"Outlier Stream Validator and Classificator, Query Process, Ad-hoc Queries*, and *Semantic Annotations"*:

- *Outlier Stream Validator and Classificator* (OSVC) – is an element of the Spark Streaming Stream Processor 's architecture that validates sensor streaming data in real time and assigns one of two statuses to it: *'valid'* or *'outlier*'. Validated data is processed further, whilst in the Invalid Data Streams (IDS), incorrect data is saved. An outlier, also known as a noise or anomaly, is a data stream object that deviates from expected behavior. Because the pH phenomena has a value range of 0 to 14, data showing a pH sensor reading of '-3' or 'NULL' will be considered an oddity. Outliers can occur for a variety of causes, including mechanical failures, system modifications, fraudulent behavior, instrument error, human mistake, or natural variance (Yu, 2020). As a result, quality of the data is provided by the *Outlier Stream Validator & Classificator* for real-time IoT monitoring applications.
- *Query Processor* - searches streaming data in real-time and is refreshed when additional streams enter. The continuous query's result is generated over time, continuously reflecting the observed sensor data viewed thus far. Semantically annotated data can be included in the answers returned by our Query Processor.

- *Ad-hoc Queries* — users' ad hoc requests for information. Prior to this, someone inquired for the present status of a sensor stream data or sensor streams data. Streaming and persistent data can be combined in ad hoc queries on Archival Data Streams, WSNs Metadata, or Working Data Streams can also be specified by users. Additionally, the response of the *Ad-hoc Queries* can contains semantically tagged data.
- *Semantic Annotation Stream Processor* – provides for the real-time incorporation of semantics into the observed data of different WSN types using the Spark Streaming processor. This component can add semantic annotations to observed WSN data using sensor metadata, archive data streams, IoT domain rules (based on ontologies), or other semantic sources. *Working Data Stream Annotations* save the semantic annotations of observed WSN data.

C. ***Data Modelling*** – *Annotations for Processor Data, Working Data, Archival Data, Invalid Data, and WSNs Metadata* are all included in the *Apache Cassandra database's* model of data flow and data storage.

• *Processor Data Streams (PDS)* – overview of streaming data that can be used to answer inquiries for the Stream Processor. Only one row is saved for each deployed sensor, and it contains the following information:

- *WSN Id* - a unique id that distinguishes a WSN from others.
- *WSN parameter* - title of a factor that the WSN measures or observes (for example pressure, wind speed, PM2.5, etc.)
- *WSN Current Value* - the sensor's current reading.
- *WSN Total Rows* - the sensor's total number of observations once it has been put into use.
- *WSN Max Value* - highest value recorded by the WSN since its installation.
- *WSN Min Value* - sensor's lowest reading since it was deployed.
- *WSN Sum Value* - total of the WSN measurements since its installation.
- *WSN Avg Value* - since the WSN's installation, its average value, calculated by dividing the sum of values observed by the WSN by the total number of observed by the WSN (WSN Sum Value / WSN Total Rows).

- *WSN Window Max* - the sliding window's maximum value. The last n data sent by the sensor are displayed in the sliding window, where n is a customizable integer (for example 16 last observed values).
- *WSN Window Min* - sliding window's minimum value.
- *WSN Window Avg* - the sliding window's average value.
- *WSN Current Timestamp* - the sensor's current measured timestamp.
- *Current Latitude & Longitude of WSN* - WSN's current position in terms of longitude & latitude (location from where the WSN sent observed data stream)

• *Working Data Streams (WDS)* – stream Processor operation comprises streaming data, which can be configured by amount and used to answer queries. Consequently, the Fixed Sliding Window displays the most recent WSN observed data (for example 16 last observed values - its customizable number).The following information is recorded for each measured value:

- *Id* – a globally unique identifier (guid) that uniquely identifies a WDS observation.
- *WDS Id* - identification number that distinguishes one sensor from the others.
- WDS parameter - the term used to describe the phenomenon or characteristic being measured by the sensor.
- *Sensed Value* - the WSN transmits a measured value.
- *Timestamp* - the point in time at which the WSN generated the detected value.
- *Latitude & Longitude* - geolocation, or the position of the sensor that transmitted observed data stream. It is particularly useful when a Wireless Sensor Networks is attached to a moving object, like a bus, a car or a plane, or when a mobile WSN monitors a variety of ad-hoc selected locations of interest, whereas when a static sensor monitors a region, the geo place can be "NULL" because the place of these sensor types can be stored as metadata of WSN in *WSNs Metadata* component.
- *WSN Observation Id* - a globally unique identification (guid) for the measurement of a single sensor node. For instance, if a sensor node simultaneously observes three parameters (as a single measurement), all three observations will share the same observation id; else, the observation id would be NULL.
- *Entry Timestamp* - the date and time at which the Stream Processor received the streaming data.

• *Working Data Stream Annotations (WDSA)* – this component stores the sensor streaming data semantic annotations. In order to tag in real-time sensor data streams with semantic annotations, the Semantic Annotation Stream Processor component is used. *Working Data Streams* can extract numerous sematic annotations from a single measurement, including information about the measurement:

- *WSN Annotation Id* - a globally unique identifier (guid) sematic annotations are identified only by this code.
- *WSN Observation Id* - an identifier for the WDS observation.
- *WSN Annotated Date* - the timestamp when sematic annotations were applied to sensor streaming data.
- *WSN Annotated Type* - specifies the annotation type, either "External" (an external resource linked to our ontology'ont-core.owl' via a 'XLink') or "Embedded" (a single value-scalar).
- *WSN Annotated Value* - the semantic annotated value is stored here. An example of 'Embedded' annotation type values for air quality AQI Index:

```
<annotation embedded:AQI_Index ="85"/>
```

  An example of a "External" annotation type value is:

```
<annotation xlink:href="http://myserver/ontologies/ont-
core.owl#Health_Implications_Moderate "/>
```

  (for more details see Figure 21).

• *Archival Data Streams (ADS)* – are data streams that are archived for the purpose of creating reports and statistics. The data modeling structure of *ADS* is identical to that of *WDS*. Information is kept for each measured value as indicated:

- *Id* - a globally unique identifier (guid) which uniquely identify the observation in the *Working Data Streams.*
- *WSN Id* - identification number that distinguishes one sensor from the others.
- *WSN Parameter* - the term used to describe the phenomenon or characteristic being measured by the sensor.
- *Sensed Value* - the WSN transmits a measured value.
- *Timestamp* - the point in time at which the WSN generated the detected value.

- *Latitude & Longitude* - geolocation, or the position of the sensor that transmitted observed data stream. It is particularly useful when a Wireless Sensor Networks is attached to a moving object, like a bus, a car or a plane, or when a mobile WSN monitors a variety of ad-hoc selected locations of interest, whereas when a static sensor monitors a region, the geo place can be “NULL” because the place of these sensor types can be stored as metadata of WSN in *WSNs Metadata* component.
- *WSN Observation Id* - a globally unique identification (guid) for the measurement of a single sensor node. For instance, if a sensor node simultaneously observes three parameters (as a single measurement), all three observations will share the same observation id, else the observation id would be NULL.
- *Entry Timestamp* - the date and time at which the Stream Processor received the streaming data.

• *Archival Data Stream Annotations (ADSA)* – stores semantic annotations of sensor stream data for the purpose of producing reports and statistics. The *ADSA* data modeling structure is identical to the *WDSA* data modeling structure. Several sematic annotations can be obtained from a single measurement in *Archival Data Streams*, including information on the measurement:

- *WSN Annotation Id* - a globally unique identifier (guid) sematic annotations are identified only by this code.
- *WSN Observation Id* - references to ADS WSN Observation Id
- *WSN Annotated Date* - the timestamp when sematic annotations were applied to sensor streaming data.
- *WSN Annotated Type* - specifies the annotation type, either “External” (an external resource linked to our ontology'ont-core.owl' via a 'XLink') or “Embedded” (a single value-scalar).
- *WSN Annotated Value* - the semantically annotated value is stored here. 'High', 'Bad', 'Poor', 'Moderate', or 'Good' are examples of values for water quality status. An example of ‘Embedded' annotation type values for water quality status:

  ```
  <annotation embedded:WaterStatus="Bad"/>
  ```

  Example value of ‘External’ annotation type can be:

```
<annotation xlink:href="http://myserver/ontologies/ont-core.owl#WaterStatus_ClassV"/>
```

(for more details see Figure 26).

• *WSNs Metadata (WMD)* - data that describes WSNs, their equipment, and the site allocation data that goes with them. This information is referred to as static data because it represents the configuration of a WSN, which may include several types of sensor nodes such as sensing nodes, gateway nodes, centralized monitoring nodes, and information on the sensors themselves (including serial numbers, producer, and kind), and also information about the implementation sites, such as sensor location, and so on.

• *Invalid Data Streams (IDS) – OSVC* classifies invalid sensor stream data as an outlier and stores it in Invalid Data Streams (IDS). The data kept in IDS is optional and is determined by the system's needs. The following data is included in IDS:

- *Id* - a globally unique identifier (guid) that identifies each observation in the Invalid Data Streams.
- *WSN Id* - identification number that distinguishes one sensor from the others (assuming it is a valid value).
- *Sensor Parameter* - name of the parameter or phenomenon (e.g. temperature) observed by the sensor (given valid value).
- *WSN parameter* - title of a factor that the WSN measures or observes (for example pressure, wind speed, PM2.5, etc.).
- *Sensed Value* - the WSN transmits a measured value (if is valid value).
- *Timestamp* - the point in time at which the WSN generated the detected value (if is valid value).
- *Latitude & Longitude* - geolocation, or the position of the sensor that transmitted observed data stream (if valid values). It is particularly useful when a Wireless Sensor Networks is attached to a moving object, like a bus, a car or a plane, or when a mobile WSN monitors a variety of ad-hoc selected locations of interest, whereas when a static sensor monitors a region, the geo place can be “NULL” because the place of these sensor types can be stored as metadata of WSN in *WSNs Metadata* component.

- *Entry Timestamp* - the date and time at which the Stream Processor received the streaming data.
- *WSN Observation Id* - A globally unique identifier (guid) that distinguishes a single sensor node measurement (assuming it has a valid value). If a sensor node measures three parameters as a single measurement, the observation id for all three measurements will be the same, unless the observation id is NULL.

D. *OGC Standards* - IoT real-time applications will receive semantically annotated sensor stream data in OGC standards format, namely ver. 2 of the Sensor Observation Service Observation & Measurement standard, as previously indicated.

E. *Ontology* - the "*ont-core.owl*" ontology has been constructed. For example, for the "Air Quality and Weather Alerts Monitoring" domains, semantic annotations like "*#AQI Index*, *#Air Pollution Level*, *#Health Implications*, *#Blizzard*, *#Flurry*, *#Rain Shower*, and *#Rain Storm"* are created (see Figure 31, and Figure 32), while for the Water Quality Monitoring domain, semantic annotations such as *#UNCEF* and *#WFD* are created (see Figure 40).

The following are the details of this model's working cycle: *Input Data Streams* (Apache Kafka) are streams of data sent by wireless sensor networks. As seen below, the observed WSN is an array of different kinds that contains the sensor id (sid), parameter name, sensor measured value, geographical location (latitude and longitude), and timestamp:

```
'SId: 3ae2b42t;
Parameter: Temperature;
Value: 17.15;
Lat: 42.706703186;
Long: 21.038431;
Timestamp: 20200312165213'
```

The stream data elements are then validated using the *Outlier Stream Validator*, which assigns a validity status to each sensor stream data element (*true* - if the data is legitimate, *false* - if the data is outlier). When data is validated as '*true*', it is delivered to the "*Semantic Annotated Stream Processor"* for additional processing, that allows for real-time integrating semantics into the observed WSN data. Then, the enriched observed WSN data will be translated into SOS Observation & Measurement standard for display in the real-time IoT applications, and the results of the semantic annotations is going to be preserved in WDSA & WDS.

When the sensor detects a new value, it sends it to *Working Data Streams* (where their semantic annotations are recorded in *Working Data Stream Annotations*), and the oldest value is deleted from *Working Data Streams* and sent to *Archival Data Streams* (or *Archival Data Stream Annotations*) for archiving. As a result, data in *Archival Data Streams* and *Archival Data Stream Annotations* has been archived and can be utilized to construct reports and statistics over longer periods of time.

Part III.

Implementation and Testing

5
Chapter

5. Development and Implementation of the System

To test the introduced techniques and model for real-time integration and interpretation of semantic annotations into the different types of observed WSN data and WSN metadata with context in the Internet of Things, a prototype system called “IoT Semantic Annotations System (IoTSAS)” has been developed, as shown in Figure 17. The system is divided into modules, which are as follows:

1. The real-time integrating and interpreting of semantic annotations into the observed WSN data module,
2. Data modelling module (see section 4.3),
3. The module for managing metadata,
4. Monitoring module for air quality,
5. Monitoring module for weather warnings,
6. Water quality monitoring module,
7. The module for external systems - RESTful APIs.

The IoTSAS system's real-time processing capabilities include *continuous input* of observed data from different WSNs types, *processing* of the monitoring systems with low processing latency requirements can benefit from semantically annotated and interpreted data, as well as data *produced* in the SOS O&M format.

The real-time integrating and interpreting of semantic annotations into the observed WSN data is the primary module build in Apache Spark Streaming. There are three languages that can be used for Spark Streaming applications: Java (the default), Python, and Scala. The module is built using Eclipse's Java programming. Figure 18.a shown the Java packages for this module:

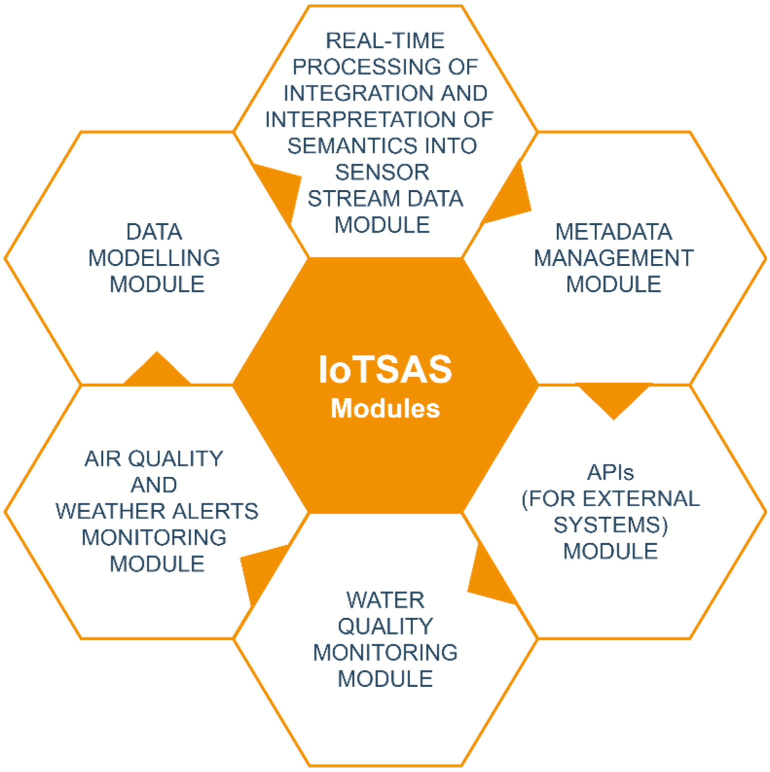


Figure 17. Modules for the IoTSAS System

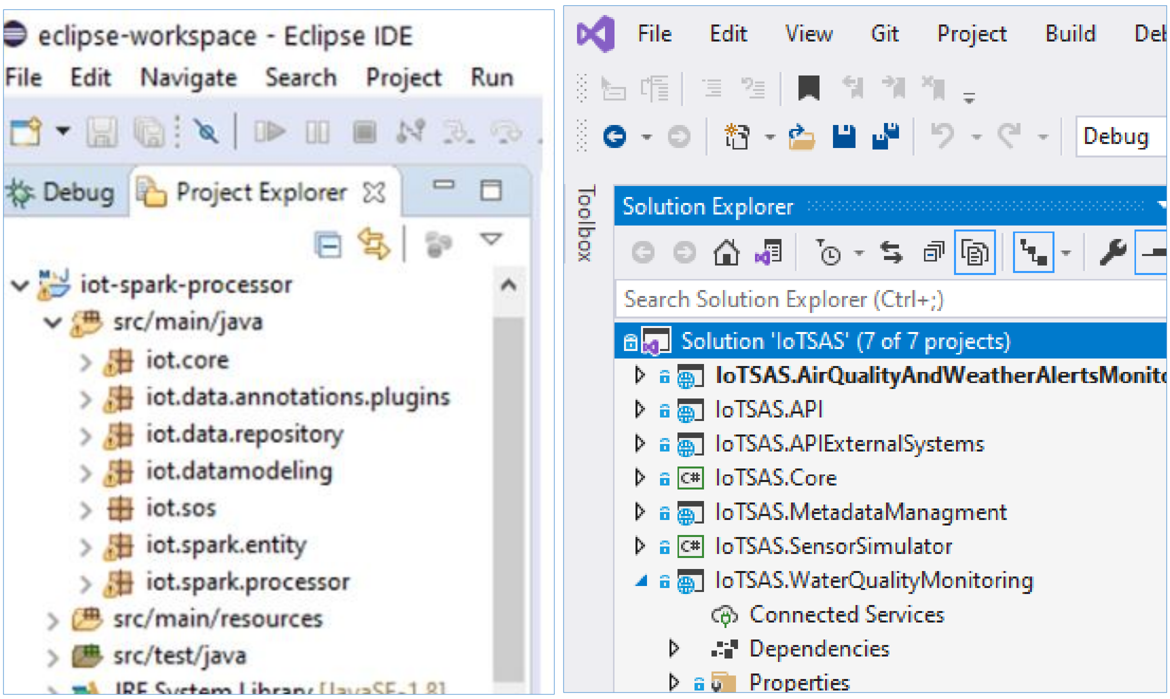


Figure 18. Workspaces for IoTSAS solutions: a) Core module of IoTSAS Java packages, b) .Net C# projects for IoTSAS other modules

- *iot.core*
    - Real Time Outlier Detection, IoT Data Stream Decoder, Input IoT Data Stream, Query Processor, and IoT Domain.
- *iot.data.repository*
    - Processor IoT Data Stream Repository, Cassandra Utils, Archival IoT Data Stream Annotation Repository, Cassandra Connector, Working IoT Data Stream Repository, Working IoT Data Stream Annotation Repository, and Archival IoT Data Stream Repository.
- *iot.data.annotations.plugins*
    - Water Quality Annotations, Air Quality Annotations, and Weather Alert Annotations.
- *iot.sos*
    - OGC SOS Standards (Transform to O&M Observation), and Get Observation Response.
- *iot.datamodeling*
    - Create Working IoT Data Streams Model, Create Keyspace, Data Modeling, Create Processor IoT Data Streams Model, Create Archival IoT Data Streams Model, Create Invalid IoT Data Stream, Create Archival IoT Data Stream Annotations Model, and Create Working IoT Data Stream Annotations Model.
- *iot.spark.processor*
    - RTISAE Engine, RTSAE Engine, and IoT Spark Processor.
- *iot.spark.entity*
    - IoT Sensor, Working IoT Data Stream, Processor IoT Data Stream, Parameters, IoT Annotation, Ontology Source, IoT Sensing Node Device, Archival IoT Data Stream, IoT Data Stream, Sensing Node Device, Archival IoT Data Stream Annotation, Invalid IoT Data Stream, Sensing Node, Working IoT Data Stream Annotation, and Ontology Classes.

Other modules are developed in .Net Core C#, depending on performance (Dhalla, 2020) and our extensive knowledge with .Net C# technology. As seen in Figure 18.b, the.NET C# has the following solutions:

- IoTSAS – Core

- IoTSAS – Metadata Managment
- IoTSAS – API
- IoTSAS – Sensor Simulator
- IoTSAS – Air Quality And Weather Alerts Monitoring
- IoTSAS – Water Quality Monitoring
- IoTSAS – API External Systems

Each of the modules is detailed in detail in the following paragraphs.

5.1. The real-time integrating and interpreting of semantic annotations into the observed WSN data module

The real-time processing module delivers the system's functionality by integrating and interpreting semantics into the observed WSN data. The Spark Streaming, Kafka, Casandra DB, and OGC Observations & Measurements standards, all of which are mentioned in Section 4.2 of the system design, are all utilized.

The figure 19 depicts a high-level look of the IoTSAS's process architecture. The data collected by the heterogeneous WSNs is transmitted to Kafka in a various formats. A Kafka Producer is applied in Kafa that reads various formats of sensor data, converts it to an appropriate type, and publishes it to Kafka topics. The name of a Kafka topic is a global attribute of the Kafka cluster namespace. Kafka topics are a set of messages published by one or more Kafka producers and consumed by one or more Kafka consumers in a queue or logical order. Kafka transforms all messages into byte arrays. TCP is utilized in Kafka to communicate between producers, consumers, and clusters. A Kafka broker is composed of one or more topics, each of which is further subdivided into a single or several partitions.

The changed observed WSN data is routed over Kafka cluster to the Spark Streaming for additional processing. Apache Spark Streaming separates the observed WSN data into 50 millisecond intervals known as Discretized Streams (*DStreams*), consisting of RDDs, one for each batch interval, that serve as a foundation for the entire system. The observed WSN data collected throughout the batch interval is stored in each RDD. The observed WSN data included in RDD is partitioned, and operations on the data cached in memory are done in parallel, enabling great performance at scale while minimizing disk input/output (read/write). The *filter* function is used to remove outliers from the RDD observed WSN data. The RDD observed WSN data are then converted to “WorkingIoTDataStream” using the *transform* function by appending an unique ID (Universally Unique Identifier – UUID) that uniquely identifies the observed WSN data, and an entry timestamp that indicates when the observed WSN data arrived at the Spark *Stream Processor*. This is followed by the mapping of RDDs to IoT domains (for example, monitoring air quality, weather warnings, or water quality), and then utilizing built plugins via the RTSA component, the RDD observed WSN stream is enhanced with semantic annotations from an OWL source and then executed and interpreted stake annotated expressions using the RTISA component. Finally, semantically annotated RDDs are translated to OGC Observation and Measurement standards via the *transform* function, and used by IoT applications for real-time monitoring and then are saved to Cassandra DB. Figure 19 presents the overall process.

As illustrated and detailed in Table 4, a new form of observation called *"SemObservation"* with the result type *"gml:SemMeasureType"* has been developed. The OGC Observation and Measurement format without semantic annotations is displayed in Figure 20, while the OGC Observation and Measurement format with semantic annotations, including the developed type "*SemMeasureType"*, is shown in Figure 21.

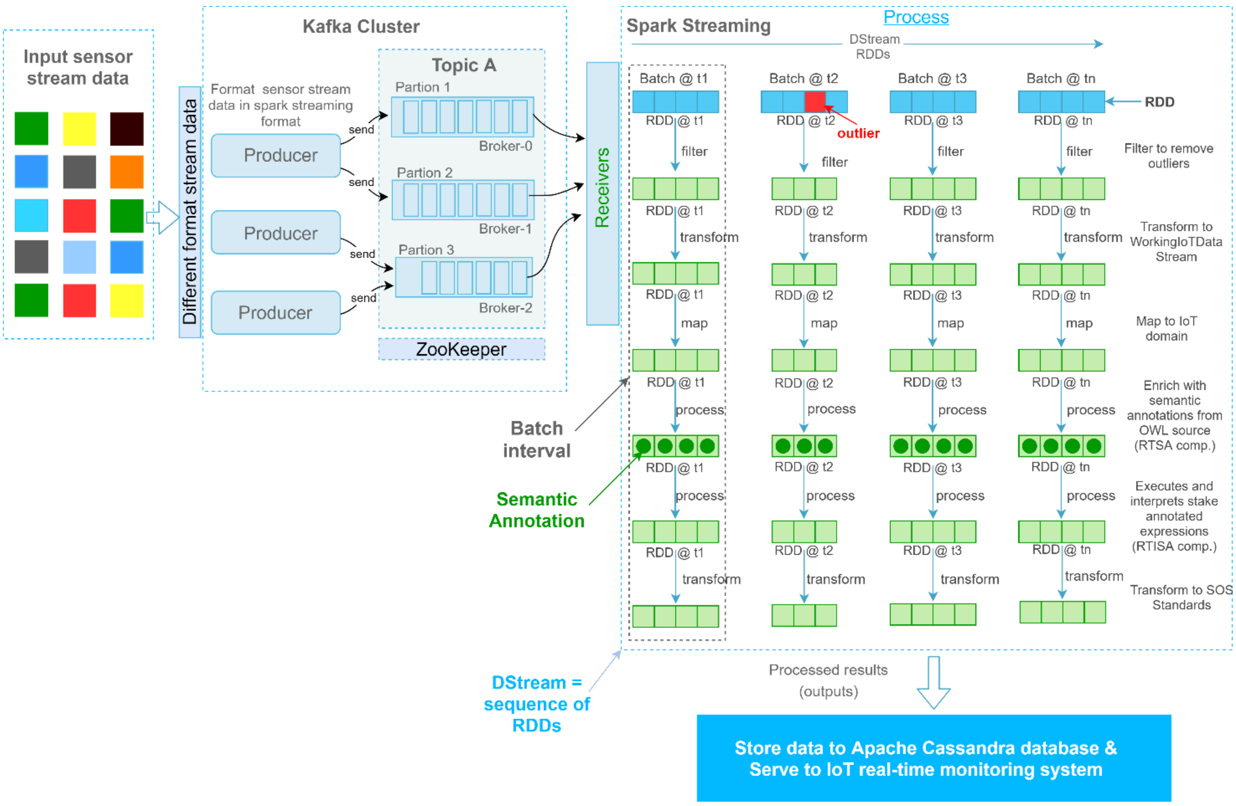


Figure 19. Real-time integrating and interpreting of semantic annotations into the observed WSN data

Table 4. SemObservation – the observation type that have been developed

Observation Type	Result Type	Description	Example
SemObservation	*gml:SemMeasureType*	Inside the result element, three children elements will be added: *value*, *sem-annotations, and sem-intpretations*. The *value* element will contain a scalar numerical value, the *sem-annotations* element will contain one or more annotation empty elements, while the *sem-interpretations* element will contain the textual interpretations.	*<om:result xsi:type="gml:SemMeasureType" uom="pm25"> <value>58</value> <sem-annotations> <annotation xlink:href="http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Moderate"/> <annotation embedded:AQI_Index ="58"/> <annotation xlink:href="http://myserver/ontologies/ont-core.owl#Health_Implications_Moderate"/> </sem-annotations> <sem-interpretations> Textual interpretatinos.. </sem-interpretations> </om:result>*

Observations in OGC Observation and Measurement can have a single instance of the observation or several instances of the *observationData* element. The following characteristics are frequently observed (the prefix "om" denotes that it is specified in OGC 10-025r1, whereas the prefix "gml" denotes that this element is specified in OGC 07-033) (Jirka, 2014):

- *"gml:identifier"* (required): this is used to identify or refer to a particular observed WSN data. In our example, the observation is identified by a produced UUID (e.g. 79c22ab1-g390-b734-7b44-8b8b6df9818).
- *"om:phenomenonTime"* (required): indicates the timestamp when the observed WSN data was taken.
- *"om:resultTime"* (required): shows the date and time that the result was produced (frequently this is the same as the "*om:phenomenonTime*").
- *"om:procedure"* (required): is the identification of the WSN for the observation.
- *"om:observedProperty"* (required): the parameters observed in the Internet of Things domain.

- *“om:featureOfInterest”* (required): the feature of interest identification (e.g., the station of WSN) to which the observed WSN data is related.
- *“om:result”* (required): the WSN observation value (result); the result type must be one of the following: *gml:MeasureType, gml:ReferenceType, xs:boolean, xs:string, xs:integer, swe:DataArray*, or *swe:DataRecordPropertyType* are all acceptable values.

Additionally, a complex type OGC Observation & Measurement is implemented in our solution. Figure 22 illustrates the basic complex type OGC Observation & Measurement standard format, that we developed by incorporating two additional features (elements):

- *“swe:sem-annotations”* – includes 1 or more empty annotation elements that can be “*Xlink*” or “*Embedded*”. The RTSA component produces the annotation elements.
- *“swe:sem-interpretations”* – sensor's observed data is interpreted in this element. The component RTISA produces the interpreted data.

```
<sos:Observation ...="">
<observationData>
  <om:OM_Observation gml:id="o1">
    <om:type xlink:href="http://www.opengis.net/def/observationType/OGC-
OM/2.0/OM_Measurement"/>
    <om:phenomenonTime>
      <gml:TimeInstant gml:id="phenomenonTime_1">
        <gml:timePosition>2020-03-25T20:00:00+09:00</gml:timePosition>
      </gml:TimeInstant>
    </om:phenomenonTime>
    <om:resultTime xlink:href="#phenomenonTime_1"/>
    <om:procedure xlink:href="http://myserver/ontologies/ont-core.owl#SensorId"/>
    <om:observedProperty xlink:href="http://myserver/ontologies/ont-
core.owl#parameterName"/>
    <om:featureOfInterest xlink:href="http://myserver/ontologies/ont-core.owl#Location"/>
    <om:result xsi:type="gml:MeasureType" uom="parameterUnit">
      parameterValue
    </om:result>
  </om:OM_Observation>
</observationData>
. . .
</sos:Observation>
```

Figure 20. OGC Observation & Measurement standard document without semantic annotations.

```
<sos:Observation ...="">
<observationData>
  <om:OM_Observation gml:id="o1">
    <om:type xlink:href="http://www.opengis.net/def/observationType/OGC-
OM/2.0/OM_Measurement"/>
    <om:phenomenonTime>
      <gml:TimeInstant gml:id="phenomenonTime_1">
        <gml:timePosition>2020-03-25T20:00:00+09:00</gml:timePosition>
      </gml:TimeInstant>
    </om:phenomenonTime>
    <om:resultTime xlink:href="#phenomenonTime_1"/>
    <om:procedure xlink:href="http://myserver/ontologies/ont-core.owl#SensorId"/>
    <om:observedProperty xlink:href="http://myserver/ontologies/ont-
core.owl#parameterName"/>
    <om:featureOfInterest xlink:href="http://myserver/ontologies/ont-core.owl#Location"/>
    <om:result xsi:type="gml:SemMeasureType" uom="parameterUnit">
      <value>parameterValue</value>
      <sem-annotations>
        <annotation xlink:href="http://myserver/ontologies/ont-
core.owl#semanticAnnotation1"/>
        <annotation embedded:semanticAnnotation2 ="value"/>
        <annotation xlink:href="http://myserver/ontologies/ont-core.owl#
semanticAnnotation3"/>
      </sem-annotations>
      <sem-interpretations>
         Textual interpretatinos..
     </sem-interpretations>
    </om:result>
  </om:OM_Observation>
</observationData>
. . .
</sos:Observation>
```

Figure 21. OGC Observation & Measurement standard document with semantic annotations.

```xml
<sos:Observations>
  <om:OM_Observation gml:id="o1" ...
    <gml:description>Complex Observation instance</gml:description>
    <gml:name>Complex Observation</gml:name>
    <om:type xlink:href="http://www.opengis.net/def/observationType/OGC-
OM/2.0/OM_ComplexObservation"/>
    <om:phenomenonTime>
      <gml:TimeInstant
        gml:id="ot1t">
        <gml:timePosition>2020-03-25T20:00:00+09:00</gml:timePosition>
      </gml:TimeInstant>
    </om:phenomenonTime>
    <om:resultTime xlink:href="#ot1t"/>
    <om:procedure xlink:href="http://localhost/ontologies/ont-core.owl#SensingNode"/>
    <om:observedProperty xlink:href="http://localhost/ontologies/ont-
core.owl#IoTDomain"/>
    <om:featureOfInterest xlink:href="http://localhost/ontologies/ont-
core.owl#Location"/>
    <om:result xsi:type="swe:DataRecordPropertyType">
      <swe:DataRecord>
        <swe:field name="parameterName1">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#SensorId1">
            <swe:uom code="parameterUnit1"/>
            <swe:value>parameterValue1</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="parameterName2">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#SensorId2">
            <swe:uom code="parameterUnit2"/>
            <swe:value>parameterValue2</swe:value>
          </swe:Quantity>
        </swe:field>
        ...
      </swe:DataRecord>
      <swe:sem-annotations>
        <swe:annotation xlink:href="http://myserver/ontologies/ont-
core.owl#semanticAnnotation1"/>
        <swe:annotation embedded:semanticAnnotation2 ="value"/>
        <swe:annotation xlink:href="http://myserver/ontologies/ont-core.owl#
semanticAnnotation3"/>
      </swe:sem-annotations>
      <swe:sem-interpretations>
        Textual interpretatinos..
      </swe:sem-interpretations>
    </om:result>
  </om:OM_Observation>
</sos:Observations>
```

Figure 22. Complex OGC Observation & Measurement standard document with semantic annotations & interpretation

5.2. Data modelling module

The components of the data modeling are built in the Casandra DB, which include: "*Processor Data Streams* (stores a summary data of each sensor for Stream Processor operations), *Working Data Streams* (a fixed sliding window that stores 15 last measured values for each sensor), *Working Data Stream Annotations* (stores semantic annotations of Working Data Streams observations data), *Archival Data Streams* (archives sensor stream data for generating reports and different statistics), *Archival Data Stream Annotations* (archives semantic annotations of sensor stream data), *Invalid Data Streams* (stores invalid sensor stream data that are classified as outlier), and *WSNs Metadata* (known as static data that store data and metadata about sensors, sensors types, sensing nodes, gateway nodes, central monitoring nodes, etc.)". The data processing cycle is detailed in Section 4.3.

Figure 23 illustrates the data modeling diagram for the proposed data stream management architecture. Each class is defined by its own set of properties, methods, and events.

To illustrate how data is saved in the Cassandra DB, the following graphics are included: Figure 24 depicts *Processor Data Streams*; Figure 25 depicts *Archival Data Streams*; and Figure 26 depicts *Archival Data Stream Annotations*.

5.3. The module for managing IoT metadata

The module for managing IoT metadata allows for the administration of data known as static data, which comprises the following:

A. Devices meta data,
B. Nodes meta data, and
C. Phenomenon (parameters) meta data.

A. Device meta data – includes information for the various sorts of devices, such as sensors, microcontrollers, servers, clusters, and cables, as well as information about individual *devices*, such as device name, a description of the device, the SN of the device, the sensor code (if the equipment is a WSN), the status of the device (passive/active), the producer, and the phenomenon measured by the WSN, for example PM2.5 ($\mu g/m^3$), O3 (ppb), PM10 ($\mu g/m^3$), Humidity (percent), CO (ppm), SO2 (ppb), Pressure (mb), NO2 (ppb), etc.

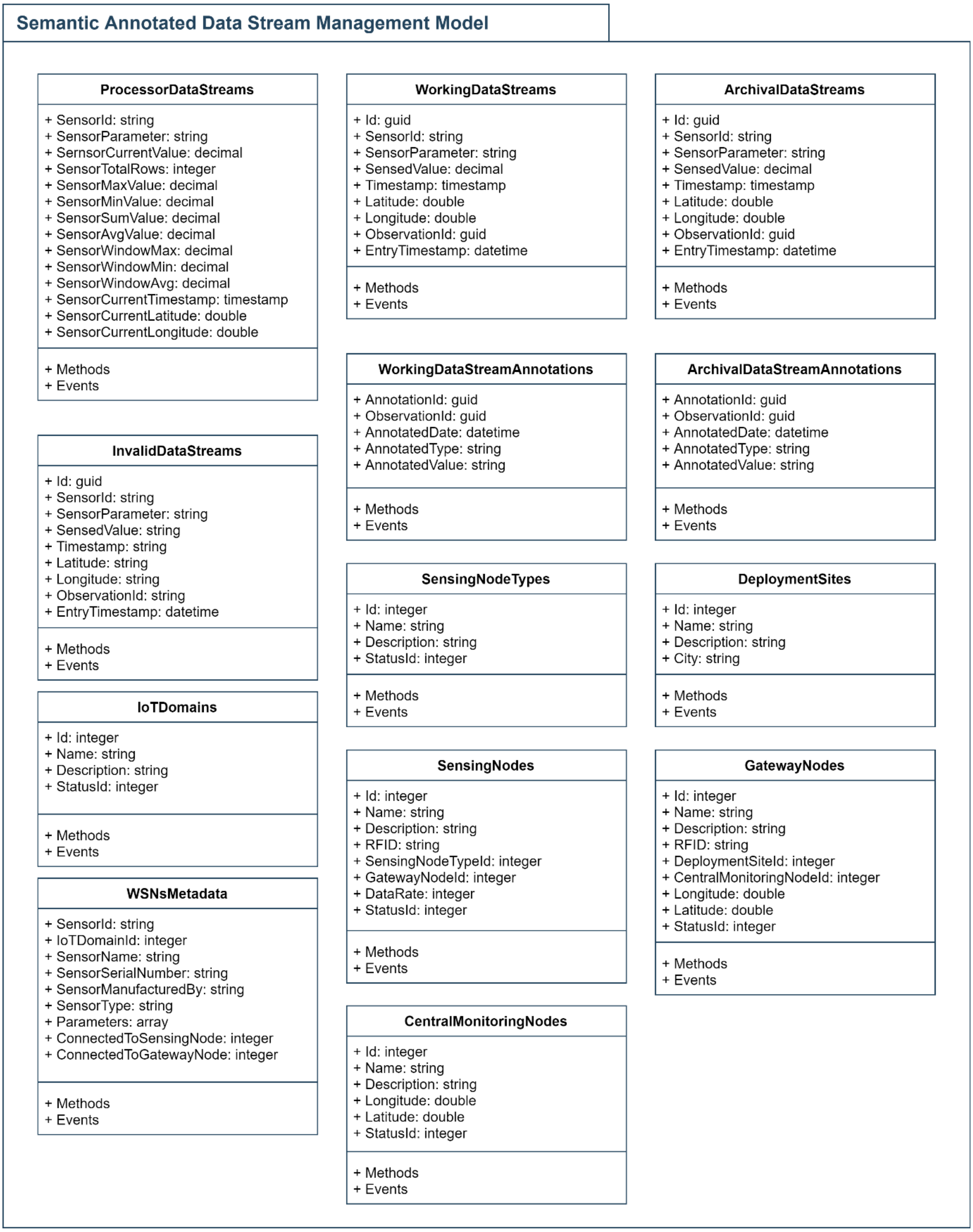


Figure 23. Data Modeling Diagram

Cassandra 3.11.4 : IoTSAS.Cassandra.Connection : iot : processoriotdatastreams

sensorid	all_avg	all_max	all_min	all_sum	lastentrydate	lastlatitude	lastlongitude	lastobservationid	lasttimestamp	lastvalue	parameter	totalrow	win_avg	win_max	win_min	win_row	win_sum
S0026	6.9	6.9	6.9	627.899999999999	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	6.9	NO2	91	4.6	6.9	6.9	3	13.8
S0001	1.1	13.1	1.1	619.900000000001	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	1.1	CO	93	0.7333333...	1.1	1.1	3	2.2
S0037	23.7	98030.78	23.7	316202.43	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	197.77	Visibility	33	95.29	197.77	88.1	3	285.87
S0008	0.6	11.9	0.6	794.499999999998	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	0.8	SO2	93	0.5333333...	0.8	0.8	3	1.6
S0030	38	109	38	8574	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	55	PM2.5	91	36.666666...	55	55	3	110
S0023	0.8	14.4	0.8	816.3	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	9.6	WG	96	6.4	9.6	9.6	3	19.2
S0006	6	38	6	1332	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	6	PM10	93	4	6	6	3	12
S0036	45.92	96895.87	45.92	291484.37	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	296.57	Visibility	36	212.12333...	339.8	296.57	3	636.37
S0015	8	38.5	8	1789.3	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	38.1	O3	97	25.4	38.1	38.1	3	76.2
S0018	24	99	24	6823	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	24	PM2.5	96	16	24	24	3	48
S0005	1010	1020	1010	94467	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	1015	Pressure	93	676.66666...	1015	1015	3	2030
S0028	1007.4	1016	1007.4	92279.2	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	1013.3	Pressure	91	675.53333...	1013.3	1013.3	3	2026.6
S0039	60	70	60	2336	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	70	Precipitation	36	46.666666...	70	70	3	140
S0011	9	9	9	837	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	9	WG	93	6	9	9	3	18
S0035	6.78	80070.67	6.78	293147.48	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	26.54	Visibility	31	95.45	259.81	26.54	3	286.35
S0024	15	15	15	1365	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	15	CO	91	10	15	15	3	30
S0021	1.5	23.7	1.5	524.8	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	15.9	Temp.	96	10.6	15.9	15.9	3	31.8
S0032	1.5	22.3	1.5	532.4	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	16.1	Temp.	91	10.733333...	16.1	16.1	3	32.2
S0013	40.8	77	40.8	6601.6	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	49.8	Humidity	97	33.2	49.8	49.8	3	99.6
S0007	15	82	15	3496	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	15	PM2.5	93	10	15	15	3	30
S0014	3.1	20.6	3.1	1238.3	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	3.1	NO2	97	2.0666666...	3.1	3.1	3	6.2
S0010	1.5	399.35	1.5	2630.25	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	399.35	Wind	93	241.49	399.35	325.12	3	724.47
S0017	14	14	14	1344	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	14	PM10	96	9.3333333...	14	14	3	28
S0029	14	14	14	1274	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	14	PM10	91	9.3333333...	14	14	3	28
S0033	1	386.46	1	2038.22	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	386.46	Wind	91	145.29333...	386.46	49.42	3	435.88
S0027	32.8	32.8	32.8	2984.8	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	32.8	O3	91	21.866666...	32.8	32.8	3	65.6
S0003	1.1	15	1.1	608.9	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	1.1	NO2	93	0.7333333...	1.1	1.1	3	2.2
S0034	1	10.5	1	661.1	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	10.5	WG	91	7	10.5	10.5	3	21
S0038	40	90	40	2020	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	70	Precipitation	31	46.666666...	70	70	3	140
S0019	1.7	6.7	1.7	356.3	2021-05-06 09:00:00	42.648872	21.137121	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 09:58:24	1.7	SO2	96	1.1333333...	1.7	1.7	3	3.4
S0031	2.4	2.4	2.4	218.4	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	2.4	SO2	91	1.6	2.4	2.4	3	4.8
S0009	1.5	24	1.5	458.5	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	17	Temp.	93	11.333333...	17	17	3	34
S0025	26.2	77	26.2	6237.19999999999	2021-05-06 09:00:00	42.661995	21.15055	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 09:58:27	50	Humidity	91	33.333333...	50	50	3	100
S0040	0	90	0	2079	2021-05-06 09:00:00	42.625349	20.891036	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 09:58:26	70	Precipitation	33	46.666666...	70	70	3	140

Figure 24. Data of the ProcessorDataStreams

Cassandra 3.11.4 : IoTSAS.Cassandra.Connection : iot : archivaliotdatastream

SQL Query

sensorid	entrydate	id	latitude	longitude	observationid	parameter	timestamp	value
S0026	2021-05-06 21:56:14	b2cebf28-beec-46d4-afcd-b4440c80fcc2	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	NO2	2021-05-06 21:00:00	6.9
S0001	2021-05-06 21:56:13	321f0a84-1c3d-4171-9776-297e44f965b4	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	CO	2021-05-06 21:00:00	1.1
S0037	2021-05-06 21:56:13	1ef94a25-5dd6-420c-b840-6afcd93ec932	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	Visibility	2021-05-06 21:00:00	88.1
S0008	2021-05-06 21:56:13	13bdceb7-40f7-4e5e-a4fa-fc4954ffe99f	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	SO2	2021-05-06 21:00:00	0.8
S0030	2021-05-06 21:56:14	9a2ca29c-72e3-472e-87ef-2f325f8db31d	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	PM2.5	2021-05-06 21:00:00	55
S0023	2021-05-06 21:56:09	6fda26a0-6e07-4668-a1e3-daadc91fbea6	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	WG	2021-05-06 21:00:00	9.6
S0006	2021-05-06 21:56:13	db71809c-a7e3-4f84-a828-ce41fd58d817	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	PM10	2021-05-06 21:00:00	6
S0036	2021-05-06 21:56:09	1f0570e1-b59f-4f3b-9a9d-d2e01ff4a845	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	Visibility	2021-05-06 21:00:00	339.8
S0015	2021-05-06 21:56:09	12441dd7-73c3-4000-955b-6bec4fdc3d4a	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	O3	2021-05-06 21:00:00	38.1
S0018	2021-05-06 21:56:09	a5003a79-cb02-4293-99ce-a72e9e7b0f5d	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	PM2.5	2021-05-06 21:00:00	24
S0005	2021-05-06 21:56:13	2c310603-1f52-432c-af00-856fcb34a62b	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	Pressure	2021-05-06 21:00:00	1015
S0028	2021-05-06 21:56:14	1762d703-4cd2-4b98-902c-49139a7c84cf	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	Pressure	2021-05-06 21:00:00	1013.3
S0039	2021-05-06 21:56:09	916f05b4-3d0a-4190-96d4-820689ef6a0c	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	Precipitation	2021-05-06 21:00:00	70
S0011	2021-05-06 21:56:13	ca1c4b03-e778-46f7-9ba5-22be1ae8bed1	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	WG	2021-05-06 21:00:00	9
S0035	2021-05-06 21:56:14	f66cf328-1078-4a9a-bd11-57424664264f	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	Visibility	2021-05-06 21:00:00	259.81
S0024	2021-05-06 21:56:14	e74f0340-3f45-4c61-a778-4c80b08e0992	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	CO	2021-05-06 21:00:00	15
S0021	2021-05-06 21:56:09	aab58df2-4b16-4dd1-9889-11264d0265c7	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	Temp.	2021-05-06 21:00:00	15.9
S0032	2021-05-06 21:56:14	35140fcf-5ace-4025-8390-c95ea9c95c16	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	Temp.	2021-05-06 21:00:00	16.1
S0013	2021-05-06 21:56:09	71f6382e-a901-47bc-86ce-e6f440a48737	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	Humidity	2021-05-06 21:00:00	49.8
S0007	2021-05-06 21:56:13	f297a696-f51d-4e8d-89e7-220c8fd248f6	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	PM2.5	2021-05-06 21:00:00	15
S0014	2021-05-06 21:56:09	faad1629-4ccc-407a-ba14-b8ea35f63dde	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	NO2	2021-05-06 21:00:00	3.1
S0010	2021-05-06 21:56:13	b7a63a36-ce9c-43ca-8c49-c9097ea48c8b	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	Wind	2021-05-06 21:00:00	325.12
S0017	2021-05-06 21:56:09	99418100-29c4-458e-8e9b-03ef978742c7	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	PM10	2021-05-06 21:00:00	14
S0029	2021-05-06 21:56:14	68ef23fc-d66e-4c00-a32e-0ed2cc238ba6	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	PM10	2021-05-06 21:00:00	14
S0033	2021-05-06 21:56:14	1da00264-996c-4a1b-8cce-25c9573bd138	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	Wind	2021-05-06 21:00:00	49.42
S0027	2021-05-06 21:56:14	43befd79-e606-451e-a7a6-462338a1d379	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	O3	2021-05-06 21:00:00	32.8
S0003	2021-05-06 21:56:13	6ac83527-7f9c-4bf1-a500-5ed7a534f29c	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	NO2	2021-05-06 21:00:00	1.1
S0034	2021-05-06 21:56:14	757e1559-a963-4cad-ae2b-13fa48a4d1ba	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	WG	2021-05-06 21:00:00	10.5
S0038	2021-05-06 21:56:14	f24f79ab-5cfd-4ee4-91ec-812d291ec657	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	Precipitation	2021-05-06 21:00:00	70
S0019	2021-05-06 21:56:09	89f032b1-2863-464c-9c01-e4dfb02e6786	42.648872	21.137121	8f7fff7f-3402-4c58-af32-af429941b73f	SO2	2021-05-06 21:00:00	1.7
S0031	2021-05-06 21:56:14	2b638aa9-5e55-4244-87ff-f221d39b6ea6	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	SO2	2021-05-06 21:00:00	2.4
S0009	2021-05-06 21:56:13	d906b3b5-149f-4157-a8a3-d44f39b3d6c0	42.625349	20.891036	3c333b0e-03c3-4a99-9e36-9e0457db381a	Temp.	2021-05-06 21:00:00	17
S0025	2021-05-06 21:56:14	5f3a8eee-de7b-47de-bcdc-095a7ed723b0	42.661995	21.15055	8068d3e5-cc42-43df-b97f-be55557fa36d	Humidity	2021-05-06 21:00:00	50

Figure 25. Data of the ArchivalDataStreams

Cassandra 3.11.4 : IoTSAS.Cassandra.Connection : iot : archivaliotdatastreamannotations

annotationid	observationid	annotateddate	annotatedtype	annotatedvalue
697221bd-17a6-4bf7-86e4-781268478473	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 21:58:24	Embedded	{"MaxParam":"O3"}
bb962a75-65f8-4229-bae4-2776bb9c1bc2	8068d3e5-cc42-43df-b97f-be55557fa36d	2021-05-06 21:56:14	XLink	{"Higher_Level_Feature_Blizzard":"http://myserver/ontologies/ont-core.owl#HigherLevelFeature_Blizzard"}
d4e6962b-2d1e-4e61-926c-6c63a6108e1d	3c333b0e-03c3-4a99-9e36-9e0457db381a	2021-05-06 21:56:13	Embedded	{"AIQ_Index":"45.0"}
c19c6d79-dc45-4fa2-9342-4243d5eb1c35	8f7fff7f-3402-4c58-af32-af429941b73f	2021-05-06 21:56:09	XLink	{"Air_Pollution_Level":"http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Good"}
191f4feb-72fc-4525-aaf7-b05316eb87f7	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 21:58:24	XLink	{"Health_Implications":"http://myserver/ontologies/ont-core.owl#Health_ImplicationsGood"}
7149c748-f551-4cc2-85df-b64a3b625e88	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 21:58:24	XLink	{"Air_Pollution_Level":"http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Good"}
3f09e342-c5ad-4364-9465-f364defae2cf	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 21:58:24	Embedded	{"AIQ_Index":"38.1"}
cbe7d743-bd78-4e28-b761-ff881f570f94	3c333b0e-03c3-4a99-9e36-9e0457db381a	2021-05-06 21:56:13	Embedded	{"MaxParam":"O3"}
e322c1db-2ae3-4d1b-b184-324e4498c26a	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 21:58:24	XLink	{"Health_Implications":"http://myserver/ontologies/ont-core.owl#Health_ImplicationsModerate"}
c894fa66-2f27-406f-a637-bda1857f03ac	8f7fff7f-3402-4c58-af32-af429941b73f	2021-05-06 21:56:09	Embedded	{"MaxParam":"O3"}
776fbcd2-3950-49de-8dea-84a74a4c6b7a	8f7fff7f-3402-4c58-af32-af429941b73f	2021-05-06 21:56:09	Embedded	{"AIQ_Index":"38.1"}
e7bf7c62-037a-4e84-92d2-0bc099681d58	8068d3e5-cc42-43df-b97f-be55557fa36d	2021-05-06 21:56:14	Embedded	{"MaxParam":"PM2.5"}
a117c46f-a053-489c-8f2f-b3f0db7889a6	3c333b0e-03c3-4a99-9e36-9e0457db381a	2021-05-06 21:56:13	XLink	{"Higher_Level_Feature_Blizzard":"http://myserver/ontologies/ont-core.owl#HigherLevelFeature_Blizzard"}
9e37b928-4c12-4a88-b3c8-4b137690ab93	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 21:58:24	Embedded	{"MaxParam":"PM2.5"}
0e3fcbe4-f09d-4c6a-8647-ff6172b72bf6	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 21:58:24	Embedded	{"AIQ_Index":"55.0"}
d052fdc8-1a79-455e-bb71-e0da544520db	3c333b0e-03c3-4a99-9e36-9e0457db381a	2021-05-06 21:56:13	XLink	{"Air_Pollution_Level":"http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Good"}
5a93b10e-a27c-41a8-83e3-1fd84b02e38f	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 21:58:24	XLink	{"Air_Pollution_Level":"http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Moderate"}
08e4dec8-0abb-4554-b82e-76af1883d265	8f7fff7f-3402-4c58-af32-af429941b73f	2021-05-06 21:56:09	XLink	{"Higher_Level_Feature_Blizzard":"http://myserver/ontologies/ont-core.owl#HigherLevelFeature_Blizzard"}
8b8a82d0-d33f-49b2-9464-1df268393034	3c333b0e-03c3-4a99-9e36-9e0457db381a	2021-05-06 21:56:13	XLink	{"Health_Implications":"http://myserver/ontologies/ont-core.owl#Health_ImplicationsGood"}
33931a20-4cc5-47d7-82de-4de756e2f9c2	8f7fff7f-3402-4c58-af32-af429941b73f	2021-05-06 21:56:09	XLink	{"Health_Implications":"http://myserver/ontologies/ont-core.owl#Health_ImplicationsGood"}
7c39c91c-340f-4a0f-8659-1bfaa8d86250	8068d3e5-cc42-43df-b97f-be55557fa36d	2021-05-06 21:56:14	XLink	{"Health_Implications":"http://myserver/ontologies/ont-core.owl#Health_ImplicationsModerate"}
233c4efd-253c-4426-9aab-06fea176f45e	3173c459-a46a-481c-a8d0-ad348009695e	2021-05-06 21:58:24	XLink	{"Higher_Level_Feature_Blizzard":"http://myserver/ontologies/ont-core.owl#HigherLevelFeature_Blizzard"}
e58a4e60-36b8-4c49-80c6-5ae0fbe08ee7	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 21:58:24	Embedded	{"MaxParam":"O3"}
545fb25e-28ca-4389-9f5f-330496f40e05	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 21:58:24	Embedded	{"AIQ_Index":"45.0"}
bebcf74a-2895-4ccf-850a-ff60a7279cc4	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 21:58:24	XLink	{"Higher_Level_Feature_Blizzard":"http://myserver/ontologies/ont-core.owl#HigherLevelFeature_Blizzard"}
5a28de0d-edf9-439a-8e23-7b668cbb42a5	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 21:58:24	XLink	{"Higher_Level_Feature_Blizzard":"http://myserver/ontologies/ont-core.owl#HigherLevelFeature_Blizzard"}
d9008aab-00b2-4e27-acf8-6f42c9bc8c2f	3cfbc510-81d6-46a0-9047-9b8f44d8d3df	2021-05-06 21:58:24	XLink	{"Air_Pollution_Level":"http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Good"}
97ed8b15-c514-46a7-974b-a83b311b01...	59d2018d-e07e-4cfd-a615-84a4f9c21094	2021-05-06 21:58:24	XLink	{"Health_Implications":"http://myserver/ontologies/ont-core.owl#Health_ImplicationsGood"}
86debcba-4403-4a7f-a140-84b41b08c381	8068d3e5-cc42-43df-b97f-be55557fa36d	2021-05-06 21:56:14	XLink	{"Air_Pollution_Level":"http://myserver/ontologies/ont-core.owl#Air_Pollution_Level_Moderate"}
fcc95ed0-4114-4ef2-9446-3f20281d8b82	8068d3e5-cc42-43df-b97f-be55557fa36d	2021-05-06 21:56:14	Embedded	{"AIQ_Index":"55.0"}

Figure 26. Data of the ArchivalDataStreamAnnotations

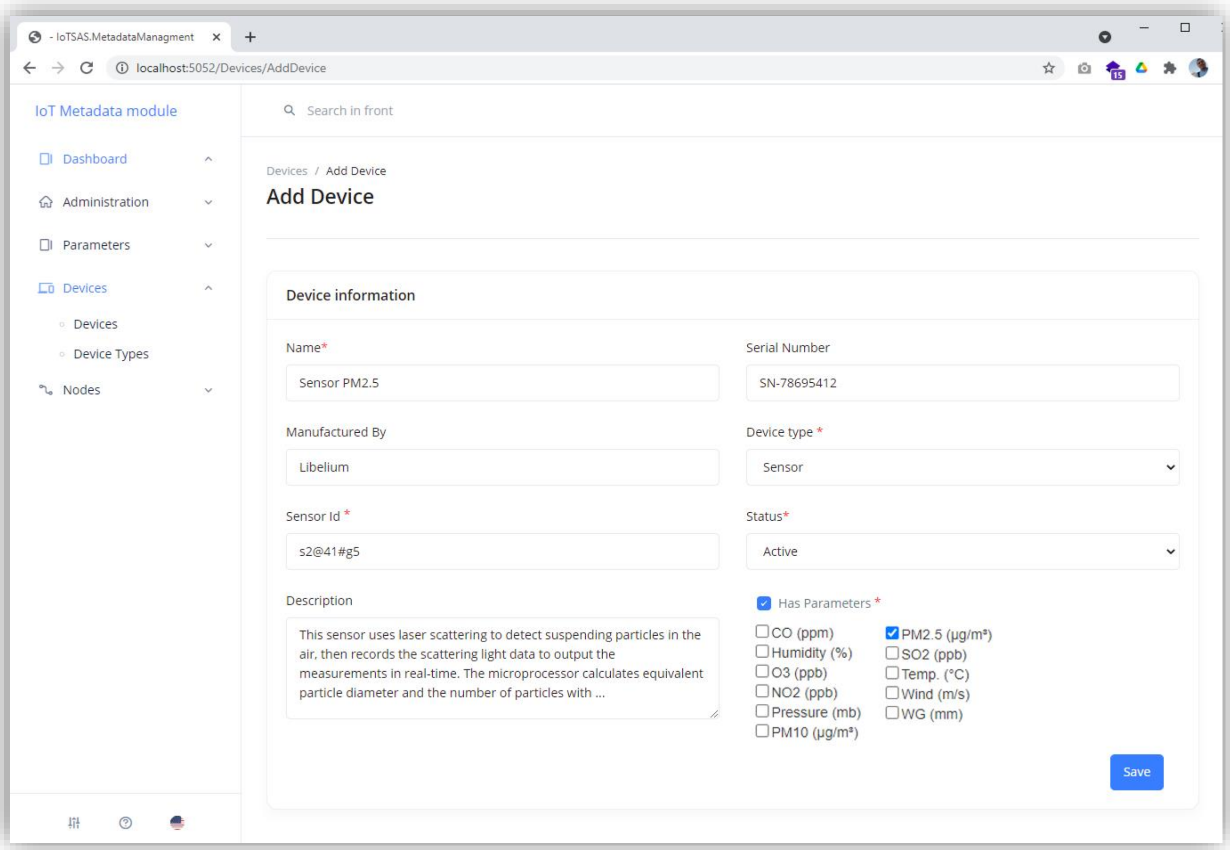


Figure 27. Module for managing IoT metadata – Register a new sensor.

B. Nodes meta data — consists of information about sensor nodes, which include components:

- *SensingNodeTypes* – such as WSN nodes that are stationary, which are used to monitor tasks in a given zone, or mobility WSN nodes, which are used to monitor missions in several places.
- *CentralMonitoringNodes* – provide information for example node's name, description, status, and geographic location.
- *GatewayNodes* – provide the name, city, deployment site, status, details info, and geographic location of the gateway node, as well as the central monitoring node to which the gateway node transmits data.
- *DeploymentLocations* – provide the name of the area, a description, and the municipality where the sensors were installed.
- *SensingNodes* – offer the following information about sensing nodes: name, details info, type of node, Radio-Frequency Identification (RIFD), location of deployment, city, data rate in min., geographical location, state of node, and the gateway node with which they interact.

C. Phenomenon meta data - includes information regarding parameters such as:

- *Parameters* – includes data such as those shown in Figure 29: parameter name (e.g. Humidity, Ozone (O3), Carbon Monoxide (CO), Wind, NO2, SO2, pm10, pm25, Temperature, Water Gauge, and so on), unit of phenomenon (e.g. µg/m³, ppm%, mb, ppb°C, mm, m/s, etc.), and rage of values.
- *Subparameter types* – information on subparameter types such as river continuity, hydrological regime, thermal conditions, morphological conditions, phytoplankton, air pollution, nutrient conditions, macrophysics, salinity, phytobenthic, and so on.
- *Parameter types* – information regarding parameter types such as physicochemical, hydromorphological, particular synthetic, biological, air quality, specific non synthetic, and so on.

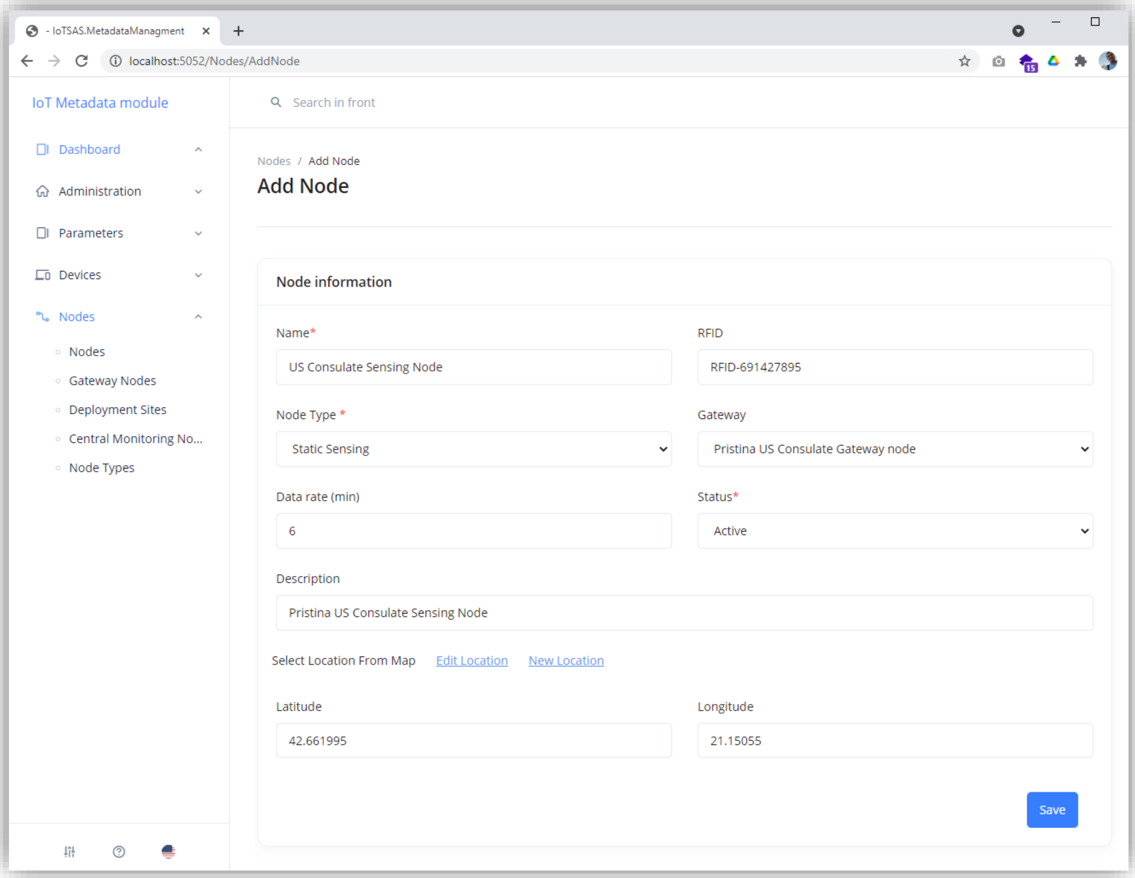


Figure 28. Module for managing IoT metadata – Register a new Sensing node.

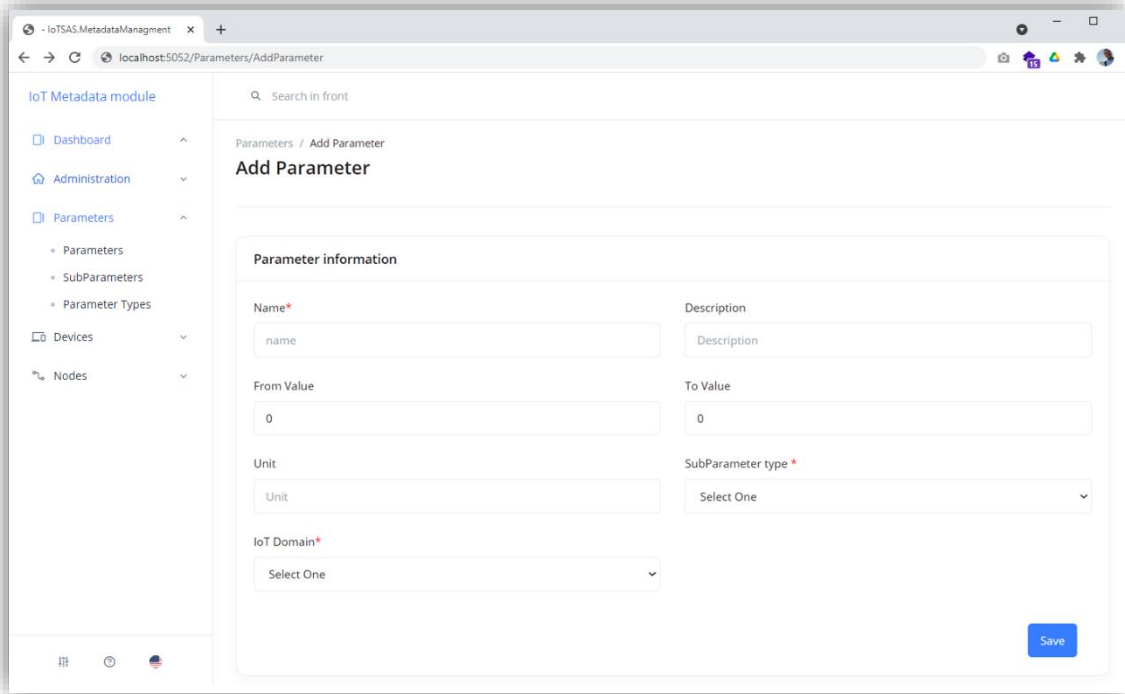


Figure 29. Module for managing IoT metadata – Register a new Parameter.

5.4. Monitoring module for air quality and weather warnings

To view observed WSN data and the semantic annotations that go along with it, a real-time monitoring IoT web application is built. Access to sensor data from the Hydrometeorological Institute of Kosovo (HMIK), the United States Consulate in Pristina, Peje, and Rilindja-Pristina is made possible via the “World Air Quality Index API (AQI API)”. Programmatic integration of the AQI API includes access to data over 10000 stations and 1000 paces, as well as title and locations of each monitoring station, a geolocation request based on lat. and long., individual AQI for each pollutant, as well as the most recent weather conditions (Aqicn, 2020).

A. Received observed WSN data format

The IoTSAS gets raw observed WSN data via the World Air Quality Index API in JSON format, as depicted in Figure 30. The system is able to continuously monitor the phenomenon such as: “Temperature (t), Ozone (o3), Sulphur Dioxide (so2), PM10 (pm10), PM25 (pm25), Nitrogen Dioxyde (no2), Carbon Monoxide (co), Humidity (h), Pressure (p), Water Gauge (wg), and Wind (w)”. There are several different types of information that can be found in JSON data, such as: *data* (*idx* - the monitoring station's unique identifier, *aqi* - air quality data in real time, *timing* - the observation timing data, *s* - local observation time, and *tz* - the time zone of the station); *city* (*name* - details about the station's location, *geo* - including its lat. and long., as well as a link to the source - *url*); *attributions* (the station's EPA attribution); and *iaqi* (*pm25* – AQI object for PM2.5, *v* - actual AQI value for PM2.5). Observation acquired by WSNs each 60 min. interval via the World Air Quality Index API are expressed numerically, for example, as 33.3 (co) for the Carbon Monoxide phenomena.

```
{
"status": "ok",
"data": {
  "aqi": 58,
  "idx": 12402,
  "attributions": [
    {
     "url": "http://worldweather.wmo.int",
     "name":"World Meteorological Organization
- surface synoptic observations (WMO-SYNOP)"
    },
    {
      "url": "http://ihmk-rks.net/",
      "name": "Instituti Hidrometeorologjik i
Kosovës",
      "logo": "Kosovo-IHMK.png"
    },
    {
     "url": "https://waqi.info/",
     "name": "World Air Quality Index Project"
    }
  ],
  "city": {
    "geo": [ 42.648872, 21.137121 ],
    "name": "Prishtine - IHMK, Kosovo",
    "url":
"https://aqicn.org/city/kosovo/prishtine-ihmk"
  },
  "dominentpol": "pm25",
  "iaqi": {
    "co": { "v": 33.3 }, "h": { "v": 76 },
    "no2": { "v": 6.2 }, "o3": { "v": 23.3 },
    "p": { "v": 1015.7 }, "pm10": { "v": 17 },
    "pm25": { "v": 58 }, "so2": { "v": 6.3 },
    "t": { "v": 1.6 },"w": { "v": 14 },
    "wg": { "v": 23 }
  },
  "time": {
    "s": "2020-03-25 20:00:00",
    "tz": "+01:00", "v": 1585166400
  },
  "debug": { "sync": "2020-03-
26T04:17:09+09:00" }
}}
```


Figure 30. Observed WSN data - JSON format

B. Integrating and interpreting of semantic annotations into the observed WSN data

As part of the developed IoTSAS, various semantic annotations for observed WSN data are created, including as:

- *#AQIIndexAnnotation*
- *#HealthImplicationsAnnotation*
- *#AirPollutionLevelAnnotation*
- *#FlurryAnnotation*
- *#BlizzardAnnotation*
- *#RainShowerAnnotation*
- *#RainStormAnnotation*

The *#AQIIndexAnnotation* – is a daily air quality index that indicates how good or filthy the air is. The AQI for five key air pollutants controlled by the Clean Air Act is calculated by the United States Environmental Protection Agency (EPA): ground-level ozone, particle pollution (also known as particulate matter), carbon monoxide, sulfur dioxide, and nitrogen dioxide . The AQI scale is a numeric value between 0 and 500. The EPA states that the greater the AQI value, the more air pollution there is and the higher the center, as illustrated in Equation 1:

$$AQI = \max(AQI_{PM2.5}, AQI_{PM10}, AQI_{O3}, ...) \quad (1)$$

Equation 1. Calculation of *#AQI_Index* (Air quality index) annotation

#AirPollutionLevelAnnotation – it is categorized into six "*Air Quality Index Levels of Health Concern*" categories based on the World Air Quality Index value:

① Good (AQI is 0 to 50)
② Moderate (AQI is 51 to 100)
③ Unhealthy for Sensitive Groups (101 to 150)
④ Unhealthy (AQI is 151 to 200)
⑤ Very Unhealthy (AQI is 201 to 300)
⑥ Hazardous (AQI is 301 to 500)

#HealthImplicationsAnnotation – all of the six above-mentioned categories correlates to a different level of health concert. *#HealthImplicationsAnnotation* indicates what they imply, e.g. "*Unhealthy for Sensitive Groups*" category suggests the following: "Although the general public is not likely to be affected at this AQI range, people with lung disease, older adults, and children are at a greater risk from exposure to ozone, whereas persons with heart and lung

disease, older adults, and children are at great.", or "Air quality is acceptable; however, for some pollutants there may be a moderate health concern for a very small number of people who are unusually sensitive to air pollution" states the "Moderate" category.

#BlizzardAnnotation – to identify a *Blizzard*, WindSpeed must exceed 15.6 m/s (high), snow precipitation must last at least 4 hours, and Visibility must be less than 400 meter (low) (Canada, 2020), as shown in Equation 2:

BLIZZARD = WindSpeed(a) ≥ 15.6 m/s (High) ∧ Duration(a) > 4hours ∧ Precipitation(b) = Snow ∧ Duration(b) > 4hours ∧ Visibility(c) < 400 meter (Low) ∧ Duration(c) > 4hours	(2)

Equation 2. Calculation of *#Blizzard* annotation

#FlurryAnnotation – there must be lower than 15.6 meters per second of wind, at least 4 hours of snow precipitation, and 400 meters of visibility in order to detect a Flurry, according to Equation 3:

FLURRY = WindSpeed(a) < 15.6 m/s (Low) ∧ Duration(a) > 4hours ∧ Precipitation (b) = Snow ∧ Duration(a) > 4hours ∧ Visibility(c) < 400 meter (Low) ∧ Duration(c) > 4hours	(3)

Equation 3. Calculation of *#Flurry* annotation

#RainStormAnnotation – to detect this annotation, the WindSpeed must be greater than 15.6 m/s (high), rain precipitation, and the temperature must be greater than 0°C, as shown in Equation 4:

RAIN STORM = WindSpeed(a) ≥ 15.6 m/s (High) Λ Precipitation(b) = Rain Λ Temperature (c) > 0°C	(4)

Equation 4. Calculation of *#RainStorm* annotation

#RainShowerAnnotation – to detect this annotation, the WindSpeed must be lower than 15.6 m/s (low), rain precipitation, the temperature must be more than 0°C, as shown in Equation 5:

RAIN SHOWER = WindSpeed(a) < 15.6 m/s (Low) Λ Precipitation (b) = Rain Λ Temperature (c) > 0°C	(5)

Equation 5. Calculation of *#RainShower* annotation

The above-mentioned annotations are being developed into an ontology called "ont-core.owl". Figure 31 shows the annotations for air quality monitoring, while Figure 32 shows the annotations for weather warnings monitoring.

When semantics are added in real time to observed data of different WSNs types in the IoT, the RTISA component is used to interpret the observed WSN data in real time to provide better understanding and to derive new knowledge from the observed WSN data. The following interpretation pattern is produced in this investigation applying integrated semantic annotation stakes:

Now (@**[#timestamp]**) in location **[#location(lat, long)]** is detected **[#AQI_index]** AQ index with primary pollutant **[#MaxParam] [#MaxParamUnit]**, and **[#Air_Pollution_Level]** air pollution level which health implications **[#Health_Implications].** Also happening a **[#HigherLevelFeature]** higher level feature which manifests **[#HigherLevelFeature_Indicates].**

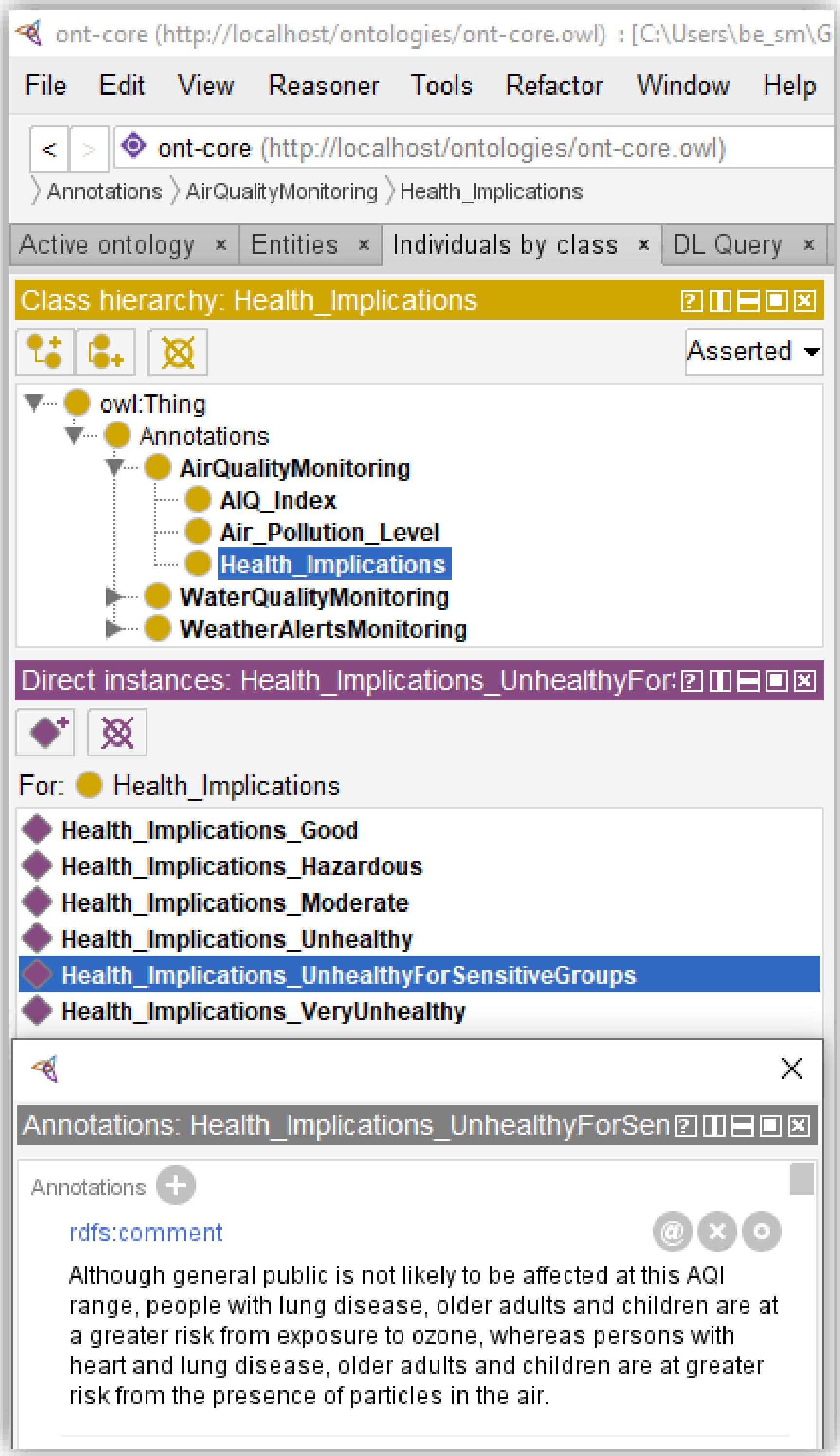


Figure 31. 'ont-core.owl' ontology for Air Quality Monitoring

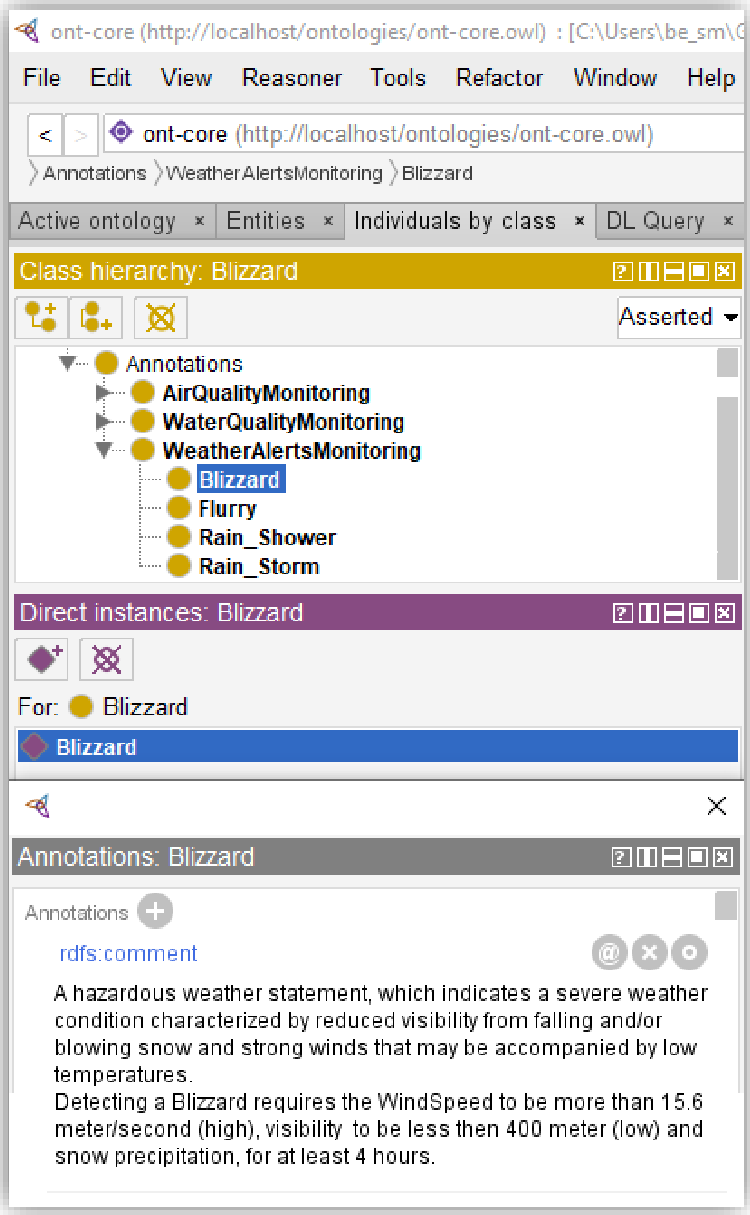


Figure 32. 'ont-core.owl' ontology for Weather Alerts Monitoring

Following a description of the various kinds of semantic annotations for observed WSN data, the technique of semantic annotations is provided in the following section.

The observed WSN data can come in many varied formats to the Kafka server (in our example, JSON format), that will translate them into an appropriate type which will be handled by Apache Spark Streaming. Following that, using Spark Streaming, the observed WSN data will be tagged with semantics and transformed to OGC Observation and Measurement standard depending on measurement values. Figure 33 depicts a piece of a sample of integrated/interpreted semantic annotations to the observation OGC Observation

& Measurement standard document utilizing stacks such as Embedded and XLink, while Figure 34 depicts a portion of a complex observation OGC Observation & Measurement standard document format.

```
<sos:Observation ...="">
<observationData>
  <om:OM_Observation gml:id="o1">
    <om:type xlink:href="http://www.opengis.net/def/observationType/OGC-
OM/2.0/OM_Measurement"/>
    <om:phenomenonTime>
      <gml:TimeInstant gml:id="phenomenonTime_1">
        <gml:timePosition>2021-05-20T12:44:02+09:00</gml:timePosition>
      </gml:TimeInstant>
    </om:phenomenonTime>
    <om:resultTime xlink:href="#phenomenonTime_1"/>
    <om:procedure xlink:href="http://myserver/ontologies/ont-core.owl#Sensor562415"/>
    <om:observedProperty xlink:href="http://myserver/ontologies/ont-core.owl#PM25"/>
    <om:featureOfInterest xlink:href="http://myserver/ontologies/ont-
core.owl#Prishtine"/>
    <om:result xsi:type="gml:SemMeasureType" uom="pm25">
      <value>58</value>
      <sem-annotations>
        <annotation xlink:href="http://myserver/ontologies/ont-
core.owl#Air_Pollution_Level_Moderate"/>
        <annotation embedded:AIQ_Index ="58"/>
        <annotation xlink:href="http://myserver/ontologies/ont-
core.owl#Health_Implications_Moderate "/>
      </sem-annotations>
      <sem-interpretations>
       Now (@2021-05-20 12:44:02) in location 'Pristina US Consulate (42.648872, 21.137121)'
is detected '58' AQ index with primary pollutant 'PM2.5 µg/m³', and 'Moderate' air pollution
level which health implications 'Air quality is acceptable; however, for some pollutants there
may be a moderate health concern for a very small number of people who are unusually sensitive
to air pollution'. Also happening a 'Blizzard' higher level feature which manifests 'A
hazardous weather statement, which indicates a severe weather condition characterized by
reduced visibility from falling and/or blowing snow and strong winds that may be accompanied
by low temperatures. Detecting a Blizzard requires the WindSpeed to be more than 15.6
meter/second (high), visibility to be less then 400 meter (low) and snow precipitation, for at
least 4 hours.'
      </sem-interpretations>
    </om:result>
  </om:OM_Observation>
</observationData>
. . .
</sos:Observation>
```

Figure 33. OGC O&M Observation – Integrated/interpreted semantic annotations to the IoT air quality monitoring sensor stream data

```
<sos:Observations>
  <om:OM_Observation gml:id="69822a61-5490-47b4-aaf4-b282b6df7824" ...
    <gml:description>Complex Observation instance</gml:description>
    <gml:name>Complex Observation</gml:name>
    <om:type xlink:href="http://www.opengis.net/def/observationType/OGC-
OM/2.0/OM_ComplexObservation"/>
    <om:phenomenonTime>
      <gml:TimeInstant
        gml:id="ot1t">
        <gml:timePosition>Thu Aug 26 11:00:00 CEST 2021</gml:timePosition>
      </gml:TimeInstant>
    </om:phenomenonTime>
```

```
    <om:resultTime xlink:href="#ot1t"/>
    <om:procedure xlink:href="http://localhost/ontologies/ont-
core.owl#IHMKSensingNode"/>
    <om:observedProperty xlink:href="http://localhost/ontologies/ont-
core.owl#AirQualityAndWeatherAlertsMonitoring"/>
    <om:featureOfInterest xlink:href="http://localhost/ontologies/ont-
core.owl#Pristina"/>
    <om:result xsi:type="swe:DataRecordPropertyType">
      <swe:DataRecord>
        <swe:field name="CO">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0012">
            <swe:uom code="ppm"/>
            <swe:value>6.0</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="Humidity">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0013">
            <swe:uom code="%"/>
            <swe:value>83.8</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="NO2">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0014">
            <swe:uom code="ppb"/>
            <swe:value>8.4</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="O3">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0015">
            <swe:uom code="ppb"/>
            <swe:value>3.5</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="Pressure">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0016">
            <swe:uom code="mb"/>
            <swe:value>1011.8</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="PM10">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0017">
            <swe:uom code="µg/m³"/>
            <swe:value>14.0</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="PM2.5">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0018">
            <swe:uom code="µg/m³"/>
            <swe:value>52.0</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="SO2">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0019">
            <swe:uom code="ppb"/>
            <swe:value>7.6</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="Temp.">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0021">
            <swe:uom code="°C"/>
            <swe:value>17.6</swe:value>
          </swe:Quantity>
```

```
        </swe:field>
        <swe:field name="Wind">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0022">
            <swe:uom code="m/s"/>
            <swe:value>271.92</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="WG">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0023">
            <swe:uom code="mm"/>
            <swe:value>4.4</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="Visibility">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0036">
            <swe:uom code="m"/>
            <swe:value>390.87</swe:value>
          </swe:Quantity>
        </swe:field>
        <swe:field name="Precipitation">
          <swe:Quantity definition="http://localhost/ontologies/ont-core.owl#S0039">
            <swe:uom code=""/>
            <swe:value>70.0</swe:value>
          </swe:Quantity>
        </swe:field>
      </swe:DataRecord>
      <swe:sem-annotations>
        <swe:annotation Embedded:AQI_Index="52.0" />
        <swe:annotation Embedded:MaxParam="PM2.5" />
        <swe:annotation XLink:href="http://localhost/ontologies/ont-
core.owl#Air_Pollution_Level_Moderate" />
        <swe:annotation XLink:href="http://localhost/ontologies/ont-
core.owl#Health_Implications_Moderate" />
        <swe:annotation XLink:href="http://localhost/ontologies/ont-
core.owl#Blizzard"/>
      </swe:sem-annotations>
      <swe:sem-interpretations>
        Now (@Thu Aug 26 11:00:00 CEST 2021) in location 'IHMK Sensing Node
(42.648872, 21.137121)' is detected '52.0' AQ index with primary pollutant 'PM2.5
µg/m³', and 'Moderate' air pollution level which health implications 'Air quality is
acceptable; however, for some pollutants there may be a moderate health concern for a
very small number of people who are unusually sensitive to ozone may experience
respiratory symptoms.'Also happening a 'Blizzard' higher level feature which manifests
'A hazardous weather statement, which indicates a severe weather condition
characterized by reduced visibility from falling and/or blowing snow and strong winds
that may be accompanied by low temperatures. Detecting a Blizzard requires the
WindSpeed to be more than 15.6  meter/second (high), visibility  to be less than 400
meter (low) and snow precipitation, for at least 4 hours.'.
      </swe:sem-interpretations>
    </om:result>
  </om:OM_Observation>
</sos:Observations>
```


Figure 34. OGC O&M Complex Observation – Integrated/interpreted semantic annotations to the IoT air quality monitoring sensor stream data

Figure 35 shows the process of semantic integration to sensor observation data for a better understanding. Table 4 depicts each of the process steps.

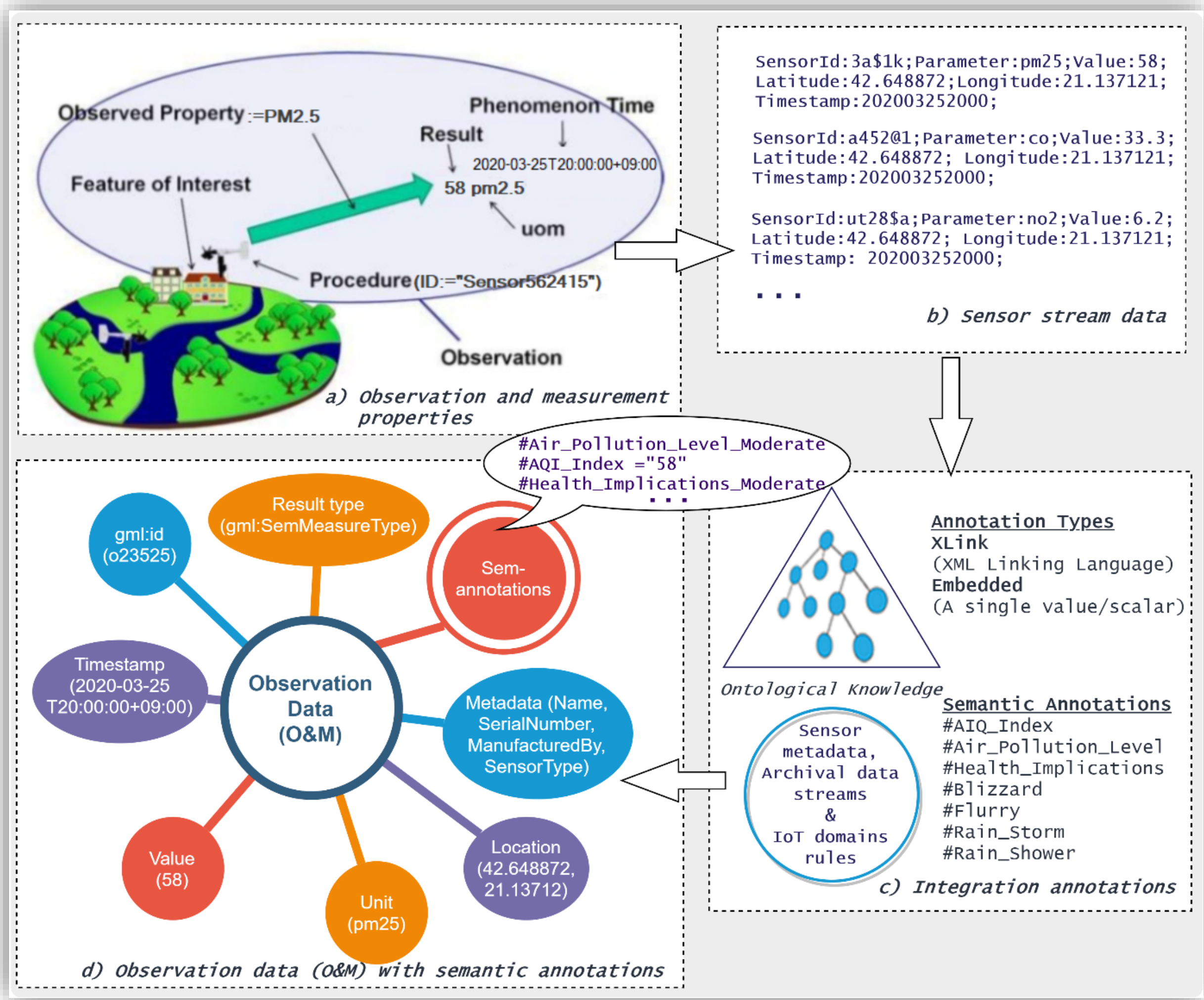


Figure 35. Integrating semantics into observed WSN data

Table 5. Description steps of integration process of semantics into observed WSN data

Step	Description
Observation and measurement properties	the concept of O&M and the relationship between the entities involved in observations,
Sensor stream data	data streams generated from wireless sensor networks
Integration annotations	sensor data integrated with sensor metadata, archival data streams, and the ontological knowledge, and finally
Observation data (O&M) with semantic annotations	semantic annotated data with attributes: sem-annotations data, the observed value, unit, metadata, location, timestamp, result type, and gml:id of observation.

C. Outputs of the System

A real-time Internet of Things application was built using ASP.NET Core MVC, a powerful framework for creating web applications that follow Model-View-Controller design pattern, to show the observed data of different WSNs types and its semantic annotations. The "DataStax C# for Apache Cassandra" package is utilized to get data from the Cassandra DB. It's a C# client library with a lot of features and a lot of configuration options. The data is displayed on the map using Leaflet, a JS framework for interactive maps. Leaflet is a little program that focuses on simplicity, efficiency, and usability.

As illustrated in Figures 36, 37, and 38, users can monitor the air quality and weather warnings at specific location designated as measurement (sensing) nodes on a map. Each marker (sensing node) provid the AQI Index associated with it to show the level of air pollution and weather warnings. When a marker is clicked, the most recent measurement values for that point are displayed, including PM2.5, PM10, SO2, O3, pressure, CO, NO2, humidity, wind speed, temperature, and water gauge, as well as semantic annotations like *#AQIIndexAnnotation*, *#AirPollutionLevelAnnotation*, *#HealthImplicationsAnnotation*, and weather warnings like *#BlizzardAnnotation*, *#FlurryAnnotation*, *#RainStormAnnotation*, or *#RainShowerAnnotation*, if any of them have been detected.

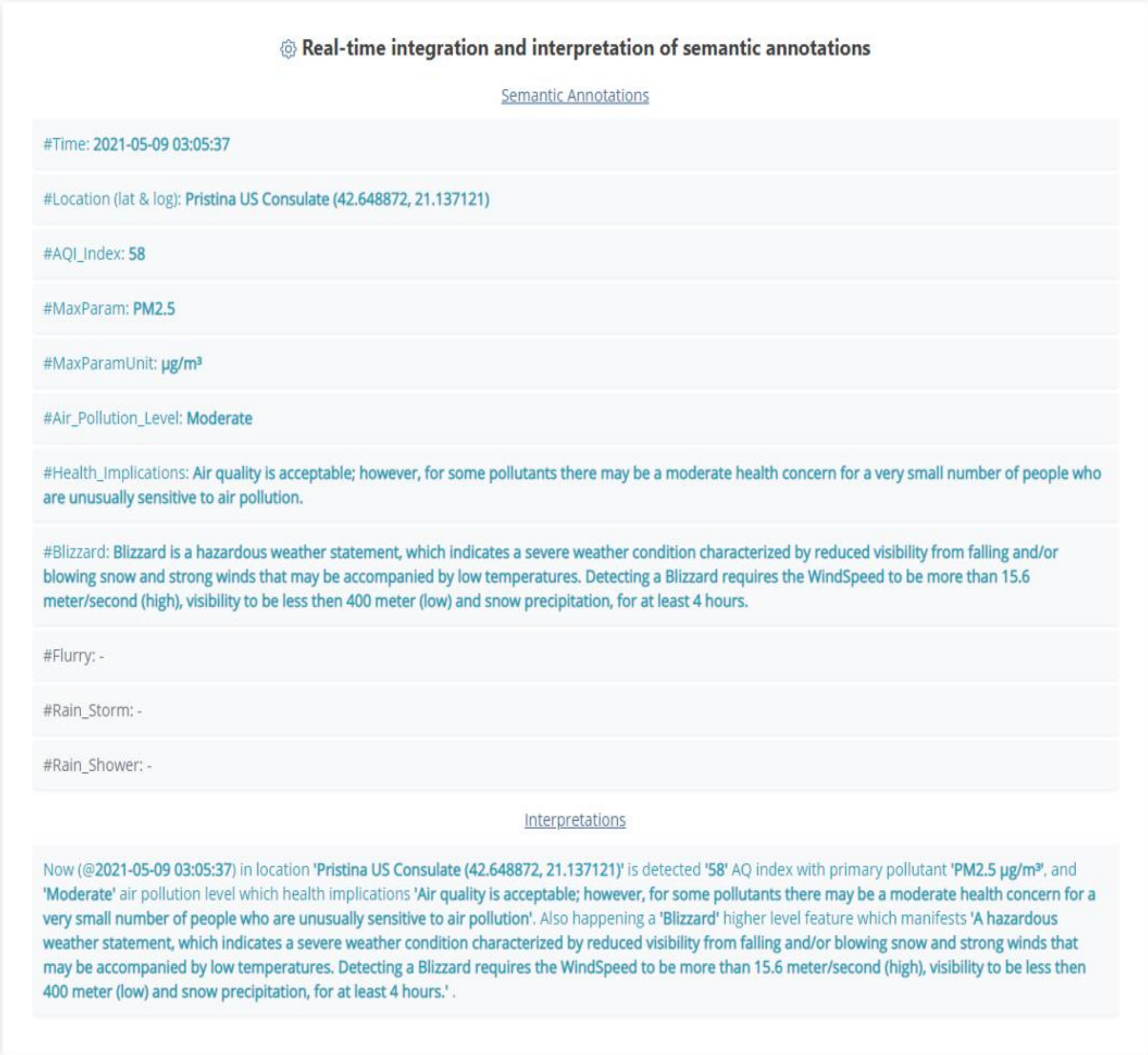
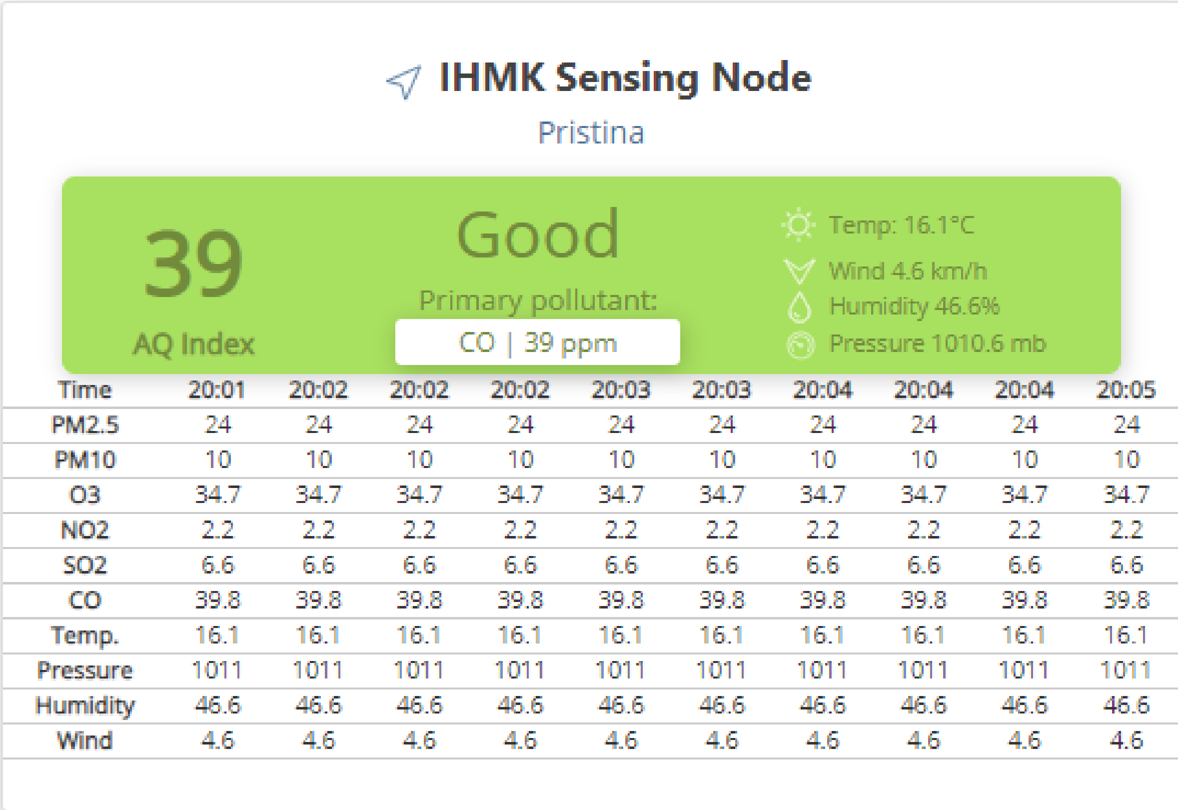


Figure 36. Outputs of the System: Real-time interpretation of the observed WSN data.



Time	20:01	20:02	20:02	20:02	20:03	20:03	20:04	20:04	20:04	20:05
PM2.5	24	24	24	24	24	24	24	24	24	24
PM10	10	10	10	10	10	10	10	10	10	10
O3	34.7	34.7	34.7	34.7	34.7	34.7	34.7	34.7	34.7	34.7
NO2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2
SO2	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6	6.6
CO	39.8	39.8	39.8	39.8	39.8	39.8	39.8	39.8	39.8	39.8
Temp.	16.1	16.1	16.1	16.1	16.1	16.1	16.1	16.1	16.1	16.1
Pressure	1011	1011	1011	1011	1011	1011	1011	1011	1011	1011
Humidity	46.6	46.6	46.6	46.6	46.6	46.6	46.6	46.6	46.6	46.6
Wind	4.6	4.6	4.6	4.6	4.6	4.6	4.6	4.6	4.6	4.6

Figure 37. Outputs of the System: air quality monitoring.

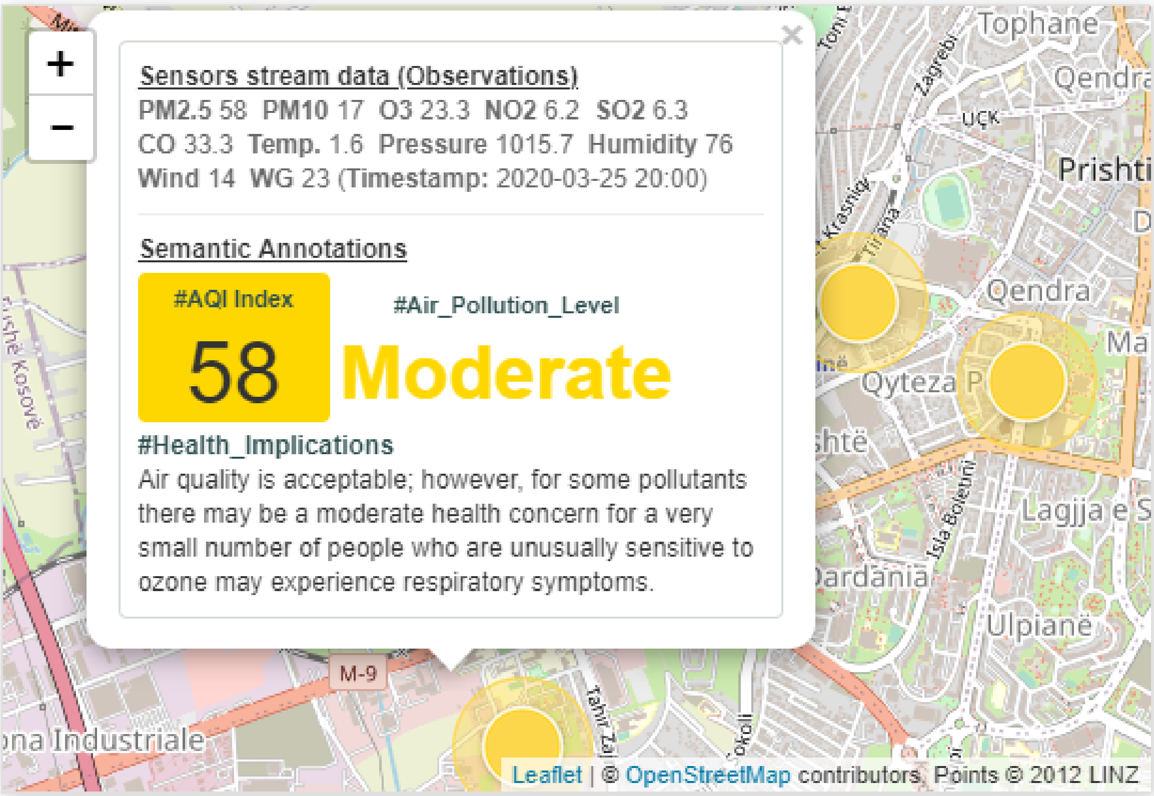


Figure 38. Outputs of the System: sensing nodes in map view.

5.5. Water quality monitoring

The module for water quality monitoring uses cutting-edge technological trends, like WSNs, that allow continuously monitoring and are comprised of nodes known as motes that are sensitive to their location of deployment, to monitor water quality in real time.

A. Received sensor stream data format

The water quality monitoring module allows for the measurement of water phenomenon like dissolved oxygen, temperature, conductivity, and hydrogen potential. Table 6 details the kind, rank, and unit of these phenomenon. The WSN outputs data in the form of a numerical value, for example, 85% for the dissolved oxygen. The WSNs are arranged so that each node transmits data once every ten minutes. The sensor stream data is obtained from InWaterSense, a European Union-funded initiative supervised by the European Union Office in Kosovo and implemented by the University of Prishtina's Faculty of Civil Engineering and Architecture and Faculty of Electrical and Computer Engineering (Ahmedi, 2018).

Table 6. Specification of water parameters.

Parameter name	Parameter Type	Range value	Unit
Temperature	Thermal conditions	-1 to +50	°C
pH	Acidification status	0 to 14	pH
Dissolved Oxygen	Oxygenation conditions	0 to 300	%
Conductivity	Salinity	150 to 5000	μS/cm

Figure 39 shows how nodes communicate. All of these components are located in Plemetin (42.70670318, 21.03843116), include a gateway node, a monitoring node, and a static wireless sensor node (Figure 40). Static wireless sensing nodes are stationary and communicate data to the central monitoring node through the gateway, while mobile WSNs (Figure 41) may move from site to location to assess water phenomenon.

The ZigBee protocol is utilized to transport sensor stream data from static sensing nodes to

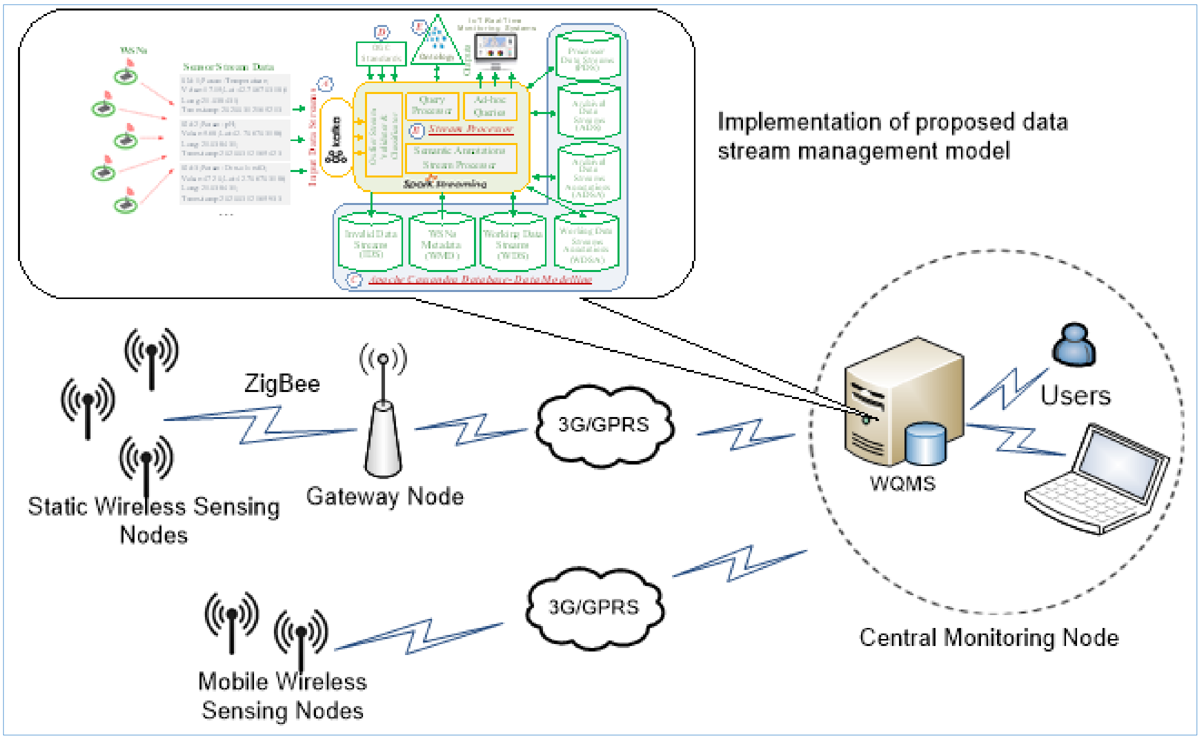


Figure 39. Communication between nodes

Figure 40. System implementation in Plemetin - static sensor nodes

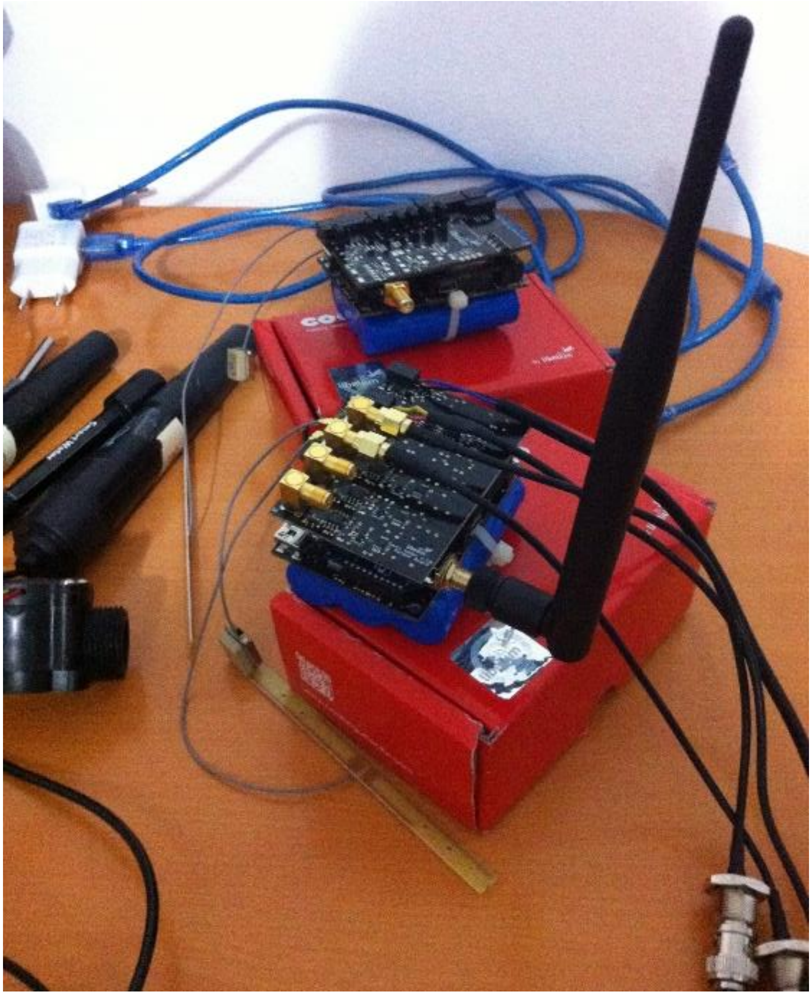


Figure 41. System implementation in Plemetin - mobile sensor nodes

gateway nodes, while the 3G/GPRS protocol is used to communicate between gateway nodes and the central monitoring node via SOAP web services.

B. Integrating and interpreting of semantic annotations into the observed WSN data

Varius water status semantic annotations for international regulating of water quality are produced in the *'ont-core.owl'* ontology (see Figure 42), such as:

- *#UNECE* – for "United Nations Economic Commission for Europe" (UNECE[17]):
    - Class I,
    - Class II,
    - Class III,
    - Class IV, and
    - Class V.

[17] https://www.unece.org/

- *#WFD* – for “Water Framework Directive” (WFD[18]):
    - Good,
    - Moderate,
    - Poor,
    - Bad, and
    - High.

An example of annotation for the Conductivity parameter (µS / cm) is given in Figure 43. If the value observed by the Conductivity sensor is in the range 0.00 – 500.00 µS / cm, the water status is categorized as high and the semantic annotation result is #High. If the value observed by the sensor is between 500.00 - 700.00 µS / cm, then the system creates the annotation #Good. Both of these types of stators are accepted in terms of water quality by WDF. For other annotations like: #Bad (2000.00 - 5000.00 µS / cm), #Poor (1000.00 - 2000.00 µS / cm), #Moderate (700.00 - 1000.00 µS / cm), and failing to achieve good (unacceptable - does not meet WDF goals).

The following are the calculation of annotations for the parameter Dissolved Oxygen (%) (Markogianni, 2018):

- #Bad: 0% - 2%
- #Poor: 2% - 4%
- #Moderate: 4% - 6.4%
- #Good: 6.4% - 9%
- #High: 9% - 300%

The calculation of water status for temperature parameter is:

- #Bad: 29.00 °C - 50.00 %
- #Good: 0.00 °C - 29.00 %

[18] ec.europa.eu

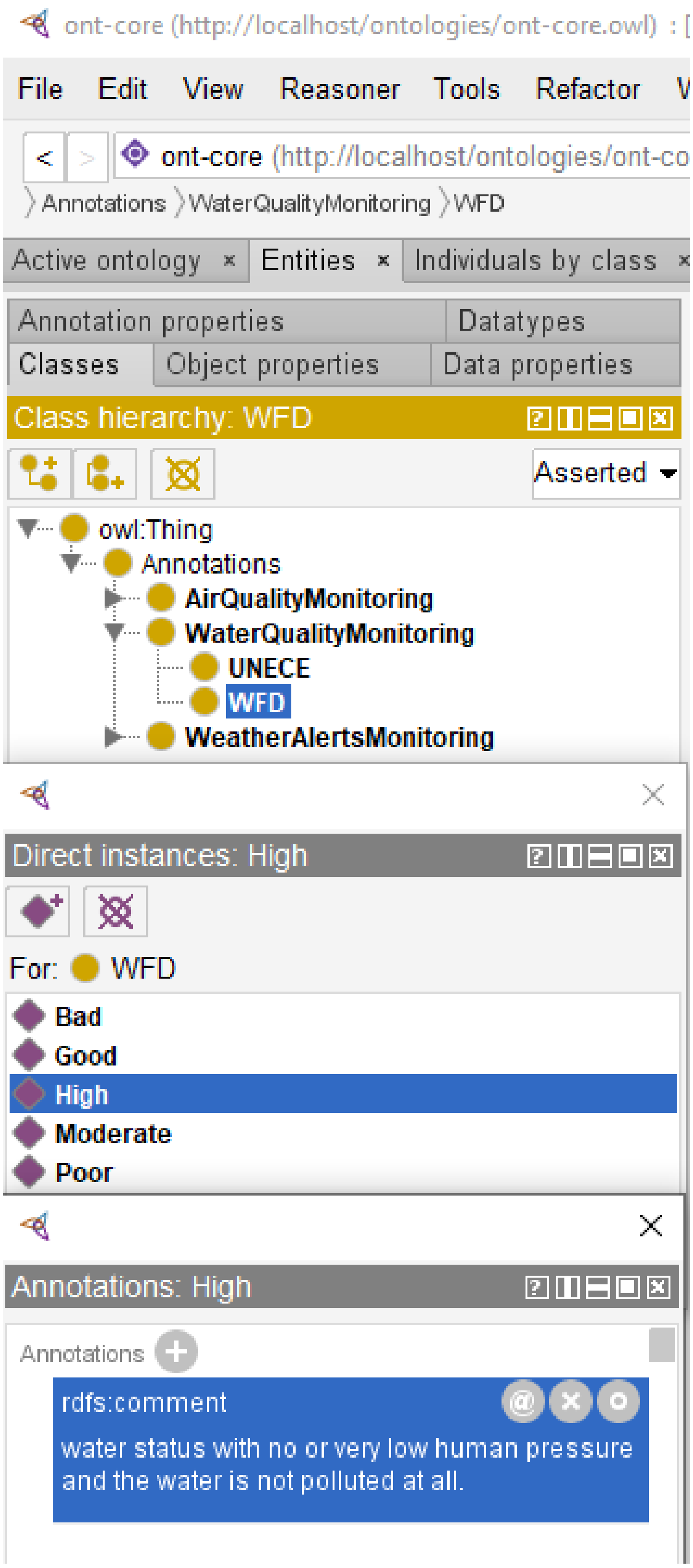


Figure 42. 'ont-core.owl' ontology for Water Quality Monitoring

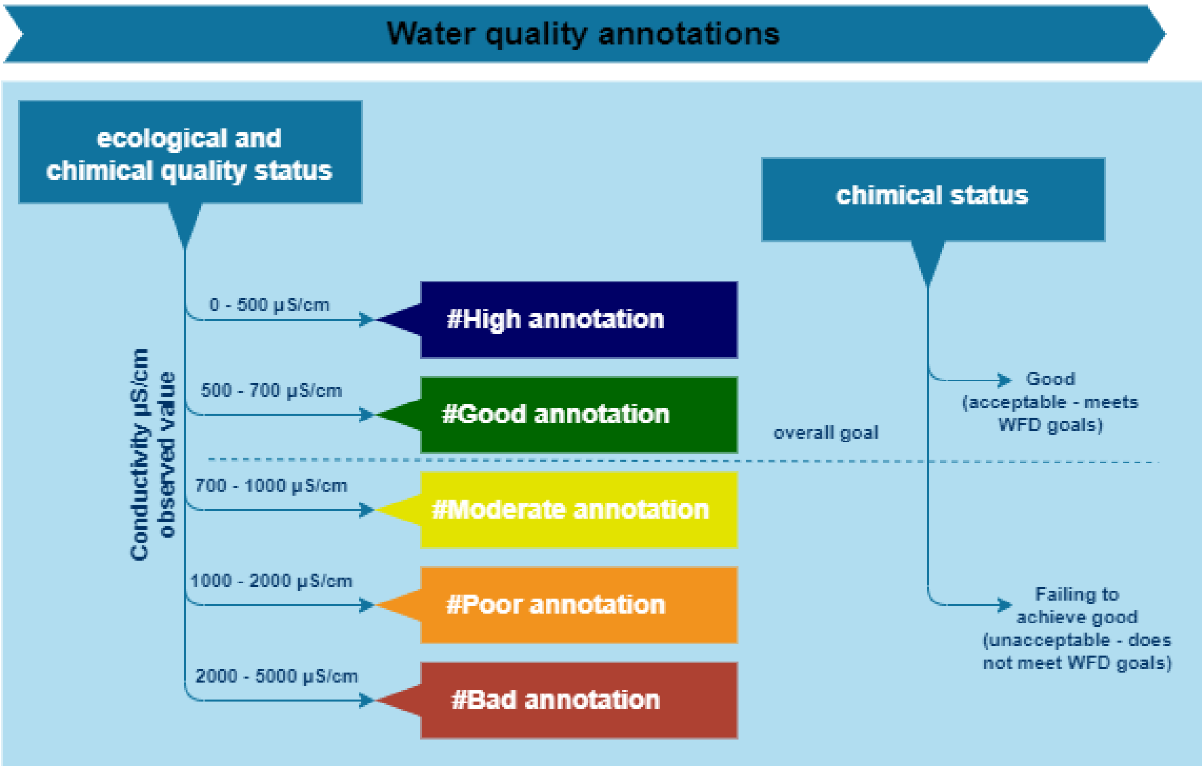


Figure 43. Water status annotation

The results of the enriched observed WSN data with semantic annotations are saved in the Cassandra DB, and they will be shown in real-time monitoring IoT applications in the style of the OGC Observation & Measurement standard using technologies like XLink. Figure 44 illustrates a portion of an output example. Following the real-time integration of semantics into observed water WSN data, the RTISA component performs real-time interpretation of the observed WSN data to enable a better understanding and infer new information from the data. For the IoT domain of water quality, the following interpretation pattern has been developed:

```
Now (@[#timestamp]) in location [#location(lat, long)] is detected
'[#ParamValue] [#ParamUnit]' [#ParamName], '[#WaterStatus]' water
status, which [#WaterStatus_Indicates].
```

```xml
<sos:Observation ...="">
  <observationData>
    <om:OM_Observation gml:id="o45125">
      <om:type xlink:href="http://www.opengis.net/def/
observationType/OGC-OM/2.0/OM_Measurement"/>
      <om:phenomenonTime>
        <gml:TimeInstant gml:id="phenomenonTime_1">
          <gml:timePosition>2020-04-03T16:40:00+09:00
</gml:timePosition>
        </gml:TimeInstant>
      </om:phenomenonTime>
      <om:resultTime xlink:href="#phenomenonTime_1"/>
      <om:procedure xlink:href="http://myserver
/ontologies/ont-core.owl#Sensor3"/>
      <om:observedProperty xlink:href="http://
myserver/ontologies/ont-core.owl#DissolvedOxygen"/>
      <om:featureOfInterest xlink:href="http://
myserver/ontologies/ont-core.owl#Plemetin"/>
     <om:result xsi:type="gml:SemMeasureType" uom="%">
        <value>47.20</value>
        <sem-annotations>
          <annotation embedded:WaterStatus ="High"/>
          <annotation xlink:href="http://myserver/
ontologies/ont-core.owl# WaterStatus_Indicates_High"/>
        </sem-annotations>
        <sem-interpretations>
          Now (@2020-04-03 16:40) in location ‘Pelemetin (42.70670318, 21.03843116)’ is
detected ’47.20%’ dissoleved oxygen, which indicates a ‘High’ water status with no or
very low human pressure and the water is not polluted at all.
        </sem-interpretations>
      </om:result>
    </om:OM_Observation>
  </observationData>
  . . .
</sos:Observation>
```

Figure 44. OGC O&M Observation – Integrated/interpreted semantic annotations to the IoT water quality monitoring sensor stream data

C. System Outputs

A real-time IoT web app is created to show the enriched observed WSN data with semantic annotations. Figure 45 depicts the application interface, which provides monitoring of water quality in real-time using mobile and static WSNs.

The water quality monitoring module of the IoTSAS system, runs continuous queries on the suggested model to present data. The data presented in the textboxes for each phenomenon is collected from *ProcessorDataStreams* component via continual queries. *WorkingDataStreams* provide the data displayed in the charts, while *WorkingDataStreamAnnotations* provide the semantic annotated data that indicate the water state.

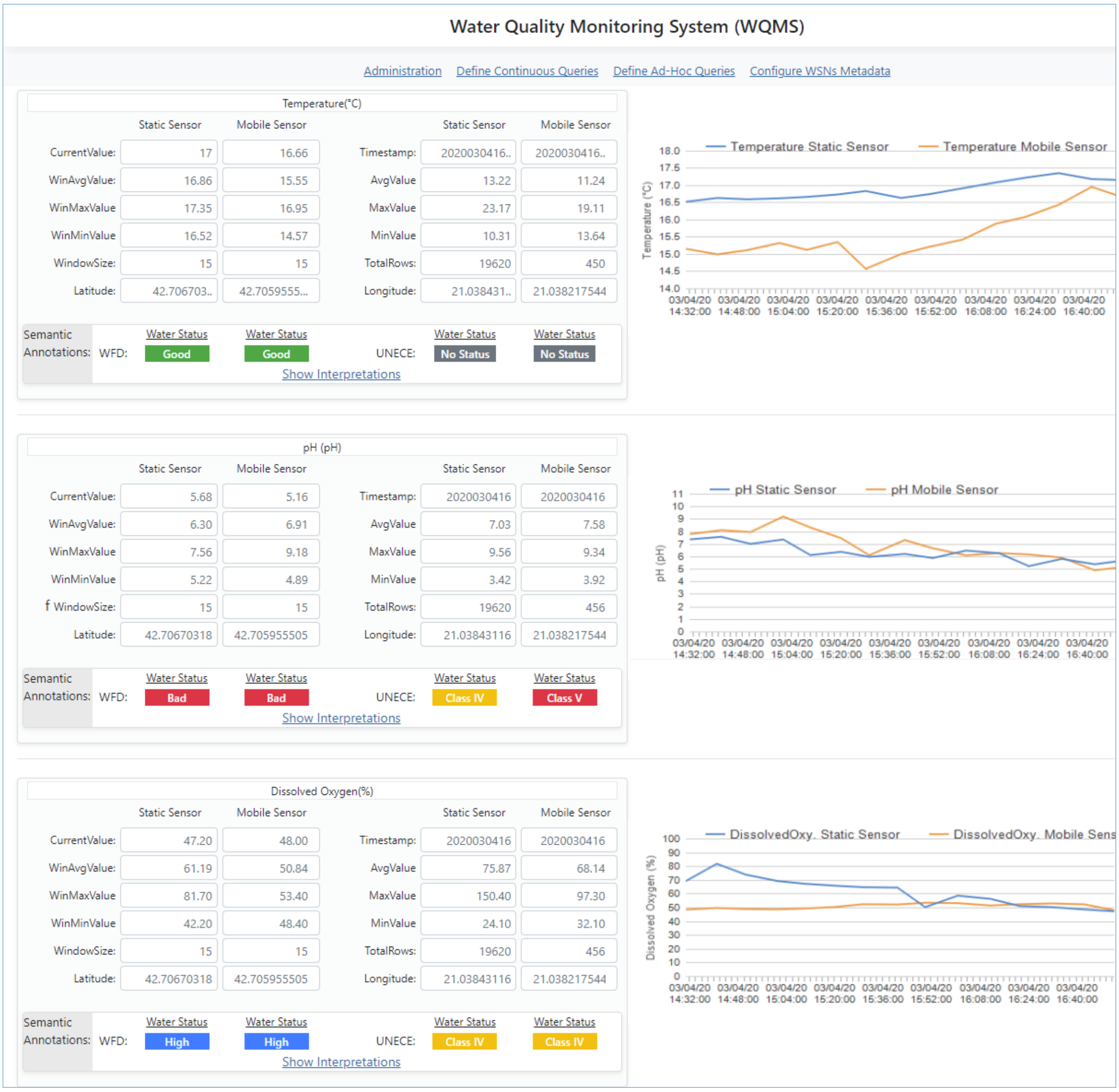


Figure 45. Outputs of water quality monitoring module.

WorkingDataStreams, as previously stated, constitute a pre-configured sliding window with a predetermined size, such as 15, that may be set up in the module. This implies that the charts show the last 15 readings from each sensor on each graph. The trigger for continually executing queries is activated as soon as observed data from the WSNs enters the IoTSAS system.

Annotation interpretation includes information such as timestamp, location (including latitude and longitude) of sensing node, phenomena with measured value, and meaning of current water status, as depicted in Figure 46.

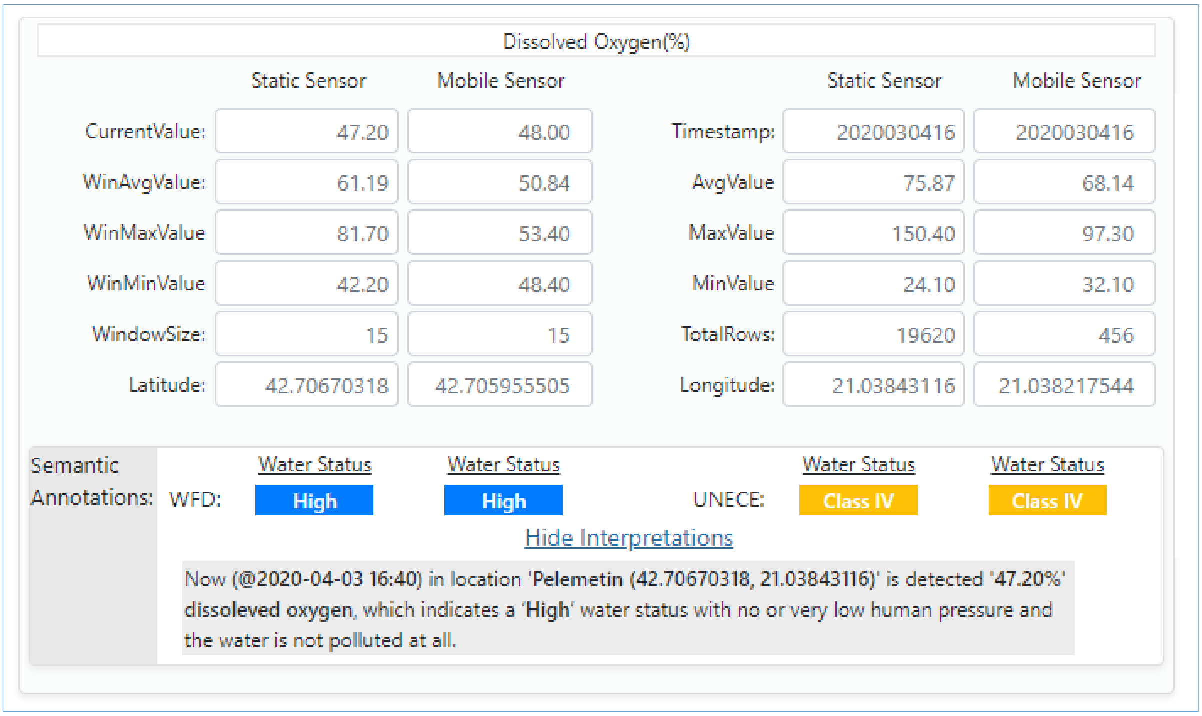


Figure 46. Interpretation of semantic annotations - water quality monitoring module.

"Now (@2020-04-03 16:40) in location 'Pelemetin (42.70670318, 21.03843116)' is detected '47.20%' dissolved oxygen, which indicates a 'High' water status with no or very low human pressure and the water is not polluted at all."

Using Wireless Sensor Networks, a water quality monitoring sample output is provided in Table 7. They are constantly tracked and shown in real time.

Table 7. The proposed model's outcomes

Water parameter	Sensor Type	Current Value	Timestamp	Min	Max	Avg.	Total rows	Window Average	Window Max	Window Min	Window Size	Location (Latitude, Longitude)	Semantic Annotations (Water Status)	
													WFD	**UNECE**
Temperature (°C)	Static	17.00	2020-04-03 16:52:33	10.31	23.17	13.22	19620	16.86	17.35	16.52	15	42.7067031... 21.0384311...	Good	No Status
	Mobile	16.66	2020-04-03 16:52:33	13.04	19.11	11.24	456	15.55	16.95	14.57	15	42.7059555... 21.0382175...	Good	No Status
pH	Static	5.68	2020-04-03 16:52:33	3.42	9.65	7.03	19620	6.30	7.56	5.22	15	42.7067031... 21.0384311...	Bad	Class IV
	Mobile	5.16	2020-04-03 16:52:33	3.92	9.34	7.58	456	6.91	9.18	4.89	15	42.7059555... 21.0382175...	Bad	Class V
	Static	47.20	2020-04-03 16:52:33	24.10	150.40	75.87	19620	61.19	81.70	47.20	15	42.7067031... 21.0384311...	High	Class IV

Dissolved Oxygen (%)	Mobile	48.00	2020-04-03 16:52:33	32.10	97.30	68.14	456	50.84	53.40	48.40	15	42.7059555... 21.0382175...	High	Class IV
...														

5.6. IoTSAS system network architecture

Figure 47 depicts the overall system network architecture, which includes the following components: *Apache Kafka Server, Spark Streaming Cluster Server, Apache Cassandra database Server, IoT Real-Time Web Application Server,* and *Web Services Server*. The function of each of the servers is described in Table 8.

Table 8. System network architecture - the function of each of the servers

Server	Function
Apache Kafka Server	Apache Kafka operates and receives streaming observed data sent by sensors.
Spark Streaming Cluster Server	Core system (developed in Apache Spark Streaming) is installed.
Apache Cassandra database Server	Server where the Apache Cassandra DB is installed to store all system data.
IoT Real-Time Web Application Server	These are hosted in Internet Information Services (IIS) modules such as the following: weather alerts monitoring module, air quality monitoring module, and metadata management module.
Web Services Server	These are deployed APIs for external systems.

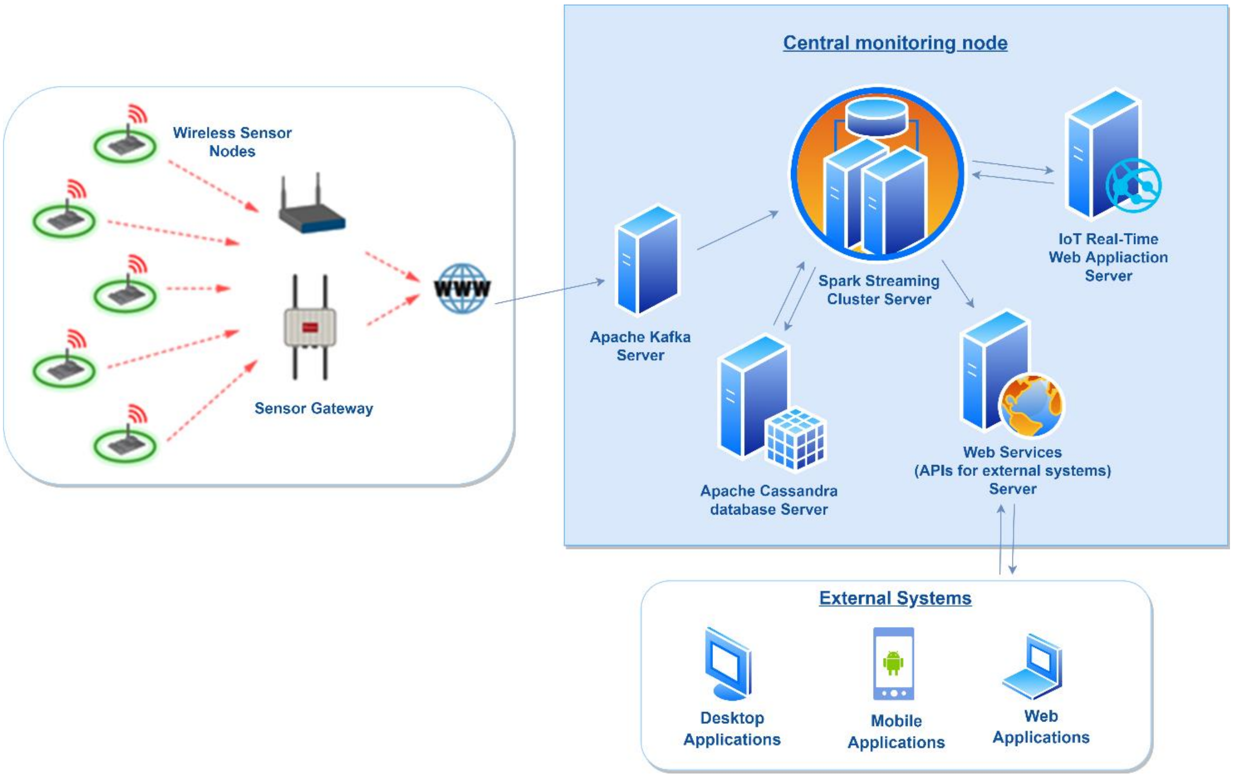


Figure 47. IoTSAS – system network architecture

5.7. IoTSAS system security

As part of the IoTSAS system's security features, each sensor is given a passcode in addition to the WSN meta data that is recorded in the module for managing IoT meta data. The WSN also communicates a passcode with the observation data, so that it can be identified.

To protect data sent across network modules, the Secure Sockets Layer (SSL) protocol is utilized. The module for managing metadata, monitoring module for air quality and weather warnings, and water quality monitoring module, are accessible to users via username and password. The .NET System.Security.Cryptography is used to encrypt the password. In other words, the HMACSHA512 hash function, which is part of the Secure Hashing Algorithm (SHA) 512 hash library. The hashing procedure includes the addition of salt in order to ensure that the passwords are unique and to improve the complexity of the password. In order to prevent SQL injection attacks, .Net 5.0 LINQ to Entities is utilized because LINQ is not vulnerable to SQL injection.

5.8. WSNs stream data simulator

It was necessary to simulate a large volume of observed WSN data in order to test the operation of the IoTSAS system. As a result, a WSNs stream data emulator, as shown in Figure 48, was created to accomplish this. The simulator generates pseudo-random observed WSN data using the Random C# class[19], within the ranges given in the metadata module for each parameter (phenomenon). According to, the simulator can generate temperature values ranging from -25 to 45 degrees Celsius (NASA, 2021). The simulator, as illustrated in Figure 48, can be set to generate data at millisecond, second, or minute intervals. Additionally, can choose a specific sensor node to generate observed WSN data, can choose a higher level feature, such as Blizzard, Flurry, Rain Shower, or Rain Storm, to generate data from sensors that induce this phenomena. The simulator may create observed WSN data in batches and transfer it to the IoTSAS system for processing.

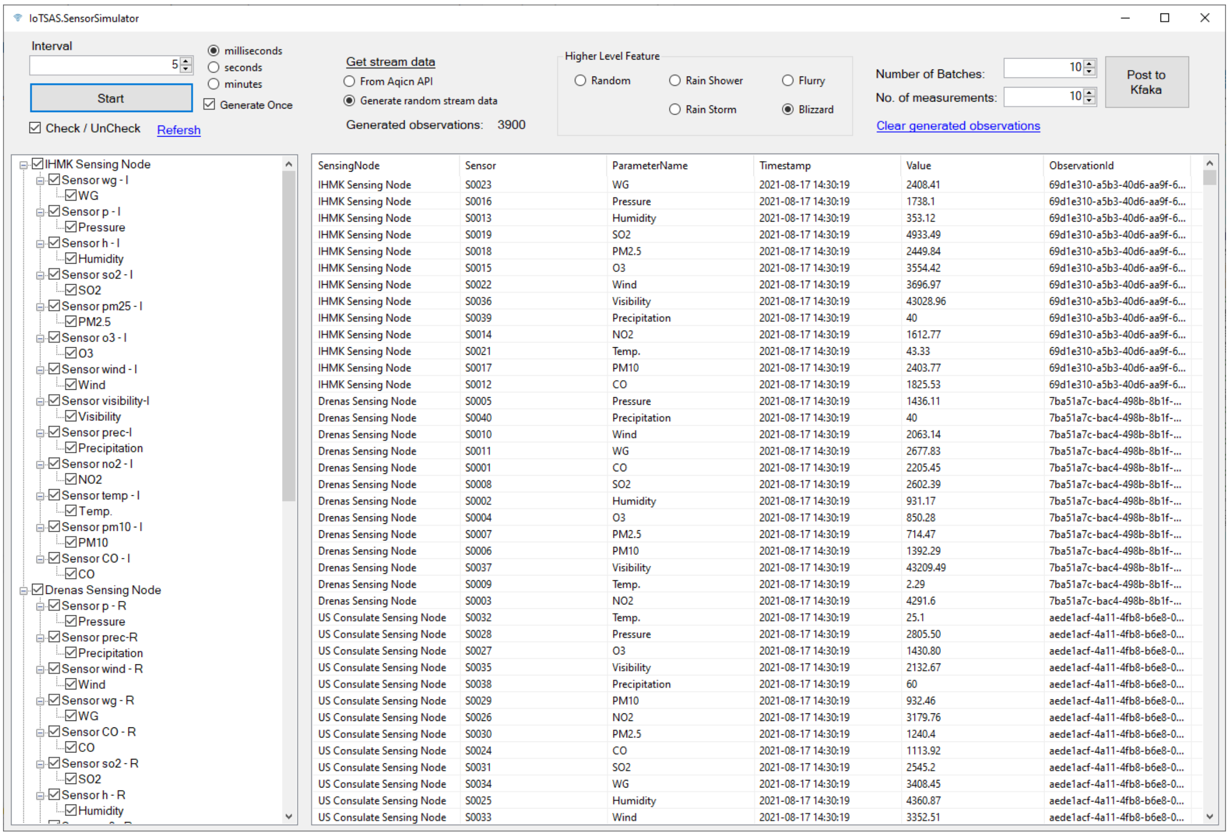


Figure 48. Sensor stream data simulator

[19] https://docs.microsoft.com/en-us/dotnet/api/system.random?view=net-5.0

6

Chapter

6. Testing of the System

On five testing stages, seven modules are tested: (1) real-time integrating and interpreting of semantic annotations into the observed WSN data module, (2) module for managing metadata, (3) monitoring module for air quality and (4) weather warnings, (5) water quality monitoring module, (6) data modelling module, (7) module for external systems - RESTful APIs.

Unit test - the unit test is based on the system specification and covers the results of errors that were made during the coding process.

Integration test - a scenario-based test is used to determine whether or not all seven components work together flawlessly. During this stage, Data Flow testing is also carried out, which includes testing each step-in turn.

System test - in this step, all modules are tested to make sure they work together without any problems, just like in the previous phase. Here the system is also checked for compliance with all the application requirements and security issues such as security level (XSS — Cross Site Scripting, SQL injections, and encryption of modules' communications), confidentiality of information, restrictions on accessibility, and immunity.

Acceptance test (alpha and beta) - in this phase, the IoTSAS is tested with real data from sensors of the HMIK, the United States Consulate in Pristina, Peje, and Rilindja-Pristina (for air quality and weather alerts monitoring domains) as well as data from the InWaterSense project (for water quality), as specified in Sections 5.4 and 5.5.

Performance testing - tests of the IoTSAS system's performance were performed using the simulator (detailed in section 5.8). Figure 47 depicts the network architecture being tested. Table 9 shows the technical details of the hardware environment in which the test is run.

Table 9. Technical details of the hardware environment

Server	Processor	RAM Memory	OS
Spark Streaming Cluster Server	Intel® Xeon® CPU x5570 @ 2.93GHz (4 CPUs), ~2.9GHz	32GB	Windows Server 2016 Datacenter 64-bit (10.0, Build 14393)
Apache Cassandra database Server	Intel® Xeon® CPU x5570 @ 2.93GHz (4 CPUs), ~2.9GHz	32GB	Windows Server 2016 Datacenter 64-bit (10.0, Build 14393)
Apache Kafka Server	Intel® Core™ i5-4200M CPU @ 2.50GHz (4 CPUs), ~2.5GHz	8GB	Windows 10 Pro 64-bit (10.0, Build 19042)
IoT Real-Time Web Application Server	Intel® Core™ 2 Duo CPU e7500 @ 2.93GHz (2 CPUs), ~2.9GHz	16GB	Windows Server 2019 Datacenter 64-bit (10.0, Build 17763)
Web Services Server	Intel® Xeon® CPU E5-2650 v4 @ 2.20GHz (4 CPUs), ~2.2GHz	6GB	Windows Server 2012 Standard 64-bit (6.3, Build 9600)

Table 10 presents the performance test results for the IoTSAS system. The tests are executed for various generated observed WSN data and are repeated three times to obtain more accurate averages considering the current load of the processor, memory in use by active processes, network, etc.

Table 10. System performance test results from the IoTSAS system

Number of observations	Test 1 (seconds)	Test 2 (seconds)	Test 3 (seconds)	AVG (seconds)
100	0.122	0.118	0.128	0.123
500	0.184	0.154	0.207	0.182
1,000	0.287	0.281	0.269	0.279
5,000	0.901	0.909	0.897	0.902
10,000	1.417	1.372	1.329	1.373
20,000	2.587	2.558	2.807	2.65
50,000	6.634	6.698	6.511	6.61
100,000	14.257	14.443	14.257	14.32
150,000	21.376	21.317	21.749	21.48
250,000	35.245	36.131	34.508	35.29
500,000	67.934	66.927	68.029	67.63
1,000,000	141.07	139.33	134.18	138.20

Figure 49 shows the IoTSAS system's performance of observed WSN stream data generated by 100 to 10,000 WSNs. Semantic annotating and interpreting 100 observed WSN stream data in real-time takes 0.123 seconds, but processing 10,000 observed WSN stream data takes 1.37 seconds on average.

Figure 50 shows the volume testing, which evaluates the IoTSAS system's efficiency when dealing with a huge number of generated observed WSN stream data. For 500,000 observed WSN stream data, the average processing time for semantic annotations and interpretation is 67.63 seconds, whereas for 1,000,000, the average processing time is 138 seconds.

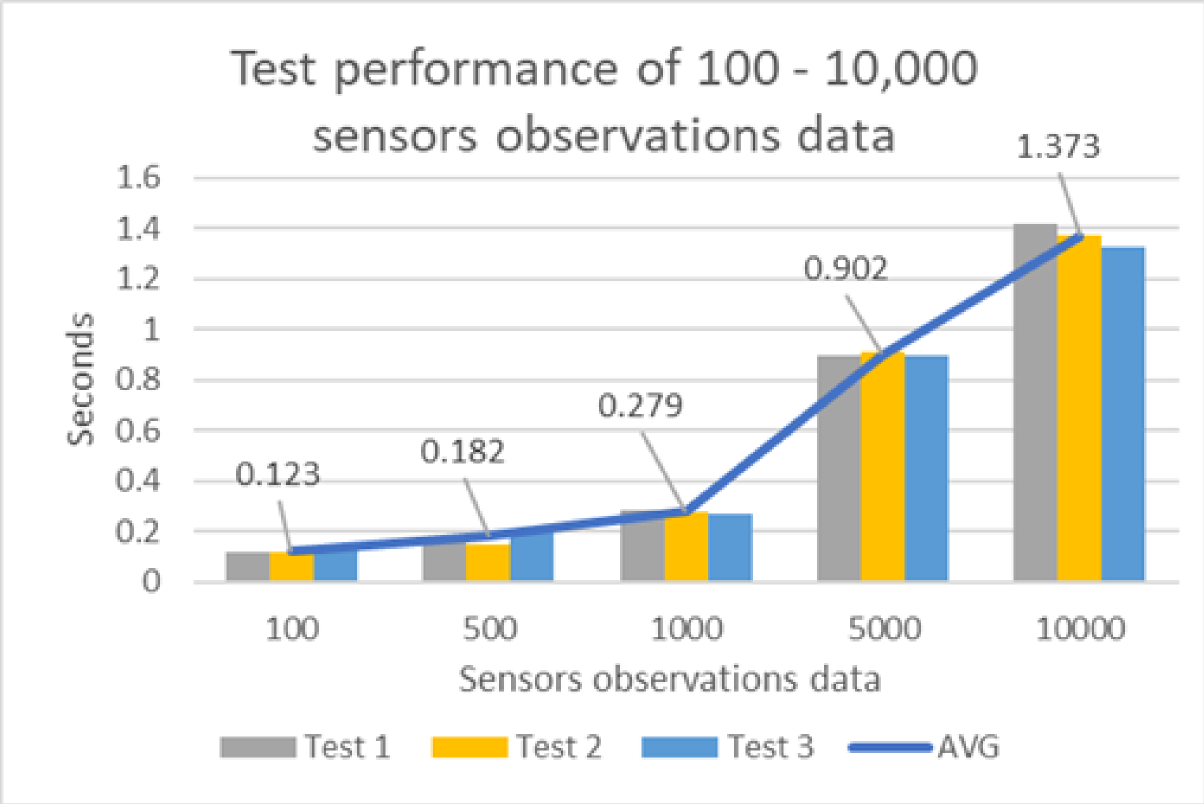


Figure 49. Test performance of 100-10,000 observed WSN data

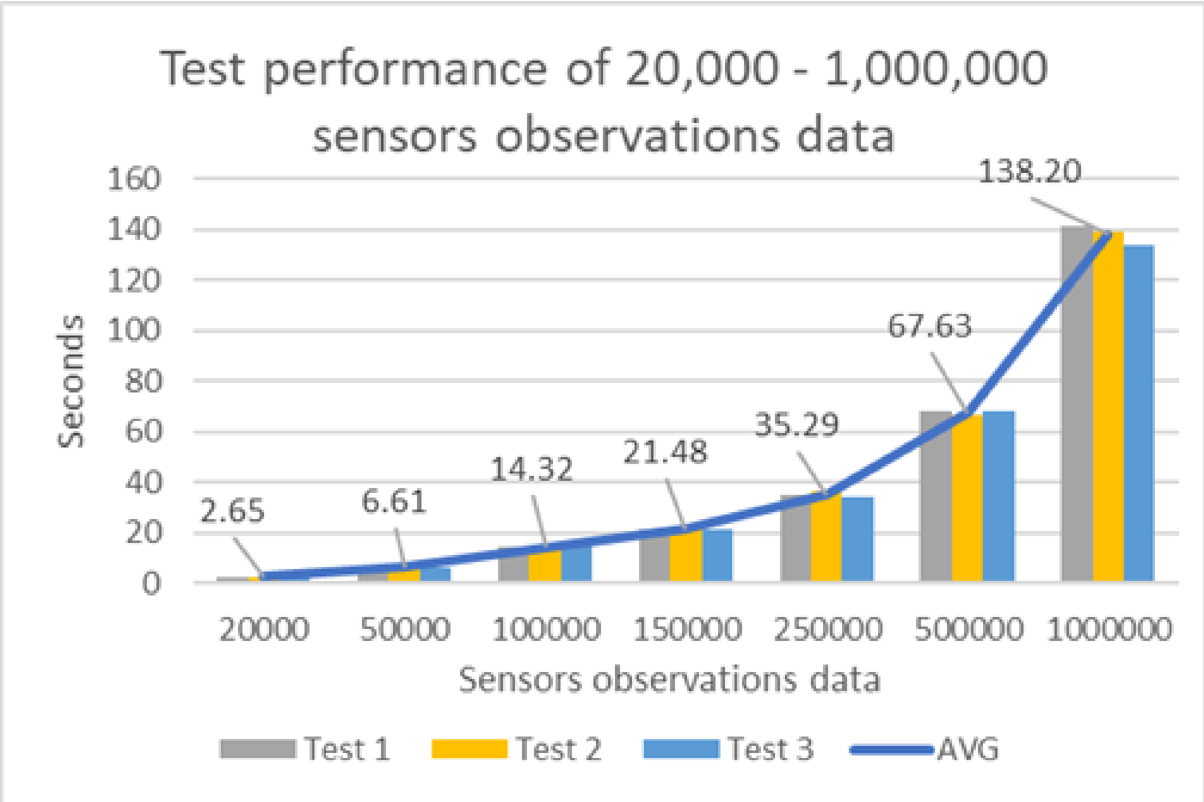


Figure 50. Test performance of 20,000-1,000,000 observed WSN data

Analyzed data from monitoring stations around Europe has been compiled from the World Air Quality Index database. According to data in Table 11, there are 2,510 air quality monitoring stations in Europe transmitting hourly observations to the World Air Quality Index database. The largest amount of parameters (observed phenomena) for a monitoring station is 13, which implies that for a single monitoring station, we have a maximum of 13 sensors observations. The 2,510 monitoring stations in Europe allow for a maximum of 32,630 sensors to collect data. The IoTSAS system can *process*, *annotate*, and *interpret* in real time in less than 50 seconds if all of the sensors' observations are submitted to the server at the same time.

Calculations based on test results show that the IoTSAS system can process, annotate, and interpret 1,000,000 observed WSN data from 76,923 monitoring stations (1,000,000 observed WSN data / 13 parameters per station) in 138 seconds, indicating good system performance.

Table 11. Statistics of monitoring stations in different European countries by area.

#	Country	No. of monitoring stations
1	Albania (link)	2
2	Andorra (link)	1
3	Armenia (link)	1
4	Austria (link)	82
5	Azerbaijan (link)	3
6	Belarus (link)	16
7	Belgium (link)	63
8	Bosnia and Herzegovina (link)	19
9	Bulgaria (link)	24
10	Croatia (link)	23
11	Cyprus (link)	9
12	Czechia (link)	131

13	Denmark (link)	8
14	Estonia (link)	12
15	Finland (link)	55
16	France (link)	158
17	Georgia (link)	6
18	Germany (link)	162
19	Greece (link)	28
20	Hungary (link)	46
21	Iceland (link)	9
22	Ireland (link)	87
23	Italy (link)	130
24	Kazakhstan (link)	47
25	Latvia (link)	23
26	Lithuania (link)	7
27	Luxembourg (link)	4
28	Malta (link)	4
29	Moldova (link)	7
30	Montenegro (link)	6
31	Netherlands (link)	98
32	North Macedonia (link)	19
33	Norway (link)	56
34	Poland (link)	78
35	Portugal (link)	17
36	Republic of Kosovo (link)	8
37	Romania (link)	165

38	Russia ([link])	41
39	Serbia ([link])	118
40	Slovakia ([link])	37
41	Slovenia ([link])	12
42	Spain ([link])	184
43	Sweden ([link])	27
44	Switzerland ([link])	29
45	Turkey ([link])	152
46	Ukraine ([link])	134
47	United Kingdom ([link])	162
	Total monitoring stations	**2,510**

Table 12 compares the IoTSAS system's performance to that of the existing system (Patni, 2011). In paper (Patni, 2011), we remind that is implemented a framework based on Semantic Web technologies, which provides annotations (such as blizzards, flurry, rain storms, and rain showers) using observed WSN data in real-time. Annotations are integrated into observed WSN data using SPARQL rule, whereas Spark Streaming has been utilized for this purpose in our research. We also incorporate, in addition to the annotations examined in the work (Patni, 2011), other annotations from the IoT domain of air quality monitoring, such as AQI index, air pollution level, and health consequences, and their interpretation is done in real-time. Our environment requires only 0.9 seconds to process 5,000 observations, unlike the 200-second processing time necessary to process 1,104 observations on an undefined hardware (Patni, 2011). Our new IoTSAS system performs better than the previous one, according to these results.

Unlike the required time over 200 seconds to process 1,104 observed WSN data on an unspecified hardware (Patni, 2011), the IoTSAS system requires only 0.9 seconds to process 5,000 number of observations in our environment. Based on these results, we may conclude that the developed IoTSAS system provides a better performance.

Table 12. IoTSAS system vs existing system (Patni, 2011) - performance comparison

	Paper: Real-Time Semantic Analysis of Sensor Streams	Our IoTSAS system
Hardware	N/A	CPU x5570 @ 2.93GHz (4 CPUs), Intel® Xeon® ~2.9GHz
Number of Observations	1,104	5,000
Average processing time (seconds)	> 200 s	0.9 s

In addition to the system performance tests presented in this paper, statistics that show the time required for each type of semantic annotations are also presented. From Table 13, it can be seen that for the *#AQI_Index* annotation, the average time is 42813 nanoseconds, for the *#MaxParam* annotation, 17615 nanoseconds are needed, for the *#Air_Pollution_Level* annotation, 16448 nanoseconds are needed, for the *#Health_Implications* annotation, 13765 nanoseconds are needed, for *#Rain_Shower* annotation 1056 nanoseconds are needed, for *#Rain_Storm* annotation 1399 nanoseconds are needed, for *#Flurry* annotation, 25159 nanoseconds are needed, and for *#Blizzard* annotation, 25564 nanoseconds are needed. From Figure 51, it can be seen that the *#Rain_Shower* annotation requires the minimum processing time, while the maximum processing time requires the *#AQI_Index* annotation.

Table 13. Results of semantic annotations performance test

Annotation	**Test 1 (nanoseconds)**	**Test 2 (nanoseconds)**	**Test 3 (nanoseconds)**	**AVG (nanoseconds)**
#AQI_Index	45146	45146	38147	42813
#MaxParam	17849	17849	17148	17615
#Air_Pollution_Level	18548	14699	16099	16448
#Health_Implications	13999	18899	8399	13765
#Rain_Shower	986	1050	1132	1056
#Rain_Storm	1049	1750	1399	1399
#Flurry	24548	25148	25781	25159
#Blizzard	25548	26597	24546	25564

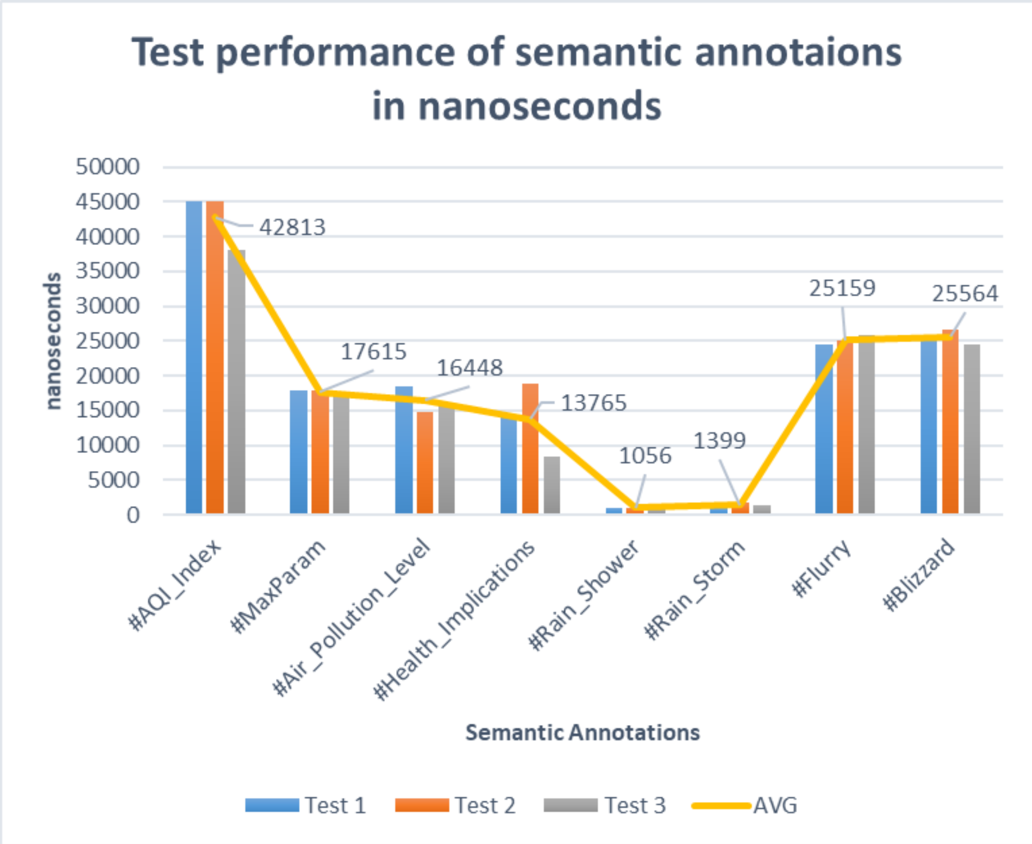


Figure 51. Test performance of semantic annotations in nanoseconds

Part IV.

Conclusion and Future Work

7
Chapter

7. Conclusion and Future Work

At the moment, billions of networked Internet of Things devices produce and exchange trillions of gigabytes of sensor data. Due to the variety of issues associated with the integration of sensor data collected by heterogeneous devices, the Internet of Things has sparked the interest of a sizable number of researchers in this subject.

The Internet of Things largely relies on sensors. A continuous stream of data, referred as observed WSN data or sensor stream data, is transmitted to a remote server for processing. Unless appropriately annotated, raw observed WSN data is of no use. By integrating semantic annotations with concept definitions from ontologies, observed WSN data may be interpreted and understood.

A real-time integration and interpretation of semantic annotations into the observed WSN data of different WSNs types with context in the IoT, is provided in this dissertation. First, are described the fundamentals of IoT, such as: IoT data transmission models, IoT applications, Wireless Sensor Networks (WSNs), sensor stream data, and semantic annotations. Next, a SLR related to the semantics integrated into the observed data of different WSN types is presented, which can be used by other academics to compare their technique to the existing ones. The review is carried out in accordance with the steps defined by (Petersen, 2008). First, the research questions are formulated, then a search strategy is devised, and ultimately, inclusion and exclusion criteria are established. The translation of the review's goal into research questions, which includes the use of SSW technologies and the primary solutions for integrating semantic annotations into observed WSN data, SW standards, stream processing models in real-time, and the semantic IoT trend domains, is presented as the first and most critical step in literature review.

Furthermore, annotating techniques for real-time integrating and interpreting semantics into heterogeneous observed WSN data with context in the IoT have been introduced. Spark Streaming, Kafka, and Cassandra DB, as well as SOS standards, are some of the technologies being used in this context.

Devise techniques consist of the main components such as, *Input Data Stream*, *Real-Time Detection of Outliers*, *Real-Time Semantic Annotation (RTSA), Real-Time Interpreting Semantically Annotated* (*RTISA*), *Ad-hoc requests*, *IoT domains rules*, with concept definitions from semantic sources (for example ontologies), that provide the understanding and more meaningful descriptions, allowing the IoT applications to become quite intelligent. To manage the data modelling of the processed sensor stream data with their semantic annotations is introduced a management model that comprises *WSNs Meta data*, *Invalid Data Streams*, *Working Data Streams*, *Archival Data Streams*, *Working Data Stream Annotations*, *Processor Data Streams,* and *Archival Data Stream Annotations*.

To implement the proposed annotating techniques for real-time integrating and interpreting of semantic annotations into heterogeneous observed WSN data with context in the IoT, an integrated system called IoTSAS (IoT Semantic Annotations System) is built. It consists of the following modules: (a) real-time integrating and interpreting of semantic annotations into the observed WSN data module, (b) module for managing metadata, (c) monitoring module for air quality and (d) weather warnings, (e) water quality monitoring module, (f) data modelling module, (g) module for external systems - RESTful APIs.

The WSN stream data from the World AQI API as well as WSN stream data from the InWaterSense project are used to demonstrate the validity of IoTSAS and the suggested system architecture. Finally, a WSNs stream data simulator is created to evaluate the IoTSAS's performance. According to the findings of the performance tests, the IoTSAS system only took 138 seconds to analyze the 1,000,000 sensor observations data by annotating with semantics and interpreting the semantic annotations, demonstrating the veracity of the excellent system performance.

For future work is left:

- To more advanced annotation techniques like XPath annotations to integrate and analyze semantic annotations in real time into observed WSN data and meta data in the IoT.
- To create a module that illustrates a healthcare monitoring use case, which will allow clinicians to monitor their patients in real time and notify them of changes in their health state.
- To extend the suggested system architecture for supporting the insertion of sensors with an XML request utilizing SWE standard as well as the SOS standard v2.0.
- To improved Outlier Stream Validator and Classifier components of the proposed model by implementing advanced outlier detection methods for real-time unsupervised anomaly identification.

8

Chapter

8. Bibliography

Abadi, D. J. et al. (2003). Aurora: a new model and architecture for data stream management. *The International Journal on Very Large Data Bases*, Springer-Verlag New York, Vol. 12, Issue 2, August, pp. 120-139.

Ahmedi, F., Ahmedi, L., O'Flynn, B., Kurti, A., Tahirsylaj, S., Bytyçi, E., Sejdiu, B., & Salihu, A. (2018). InWaterSense: An Intelligent Wireless Sensor Network for Monitoring Surface Water Quality to a River in Kosovo. *In P. Papajorgji, & F. Pinet (Ed.), Innovations and Trends in Environmental and Agricultural Informatics* (pp. 58-85). IGI Global. http://doi:10.4018/978-1-5225-5978-8.ch003

Ahmedi, L., Jajaga, E., & Ahmedi, F. (2013, October). An ontology framework for water quality management. In *Proceedings of the 6th International Workshop on Semantic Sensor Networks-Volume 1063* (pp. 35-50). CEUR-WS. org.

Ahmedi, L., Sejdiu, B., Bytyçi, E., & Ahmedi, F. (2015). An Integrated Web Portal for Water Quality Monitoring through Wireless Sensor Networks. *International Journal of Web Portals (IJWP)*, 7(1), 28-46.

Ahn, W. S., Kim, K. H., & Yoo, S. W. (2014). Study on robustness middleware using integrating Sensor Observation Service in Sensor Web Enablement. *2010 The 12th International Conference on Advanced Communication Technology (ICACT)*, Phoenix Park, pp. 4.

Akidau, T., et. al. (2015, August). The dataflow model: a practical approach to balancing correctness, latency, and cost in massive-scale, unbounded, out-of-order data processing. Proceedings of the VLDB Endowment - *Proceedings of the 41st International Conference on Very Large Data Bases*, Kohala Coast, Volume 8 Issue 12.

Akyildiz, I. & Vuran, M. (2010). Wireless Sensor Networks. *John Wiley & Sons, Inc*.

Ali, Sh., Khusro, Sh., Ullah, I., Khan, A., Khan, I. (2017). SmartOntoSensor: Ontology for Semantic Interpretation of Smartphone Sensors Data for Context-Aware Applications. *Journal of Sensors*, vol. 2017, Article ID 8790198, 26 pages.

Ali, S. (2016). Spectrum of the Smart World: IoT Overview and Commuincation Models. *Department of Networks and Communication Security, Institute of Space Technology*.

Aquino, A. Figueiredo, C. Nakamura, E. Buriol, S. Loureiro, F. Fernandes, O., and Coelho, N.(2006). Data Stream Algorithms For Processing of Wireless Sensor Network Application Data.

Arockiam, L. and Vijayalakshmi, A. (2016). A Study on Security Issues and Challenges in IoT", International Journal of Engineering Sciences & Management Research, November.

Arrieta, M. and Andres, C. (2021). Design and implementation of a semantic interface for service discovery in IoT. *Universidad de Ios Andes, Colombia*.

Atzori, L., Iera, A., Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, Vol. 54, Issue 15, pp. 2787-2805.

Auger, A., Exposito, E. & Lochin, E. (2017). Survey on Quality of Observation within Sensor Web systems," in IET Wireless Sensor Systems, vol. 7, no. 6, pp. 163-177, 12.

Babitski, G., Bergweiler, S., Hoffmann, J., Schön, D., Stasch, C., Walkowski A.C. (2009). Ontology-Based Integration of Sensor Web Services in Disaster Management. In *International Conference on GeoSpatial Semantics. GeoS 2009*. Lecture Notes in Computer Science, vol 5892. Springer, Berlin, Heidelberg.

Bahga, A. and Madisetti, V. (2014). Internet of Things: A Hands-On Approach. Vijay Madisetti, ISBN: 0996025529.

Bakaraniya, P. & Mehta, S. (2012, November). Features of WSN and Various Routing Techniques for WSN: A Survey. *International Journal of Research in Engineering and Technology (IJRET)*, Vol. 01, Issue 03.

Balakrishna S., Thirumaran M., Solanki V.K. (2020). IoT Sensor Data Integration in Healthcare using Semantics and Machine Learning Approaches. *In: Balas V., Solanki V., Kumar R., Ahad M. (eds) A Handbook of Internet of Things in Biomedical and Cyber Physical System. Intelligent Systems Reference Library*, vol 165. Springer, Cham. https://doi.org/10.1007/978-3-030-23983-1_11

Barnaghi, P., Wang, W., Henson, C., & Taylor, K. (2012). Semantics for the Internet of Things: Early Progress and Back to the Future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1), 1-21.

Begum, K. & Dixit, S. (2016). Industrial WSN using IoT: A survey. *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, pp. 499-504.

Bendadouche, R., Roussey, et al. (2012). Extension of the semantic sensor network ontology for wireless sensor networks: The stimulus-WSNnode-communication pattern. In *5th International Workshop on Semantic Sensor Networks in conjunction with the 11th International Semantic Web Conference (ISWC)* (pp. 16-p).

Bera, S., Misra, S., Roy, & Obaidat, M. S. (2016, November). Soft-WSN: Software-Defined WSN Management System for IoT Applications. *IEEE Systems Journal*, Vol. PP, Issue 99.

Botts, M., Percivall, G., Reed, C., & Davidson, J. (2007). OGC® sensor web enablement: Overview and high level architecture. *In International conference on GeoSensor Networks* (pp. 175-190). Springer Berlin.

Bröring, A., Echterhoff. J., Jirka, S., Simonis, I., Everding, T., Stasch, C., Liang, S. (2011). New Generation Sensor Web Enablement. Lemmens R. *Sensors (Basel)*, 11(3):2652-99.

Bröring, A., Stasch, C., & Echterhoff, J. (2012). OGC sensor observation service interface standard. *Open Geospatial Consortium Interface Standard*, 12-006.

Bytyçi, E., Sejdiu, B., Avdiu, A., & Ahmedi, L. (2017). SEMDPA: A Semantic Web Crossroad Architecture for WSNs in the Internet of Things. *International Journal on Semantic Web and Information Systems (IJSWIS)*, Vol. 13, no. 3, pp. 1-21, 2017.

Calbimonte, J. (2010). Enabling Semantic Integration of Streaming Data Sources. *Ontology Engineering Group, Departamento de Inteligencia Artificial*, Facultad de Inform´atica, Universidad Polit´ecnica de Madrid,

Calbimonte, J. (2013).Ontology-based access to sensor data streams. *A PhD thesis* under the supervision Oscar Corcho, Universidad Politecnica de Madrid.

Calbimonte, J. P., Jeung, H., Corcho, O., & Aberer, K. (2011, October). Semantic sensor data search in a large-scale federated sensor network. In *Proceedings of the 4th International Conference on Semantic Sensor Networks*-Volume 839 (pp. 23-38). CEUR-WS. org.

Calcaterra, D., Cavallo, M., Di Modica, G., & Tomarchio, O. (2016). Modelling and Monitoring Environmental Risks through a Semantic Framework. *International Journal of Distributed Systems and Technologies (IJDST)*, 7(4), 1-21.

Canada. (2020). How to use the public weather forecast. https://www.canada.ca/en/environment-climate-change/services/types-weather-forecasts-use/public.html, Accessed 15 December 2020.

Carbone, A., Katsifodimos, A., Ewen, et al. (2016). Apache Flink TM: Stream and Batch Processing in a Single Engine Paris. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 36(4).

Chandrasekaran, S., Cooper, O., Deshpande, A. et al. (2003, Jun). TelegraphCQ: Continuous Dataflow Processing for an Uncertain World. *SIGMOD '03 Proceedings of the 2003 ACM SIGMOD International Conference on Management of data*, San Diego, California, pp. 668-668.

Chen, J., DeWitt, D. J., Tian, F., & Wang, Y. (2000, June). NiagaraCQ: A Scalable Continuous Query System for Internet Databases. *SIGMOD '00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, Vol. 29, Issue 2, pp. 379-390.

Chen, L., et al. (2009). Semantic Smart Homes: Towards Knowledge Rich Assisted Living Environments. In *Intelligent Patient Management*. Studies in Computational Intelligence, vol 189. Springer, Berlin, Heidelberg

Chen, X., Chen, H., et al. (2015). Elastic Streaming Semantic Engine for Web of Things. *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, Singapore, pp. 305-308.

Chinnachodteeranun, R., & Honda, R. (2016). Sensor Observation Service API for Providing Gridded Climate Data to Agricultural Applications. *Future Internet*, Vol. 8, No. 1.

Cisco (2016). At a Glance for the Internet of Things Solution. Report C45-731471-01 06/16. https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/at-a-glance-c45-731471.pdf, Accessed 13 December 2019.

Compton, M, Barnaghi, P., Bermudez, P. & et al. (2012). The SSN ontology of the W3C semantic sensor network incubator group. *Web Semantics: Science, Services and Agents on the World Wide Web*, Volume 17, pp. 25-32.

Compton, M., Henson, C., Lefort, L, Neuhaus, H., & Sheth, A. (2009). A Survey of the Semantic Specification of Sensors. *Proc. Semantic Sensor Networks 2009*.

Conover, H., Goodman, H. M., Zavodsky, B., Regner, K., Maskey, M., Lu, J., Li, X., Botts, M. & Berthiau, G. (2008). Intelligent Assimilation of Satellite Data into a Forecast Model Using Sensor Web Processes and Protocols. In *Earth Science Technology Conference*.

Corcho, Ó., & Garcia-Castro, R. (2010). Five challenges for the Semantic Sensor Web. *Semantic Web*, vol. 1, no. 1,2, pp. 121-125

Corcho, O., Calbimonte J., Jeung, H., Aberer, K. (2012). Enabling Query Technologies for the Semantic Sensor Web. *International Journal on Semantic Web & Information Systems*, v.8 n.1, p.43-63.

Cox, S. (2011). Observations and measurements-xml implementation. *OGC document*.

Dean, J., & Ghemawat, S. (2008, January). MapReduce: simplified data processing on large clusters. *Communications of the ACM - 50th anniversary issue: 1958*, 1, 107-113.

Demers, A., Gehrke, J., Rajaraman, R., Trigoni, N., & Yao, Y. (2003). The Cougar Project: A Work-In-Progress Report. *ACM SIGMOD Record*, Vol. 32, pp. 53-59.

Dhalla, H. K. (2020). A Performance Comparison of RESTful Applications Implemented in Spring Boot Java and MS.NET Core. Virtual Conference on Engineering, Science and Technology (ViCEST) 2020 Kuala Lumpur, Malaysia.

Djenouri, Y., Belhadi, H., Akli-Astouati, K., Cano, A. Chun-Wei Lin, J. (2021). An ontology matching approach for semantic modeling: A case study in smart cities. *Computational Intelligence,* 2021, Wiley online library. 1-27.

Echterhoff, J. (2011) Open Geospatial Consortium. OpenGIS SWE Service Model Implementation Standard. *Open Geospatial Consoritum*, 155.

Elnahrawy, E. (2003). Research Directions in Sensor Data Streams: Solutions and Challenges. Rutgers University, *Tech. Rep. DCIS-TR-527*.

Elsaleh, T.; Enshaeifar, S.; Rezvani, R.; Acton, S.T.; Janeiko, V.; Bermudez-Edo, M (2020). IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams and Its Use with Data Analytics and Event Detection Services. Sensors 2020, 953.

Feng, T., Zhuang, Zh., Pan, Y., & Ramachandra, H. (2015). A memory capacity model for high performing data-filtering applications in Samza framework. In *Proceedings of the 2015 IEEE International Conference on Big Data (Big Data)* (BIG DATA '15). IEEE Computer Society, Washington, DC, USA, 2600-2605

Fensel, A.; Tomic, S.; Kumar, V.; Stefanovic, M.; Aleshin, S.V.; Novikov, D.O. (2013). SESAME-S: Semantic Smart Home System for Energy Efficiency. *Informatik-Spektrum*, 36, 46–57.

Gama, J. and Gaber, M. M. (2007). Learning from Data Streams - Processing Techniques in Sensor Networks. *Springer-Verlag Berlin Heidelberg*.

Ge, Y., Liang, X., Zhou, Y. C., Pan, et al. (2016). Adaptive Analytic Service for Real-Time Internet of Things Applications. *2016 IEEE International Conference on Web Services (ICWS)*, San Francisco, CA, pp. 484-491.

Ghazal R, Malik AK, Raza B, Qadeer N, Qamar N, Bhatia S. (2021). Agent-Based Semantic Role Mining for Intelligent Access Control in Multi-Domain Collaborative Applications of Smart Cities. Sensors. 21(13):4253. https://doi.org/10.3390/s21134253

Golab, L. & Ozsu, M. T. (2003, June). Issues in Data Stream Management. *SIGMOD Record*, Vol. 32, No. 2.

Gorasiya, D. V. (2019). Comparison of Open-Source Data Stream Processing Engines: Spark Streaming, Flink and Storm. Technical Report, DOI: 10.13140/RG.2.2. 16747.49440.

Gonzalez, R., et al. (2017). AAL open source system for the monitoring and intelligent control of nursing homes. *IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, Calabria, pp. 84-89.

Gray, A.J.G.; Sadler, J.; Kit, O.; Kyzirakos, K.; Karpathiotakis, et al. (2011) A Semantic Sensor Web for Environmental Decision Support Applications. *Sensors* 2011, *11*, 8855-8887.

Gyrard, A., & Serrano, M. (2015). A Unified Semantic Engine for Internet of Things and Smart Cities: From Sensor Data to End-Users Applications. *2015 IEEE International Conference on Data Science and Data Intensive Systems*, Sydney, NSW, pp. 718-725.

Haller, A., Janowicz, K., Cox, S., Lefrançois, M., Taylor, K., Phuoc, D.L., Lieberman, J., García-Castro, R., Atkinson, R., and Stadler, C. (2018). The SOSA/SSN Ontology: A Joint W3C and OGC Standard Specifying the Semantics of Sensors, Observations, Actuation, and Sampling. *Semantic Web – Interoperability, Usability, Applicability an IOS Press Journal*.

Hammad, M. A., Mokbel, M. F., Ali, M. H., Aref, W. G., Catlin, A. C., Elmagarmid, A. K., Eltabakh, M., Elfeky, M. G. Ghanem, T. M., Gwadera, R., Ilyas, I. F. Marzouk, M., and Xiong, X. (2004). Nile: a query processing engine for data streams. *Proceedings. 20th International Conference on Data Engineering.*

Hanif, M., Yoon, H., Jang, S. & Lee, C. (2017). An adaptive SLA-based data flow mechanism for stream processing engines. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju, pp. 81-86.

Henson, C. (2014, January). A semantics-based approach to machine perception. By Cory Andrew Henson, with Prateek Jain as coordinator. *SIGWEB Newsl*. Winter, Article 3.

Henson, C. A., Pschorr, J. K., Sheth, A. P., & Thirunarayan K. (2009a). SemSOS: Semantic sensor Observation Service. *International Symposium on Collaborative Technologies and Systems*, Baltimore, MD, pp. 44-53.

Henson, C., Neuhaus, H., Sheth, A., Thirunarayan, K., & Buyya, R. (2009b). An ontological representation of time series observations on the semantic sensor web. In *1st International Workshop on the Semantic Sensor Web*.

Honti, G. M., Abonyi, J. (2019). A Review of Semantic Sensor Technologies in Internet of Things Architectures. *Complexity*, vol. 2019, Article ID 6473160. https://doi.org/10.1155/2019/6473160

Huang, X., Yi, J., Zhu, X., & Chen, S. (2016). A Semantic Approach with Decision Support for Safety Service in Smart Home Management. *Sensors*, 16(8), 1224.

Ikechukwu, M., & Okeke, F. (2018). Towards Realisation of Heterogeneous Earth-Observation Sensor Database Framework for the Sensor Observation Service based on PostGIS, CoRR *abs/1802.06183*.

Ivanov, T., & Taaffe, J. (2018). Exploratory Analysis of Spark Structured Streaming. In *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering (ICPE '18)*. ACM, New York, NY, USA, 141-146.

Jajaga, E. & Ahmedi, L. (2017a). C-SWRL: SWRL for Reasoning over Stream Data. *2017 IEEE 11th International Conference on Semantic Computing (ICSC)*, San Diego, CA, pp. 395-400.

Jajaga, E. & Ahmedi, L. (2017b). C-SWRL: A Unique Semantic Web Framework for Reasoning Over Stream Data. *International Journal of Semantic Computing (ICSC)*, vol. 11, no. 3, pp. 391-410.

Jajaga, E., Ahmedi, L., Ahmedi, F. (2015). An Expert System for Water Quality Monitoring Based on Ontology. In *Metadata and Semantics Research. MTSR 2015. Communications in Computer and Information Science*, vol 544. Springer, Cham.

Janowicz, K., Compton, M. (2010). The stimulus-sensor-observation ontology design pattern and its integration into the semantic sensor network ontology. In *3rd International Workshop on Semantic Sensor Networks*, vol. 668, CEUR-WS.

Ji, Ch., Liu, J., & Wang, X. (2014). Associating Semantic Sensor Web with Domain Ontology: The Way to Obtain Meaningful Sensor Data. *International Journal of u-and e-Service, Science and Technology*, Vol. 7, No.5, pp. 271-282.

Karau, H. & Warre, R. (2017). High Performance Spark: Best Practices for Scaling and Optimizing Apache Spark. *O'Reilly Media*.

Kaur, A. (2016). Comparative analysis of applications and protocols in WSNs. *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, pp. 3291-3295.

Kaushik, A. (2016). IOT-An Overview. International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 3.

Kenda, K.; Kazic, B.; Novak, E.; (2019b). Streaming Data Fusion for the Internet of Things. *Sensors 19*, no. 8: 1955. https://doi.org/10.3390/s19081955

Keßler, C., & Janowicz, K. (2010, November). Linking sensor data-why, to what, and how? In *Proceedings of the 3rd International Conference on Semantic Sensor Networks*-Volume 668 (pp. 48-63). CEUR-WS. org.

Khan, I., Jafrin, R., Errounda, F., Glitho, R., Crespi, N., Morrow, M., & Polakos, P. (2015). A Data Annotation Architecture for Semantic Applications in Virtualized Wireless Sensor Networks. *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM),* Ottawa, ON, pp. 27-35.

Kitagawa, H. and Watanabe, Y. (2007). Stream Data Management Based on Integration of a Stream Processing Engine and Databases. *2007 IFIP International Conference on Network and Parallel Computing Workshops (NPC 2007)*, Liaoning, pp. 18-22.

Koupaie, H., Ibrahim, S. & Hosseinkhani, J. (2013). Outlier Detection in Stream Data by Clustering Method. *International Journal of Advanced Computer Science and Information Technology (IJACSIT)*, Vol. 2, No. 3, 2013, Page: 25-34.

Krummenacher, R., Simperl, E., Nixon L. J. B., Cerizza D., & Valle, E. D. (2007). Enabling the European Patient Summary through Triplespaces. *Twentieth IEEE International Symposium on Computer-Based Medical Systems (CBMS'07)*, Maribor, pp. 319-324.

Lazarescu, M.T. (2017) Wireless Sensor Networks for the Internet of Things: Barriers and Synergies. In: *Keramidas G., Voros N., Hübner M. (eds) Components and Services for IoT Platforms*. Springer, Cham.

Lee, H. J. & Kim, H. S. (2015). eHealth Recommendation Service System Using Ontology and Case-Based Reasoning. *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, Chengdu, pp. 1108-1113.

Lee, Y. J., Trevathan, J., Atkinson, I., & Read, W. (2015). The Integration, Analysis and Visualization of Sensor Data from Dispersed Wireless Sensor Network Systems Using the SWE Framework. *Journal of Telecommunications and Information Technology*, (4), 86.

Lefort, L. (2009). Review of semantic enablement techniques used in geospatial and semantic standards for legacy and opportunistic mashups. *In Proceedings of the Fifth Australasian Ontology Workshop - Volume 112 (AOW '09)*, Vol. 112. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 17-26.

Le-Phuoc, D., Quoc, H. N. M., Parreira, J. X., & Hauswirth, M. (2011). The linked sensor middleware–connecting the real world and the semantic web. *Proceedings of the Semantic Web Challenge*, *152*.

Liefde, I., Vries, M. & Meijers, B. (2016, June). Exploring the use of the semantic web for discovering, retrieving and processing data from Sensor Observation Services. A master thesis under the joint supervision of Vries, M. and Meijers, B. Delft University of Technology, Netherlands

Markogianni, V., Anastasopoulou, E., Tsoupras, A., Dimitriou, E. (2018). Identification of Pollution Patterns and Sources in a Semi-Arid Urban Stream. *Journal of Ecological Engineering*, 19(2), 99-113. https://doi.org/10.12911/22998993/81784

Motwani, R., Widom, J., Arasu, A. et al. (2003). Query Processing, Resource Management, and Approximation in a Data Stream Management System. *Technical Report. Stanford InfoLab*. Publication Note: CIDR.

Muppavarapu, V., Ramesh, G., Gyrard, A., and Noura, M. (2021). Knowledge extraction using semantic similarity of concepts from Web of Things knowledge bases. Data & Knowledge Engineering. Volume 135, https://doi.org/10.1016/j.datak.2021.101923.

Na, A., & Priest M. (2007). Sensor Observation Service. *Open Geospatial Consortium Inc*. Reference number of this document: OGC 06-009r6. Implementation Standard.

Nabi, Z. (2016). Pro Spark Streaming, The Zen of Real-Time Analytics Using Apache Spark. *Apress*.

Nair, J., Shetty, S., & Shetty, S. (2018). Applying spark based machine learning model on streaming big data for health status prediction. *Computers & Electrical Engineering*, Volume 65.

NASA, NASA Earth Observatory. Available online: https://earthobservatory.nasa.gov (accessed on 04/08/2021).

Osta, M., Ahmed, B., Abdelouahed, G. (2017). A Lightweight Semantic Web-based Approach for Data Annotation on IoT Gateways. *The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks*, Procedia Computer Science, Vol. 113, pp. 186-193.

Parveen, N. & Ananth, M. (2017). Data processing for large database using feature selection. *2017 2nd International Conference on Computing and Communications Technologies (ICCCT)*, Chennai, pp. 321-326.

Patni, H.K., Henson, C.A., Cooney, M.; Sheth, A. Demonstration: Real-Time Semantic Analysis of Sensor Streams. Proc. 4th Int. Workshop Semant. Sens. Netw. 2011, 11, 119–122.

Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M. (2008). Systematic mapping studies in software engineering. *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, 71-80

Petrolo, R., Loscrì, V., & Mitton, N. (2017). Towards a smart city based on cloud of things, a survey on the smart city vision and paradigms. *Emerging Telecommunications Technologies* 28(1).

Ploennigs, J., Schumann, A., Lécué, F. (2014) Adapting Semantic Sensor Networks for Smart Building Diagnosis. In *The Semantic Web – ISWC 2014. ISWC 2014*. Lecture Notes in Computer Science, vol 8797. Springer, Cham.

Pradilla, J., Palau C., & Esteve, M. (2016). SOSLITE: Lightweight Sensor Observation Service (SOS) for the Internet of Things (IOT). *ITU Kaleidoscope: Trust in the Information Society*, Barcelona.

Pschorr, J. K. (2013). SemSOS: an Architecture for Query, Insertion, and Discovery for Semantic Sensor Networks. *Master thesis* under the supervision Krishnaprasad Thirunarayan, Wright State University.

Pu, F., Wang, Z., Du, C., Zhang, W., & Chen, N. (2016). Semantic integration of wireless sensor networks into open geospatial consortium sensor observation service to access and share environmental monitoring systems. In *IET Software*, vol. 10, no. 2, pp. 45-53, 4.

Puiu, D. *et al*. (2016). CityPulse: Large Scale Data Analytics Framework for Smart Cities. In *IEEE Access*, vol. 4, pp. 1086-1108.

Rajaraman, A., Leskovec, J., & Ullman, J. D. (2014, November). Mining of Massive Datasets. *Cambridge University Press*.

Regueiro, M., Viqueira, J., Stasch, C., Taboada, J. (2017). Semantic mediation of observation datasets through Sensor Observation Services. *Future Generation Computer Systems*, Vol. 67, pp. 47-56.

Rezvan, M., et al. (2015). Applying an innovative semantic sensor network model in internet of things. In *Proceedings of the International Conference on Information and Communication Technology Convergence*. pp. 324–328.

Rhayem, A., Mhiri, M.B.A., Gargouri, F. (2020). Semantic Web Technologies for the Internet of Things: Systematic Literature Review. Internet of Things, Volume 11, ISSN 2542-6605, https://doi.org/10.1016/j.iot.2020.100206.

Rio, J., Toma, D., Martnez, E., Reilly, T., Delory, E., Pearlman, J., Waldmann, C., & Jirka, S. (2017). A Sensor Web Architecture for Integrating Smart Oceanographic Sensors into the Semantic Sensor Web. In *IEEE Journal of Oceanic Engineering*. Vol. PP, Issue 99.

Rose, K., Eldridge, S., and Chapin, L. (2015). The Internet of Things: An Overview - Understanding the Issues and Challenges of a More Connected World. The Internet Society (ISOC).

Rouached, M., Baccar, S., & Abid, M. (2012, June). RESTful sensor web enablement services for wireless sensor networks. In *2012 IEEE Eighth World Congress on Services* (pp. 65-72). IEEE.

Rubi, J. N. S., Gondim, P. R. L. (2020). IoT-based platform for environment data sharing in smart cities. *International Journal of Communication Systems.*

Santhi, T., Prasad Rajendra, J., & Vijayalakshmi, Y. (2016, July). A review on the state of art of Internet of Things. *International Journal of Advanced Research in Computer and Communication Engineering*, Vol. 5, Issue 7.

Sheth, A., Henson, C., & Sahoo, S. (2008, July). Semantic Sensor Web. *IEEE Internet Computing 12*, 4, 78-83.

Shi, F., Li, Q., Zhu, T., Ning, H. (2018). A Survey of Data Semantization in Internet of Things. *Sensors* 18(1): 313.

Simonis, I., Lathouwer B., & Taylor T. (2016). Sensor Web Enablement (SWE) for citizen science. *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Beijing, pp. 3618-3620.

Sohraby, K., Minoli, D., & T. Znati. (2007). Wireless Sensor Networks - Technology, Protocols, and Applications, *John Wiley & Sons, Inc*.

Soldatos, J., Kefalakis, et al. (2015). Openiot: Open source internet-of-things in the cloud. In *Interoperability and open-source solutions for the internet of things* (pp. 13-25). Springer, Cham.

Sowmya, C., Naidu, C. D., Somineni, R. P., and Reddy, D. R. (2017). Implementation of Wireless Sensor Network for Real Time Overhead Tank Water Quality Monitoring. *2017 IEEE 7th International Advance Computing Conference (IACC),* Hyderabad, pp. 546-551.

Spark Apache 2.3.1 (2018). Spark Streaming Programming Guide, http://spark.apache.org/docs/latest/streaming-programming-guide.html, Accessed 10 February 2018.

Stasch, C., Proß, B., Gräler, B., et al. (2018). Coupling sensor observation services and web processing services for online geoprocessing in water dam monitoring. *International Journal of Digital Earth*, 11(1): pp. 64-78.

Suoa, H., Wana, J., Zoua, C., and Liua, J. (2012). Security in the Internet of Things: A Review. International Conference on Computer Science and Electronics Engineering.

Toshniwal, A., Taneja, S., Shukla, A. et al. (2014). Storm @ Twitter. *SIGMOD '14 Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pp. 147-156.

Tran, L, Fan, L., & SHahabi, C. (2016). Distance-based Outlier Detection in Data Streams. *Proceedings of the VLDB Endowment*, Vol. 9, No. 12.

Udayakumar, P., & Indhumathi, M. (2012). Semantic web based Sensor Planning Services (SPS) for Sensor Web Enablement (SWE). *arXiv preprint arXiv*:1207.5310.

Vannieuwenborg, F., Ongenae, F., & Demyttenaere, F. (2014). Techno-economic evaluation of an ontology-based nurse call system via discrete event simulations. *2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Natal, pp. 82-87.

Vera, D., Izquierdo, Á., Vercher, & J., Gómez, L., (2014). A Ubiquitous Sensor Network Platform for Integrating Smart Devices into the Semantic Sensor Web. *Sensors 2014*, 14, 10725-10752.

Vermesan, O. and Friess, P. (2014). Internet of Things - From Research and Innovation to Market Deployment. River Publishers, ISBN: 978-87-93102-94-1.

Vlachostergiou, A., Stratogiannis, G., Caridakis, G., Siolas, G., & Mylonas, P. (2016). User Adaptive and Context-Aware Smart Home Using Pervasive and Semantic Technologies. *Journal of Electrical and Computer Engineering*, vol. 2016, Article ID 4789803, 20 pages.

White, T., (2015). Hadoop: The Definitive Guide. *O'Reilly 4th edition*.

World Wide Web Consortium (W3C). (2010). Report Work on Semantic Markup, https://www.w3.org/2005/Incubator/ssn/wiki/Report_Work_on_Semantic_Markup, Accessed 20 February 2018.

Yinbiao, S. & Lee, K. (2014). Internet of Things: Wireless Sensor Networks. *White Paper, International Electrotechnical Commission (IEC), Switzerland*.

Zachila K., Kotis K., Dimara A., Ladikou S., Anagnostopoulos CN. (2021). Semantic Modeling of Trustworthy IoT Entities in Energy-Efficient Cultural Spaces. *In: Maglogiannis I., Macintyre J., Iliadis L. (eds) Artificial Intelligence Applications and Innovations. AIAI 2021 IFIP WG 12.5*

International Workshops. AIAI 2021. IFIP Advances in Information and Communication Technology, vol 628. Springer, Cham. https://doi.org/10.1007/978-3-030-79157-5_30

Zaharia, M, Das, T., Li, H., Hunter, T., Shenker, S., & Stoica, I. (2013). Discretized streams: fault-tolerant streaming computation at scale. *In Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles (SOSP '13)*. ACM, New York, NY, USA, 423-438.

Zhou, B. *et al*. (2018, March). Online Internet traffic monitoring system using spark streaming. In *Big Data Mining and Analytics*, vol. 1, no. 1, pp. 47-56.