

A unified ontology-based data integration approach for the internet of things

Ahmed Swar, Ghada Khoriba, Mohamed Belal

Computer Science Department, Faculty of Computers and Artificial Intelligence, Helwan University, Cairo, Egypt

Article Info

Article history:

Received Apr 30, 2021

Revised Jul 8, 2021

Accepted Jul 18, 2021

Keywords:

Data integration

Internet of things

Ontology-based

RDF

Resource description

Semantic web

ABSTRACT

Data integration enables combining data from various data sources in a standard format. Internet of things (IoT) applications use ontology approaches to provide a machine-understandable conceptualization of a domain. We propose a unified ontology schema approach to solve all IoT integration problems at once. The data unification layer maps data from different formats to data patterns based on the unified ontology model. This paper proposes a middleware consisting of an ontology-based approach that collects data from different devices. IoT middleware requires an additional semantic layer for cloud-based IoT platforms to build a schema for data generated from diverse sources. We tested the proposed model on real data consisting of approximately 160,000 readings from various sources in different formats like CSV, JSON, raw data, and XML. The data were collected through the file transfer protocol (FTP) and generated 960,000 resource description framework (RDF) triples. We evaluated the proposed approach by running different queries on different machines on SPARQL protocol and RDF query language (SPARQL) endpoints to check query processing time, validation of integration, and performance of the unified ontology model. The average response time for query execution on generated RDF triples on the three servers were approximately 0.144 seconds, 0.070 seconds, 0.062 seconds, respectively.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Ahmed Swar

Computer Science Department, Faculty of Computers and Artificial Intelligence, Helwan University, Helwan, Cairo Governorate, Egypt

Email: ahmed.swar@fci.helwan.edu.eg

1. INTRODUCTION

Kevin Ashton of Procter and Gamble has introduced the concept of the internet of things (IoT) in 1999. The IoT is the network of physical devices, sensors, cameras, radio frequency identification devices (RFIDs), vehicles, home appliances, actuators, and connections. The IoT allows these devices to exchange data using IoT communication protocols and paradigm [1]. IoT architecture [2] consists of three layers namely perception, network, and application. The perception layer consists of physical objects or things that are being controlled using sensors and actuators. The network layer is able to connect the IoT sensors and actuators to a fixed gateway and router through different kinds of wired and wireless or public communication networks such as (ZigBee, Wi-Fi, 2G, 3G, and 4G). The application layer is the top layer in IoT architecture. The application layer is responsible for receiving data from IoT sources to the cloud platform using IoT communication protocols and paradigms. The application layer represents and visualizes collected data from IoT sources to help users to make decisions in real-time and send actuation commands to actuators. There are many IoT applications in diverse domains, like intelligent transportation systems, smart homes, intelligent healthcare, industry, security, and smart grids. However, the IoT has generated heterogeneity problems by using diverse

communication protocols, software environments, and device firmware. Many IoT applications, like healthcare systems, agriculture, smart grids, and intelligent transportation systems, have generated heterogeneous data. IoT devices range from low-power capacities sensors to multi-core platforms [3]. In addition, IoT devices have an internet protocol (IP) address for internet connectivity to allow communication and data exchange with other devices and users. There were approximately 50 billion sources connected to the internet by 2020 [4]. Often several heterogeneous objects located in different places need to interconnect and communicate in various ways.

The problem of data integration is to integrate and present these heterogeneous data in a standard format for users. For example, semantic web technologies like resource description frameworks (RDF) [5] represent IoT data in a standard unified format. IoT devices generate heterogeneous data in diverse formats like XML, CSV, JSON, and other interactive formats, like images in JPEG format and video streaming from cameras. Data integration provides the ability to combine and integrate data from diverse sources in a standard format. An ontology [6] is a formal, explicit specification for shared conceptualization. The ontology provides a shared language to both differentiate and link concepts between the applications.

It is challenging to integrate IoT devices, as they are limited by their resources, such as the battery, low power radio communication, and internet protocols. Users of IoT applications have to find alternatives for communication protocols, such as IPv6 over low-power wireless personal area networks (6lowPAN), message queuing telemetry transport (MQTT), and constrained application protocol (CoAP) protocols. Although users have used CoAP and 6lowPAN to query data from recognized and known sensors, they do not use them on a large scale.

The combination of ontologies [7] provides interoperability with semantic support between applications. Data integration systems have used ontologies to create a machine-understandable format. The Semantic Web [8] is an addition to the current web, in which information is meaningful and generates knowledge from information using ontologies. Data generated from diverse sources can be semantically annotated by adding semantic tags to the raw data. Semantic annotation represents classes, properties, attributes, and relationships between object properties. It provides a unified ontology-based view of the data for the user. An ontology-based approach integrates data from different formats (CSV, JSON, and XML) and other communication protocols.

Furthermore, semantic annotation enables IoT applications to process generated heterogeneous data from diverse sensors in real-time applications automatically. It provides reasoning capabilities by adding inference rules and domain knowledge based on ontology models. The data generated is in an understandable format described in terms of properties and values using RDF triples, which describe the reading value, location, and type. SPARQL is a semantic RDF query language for databases. SPARQL protocol and RDF query language (SPARQL) selects and processes data stored in an RDF format based on sensor data. In the past [9], there have been attempts to create unified ontologies by adding semantic tags and meaningful connections between data sources and RDF models. Therefore, design middleware for data integration is essential for generating knowledge from the data collected from various devices.

In this paper, we propose a shared ontology and unified ontology schema. The shared ontology is a combination of multiple ontologies, like the sensor network model (SSN) [10], sensor, observation, sample, and actuation (SOSA) [11], geospatial (GEO) [12], quantity kinds and units (QU) [13], extensible observation ontology (OBOE) [14], and IoT-Lite [15].

In 2016, Keller *et al.* [16] combined heterogeneous data from air transportation systems. The semantic layers combine the heterogeneous data. The semantic technique is an ontology-based triple store. However, this approach did not perform well on a large scale and is not applicable to the real world. Sensors generate a large amount of data that must be integrated and manipulated in acceptable response time for real-time applications. In 2017, Rahimi and Hakimpour [17] proposed a cloud-computing-based framework for real-time storage and for analysing stored data from transportation applications. They evaluated the results using the OpenStreetMap technique on different platforms. Evaluation results showed that the dataset they used contained almost 5 million records. The rate of importing data is approximately 8 thousand records per second, and the rate of exporting data is almost 15 thousand records per second. In 2010, Patni *et al.* [18] proposed a framework that converts sensor data to RDF triples and connects to stored data through linked open data on the cloud. In 2017, Joseph *et al.* [19] proposed an IoT middleware architecture that contains an integrated framework for heterogeneous applications that enables communication between other applications in smart cities. The proposed architecture depends on centrally managed data centers.

In 2013, Zhou *et al.* [20] proposed a fusion algorithm for data collected from IoT devices based on partitioning. In 2011, Moraru *et al.* [21] proposed the SemSense architecture that collects and publishes data from the sensor from one source. The SemSense architecture uses manual mapping of sensor data in the MYSQL database with SSN ontology concepts. In 2012, Le-Phuoc *et al.* [22] developed linked stream middleware (LSM) that collects, publishes, and annotation sensor data using cloud-based infrastructure. LSM

uses semantic sensor network ontology and relates data to available resources on the linked open data cloud. The shared ontology provides conceptual knowledge for heterogeneous data from various sources. There are three main types of ontologies such as single, multiple, and hybrid as shown in Figure 1.

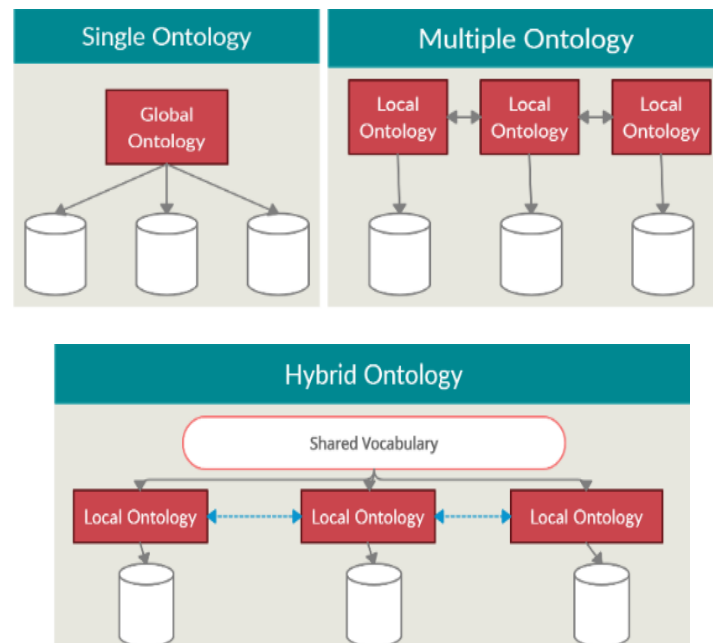


Figure 1. Mapping ontology using a shared ontology

In 2012, Elsaleh *et al.* [23] is a multi-layer framework that combines sources data in linked data form, provides access to IoT applications, and uses SPARQL endpoints. In 2017, Sarnovsky *et al.* [24] proposed integration by collecting heterogeneous data from diverse sources. They used Apache Nifi and designed workflows for processing real-time data from different locations. In 2019, Caballero *et al.* [25] proposed a Web-based middleware to manage the data from diverse sources. This middleware is based on the existing technologies to achieve interoperability and reusability. In 2019, Rahman and Hussain, [26] proposed a fog-based semantic model to exploit interoperability by migrating some commonly used cloud services to the fog to reduce task execution time and energy consumption. In 2019, Kim *et al.* [27] proposed a semantic web-based plug and play device management model in IoT by using SPARQL queries and semantics web technologies to exploit interoperability. In 2018, Guo *et al.* [28] proposed an artificial intelligence based semantic IoT to combine data from different devices to provide smart decisions in the Smart Cities as IoT application. In 2019, Venceslau *et al.* [29] described a survey in IoT semantic interoperability and focused on a systematic mapping study. In 2018, Skarmeta *et al.* [30] proposed the IoTcrawler framework that extracts metadata from the data sources. The framework sends annotated metadata to the application layer and stores the annotated metadata in an RDF metadata repository. In 2018, Giacomo *et al.* [31] proposed a semantic data integration model based on the global schema on ontologies to enhance query processing in big data and reduce the processing time in large-scale data. In 2019, Alshehab *et al.* [32] proposed a shared semantic integration model for E-government to enhance the shared knowledge between the citizens. This shared model lacks updating with the variations in government service type leads to ineffective sharing.

In 2019, Jayaratne *et al.* [33] proposed a data integration platform for patient-centred to combine the heterogeneous data from clinical sources and enhance the clinical decision making for different stakeholders. Hence, the aim of this research is to: i) construct a new semantic integration layer to model heterogeneous data collected from diverse sources; ii) propose a unified ontology schema approach based on a combination of ontologies and semantic web technologies to solve all IoT integration problems at once. The data unification layer maps fetched real data from IoT sources such as (temperature sensor, motion sensor, linear heat detection, traffic light controller, TrafiCam FLIR camera) in different formats like (like CSV, JSON, raw data, XML) to data patterns based on the unified ontology model; iii) add a semantic layer for cloud based IoT platforms to build a schema for data generated from diverse sources; iv) provide reasoning capabilities by adding inference rules and domain knowledge based on ontology models; v) develop a demo IoT application for testing the middleware in real-time. The demo used all layers of the ontology model; and vi) evaluate the proposed

approach by running different queries on different machines on SPARQL endpoints to check query processing time, validation of integration, and performance of the unified ontology model. To our knowledge, the heterogeneity issue remains a challenge. The problem is to provide a unified ontology semantic layer for heterogeneous data from multiple sources in real-time applications.

The rest of this paper is structured as follows. Section 2 presents our proposed architecture and implementation processes. Section 3 presents the results, discussion, and evaluation of the proposed model. Section 4 concludes the paper. Eventually, Section 5 discusses our future work objectives.

2. ONTOLOGY-BASED APPROACH

This paper introduces the middleware layer that collects data from heterogeneous sources, like sensors, cameras, and radio frequency identification devices (RFIDs). The aim of our research is to construct a new semantic integration layer to model heterogeneous data collected from diverse sources. Coordinating real-time traffic is a highly challenging problem. We use a combination of existing ontologies to clarify the concept of smart traffic. The integration of data from heterogeneous sources must be in a standard format. The proposed middle layer integrates data from local sources. The proposed model uses semantic web technologies and provides a uniform interface to the user for heterogeneous data from various devices, as shown in Figure 2.

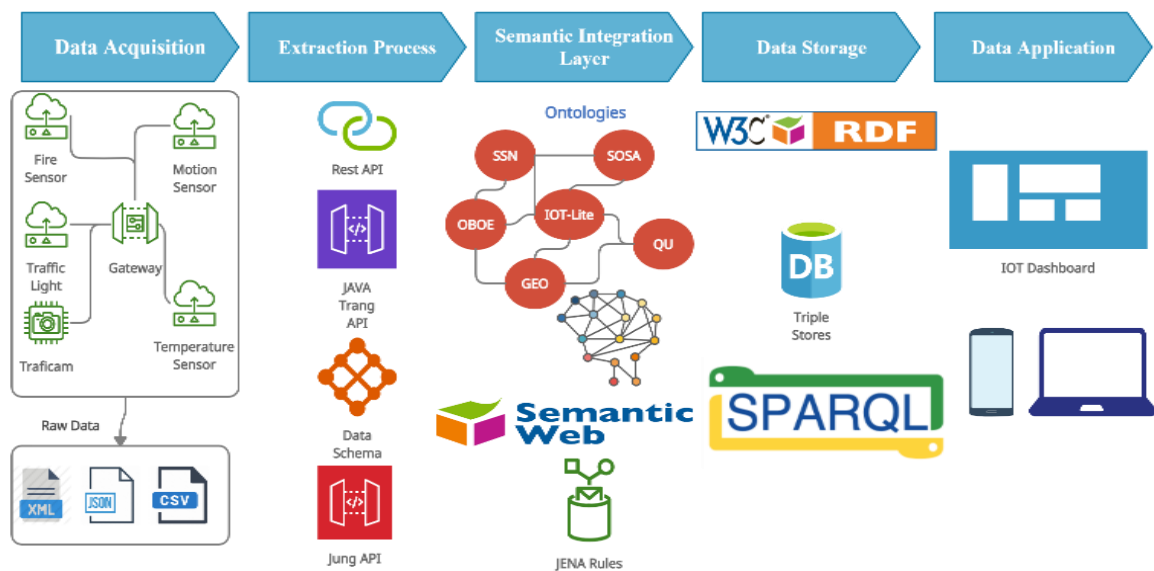


Figure 2. Data integration middleware in the IoT

Data integration is required to convert data to a standard format and combine data from various devices (roadside sensors, traffic lights, and cameras). The model allows the user to gain knowledge from sources of data and make decisions in real-time applications. An additional aim of this paper is to enhance the ontology model by extracting more knowledge and information from the data collected from diverse sources. The proposed model consists of five layers, as described in subsection 2.1 to 2.5.

2.1. Information gathering layer

This layer is concerned with collecting data from heterogeneous sources, such as traffic insights, inroads, roadside sensors, RFID, and cameras, in diverse formats, such as CSV, XML, JSON, text, image, and video. This layer provides wrappers for heterogeneous data in different formats to obtain data in a standard format.

2.2. Extraction process layer

After data is collected from various sensors in the first layer, a RESTful API web service that we developed sends the data to the extraction process, which extracts data schema from the raw data using JAVA Trang API.

2.3. Semantic integration layer

After generating the data schema using JAVA Trang API, we analyse the data schema. We use the output generated from the analysis in the JUNG framework to create a data schema graph. We use JENA API to generate OWL entities like classes, attributes, object properties, and data type properties. In the semantic integration layer, the captured data from various devices is annotated using a combination of existing ontologies like SSN, OBOE, GEO, QU, and IoT-lite. A combination of ontologies provides unified data models for various data inputs. We designed a lightweight data model to decrease traffic in the network and manipulate data with an acceptable response time for real-time applications.

Semantic annotation of data can be done by adding semantic tags to raw data captured from various sources to represent classes, properties, data object properties, and relations based on existing ontologies. It generates a set of information about the data, according to the source data, in a standard format. This allows information to be processed automatically in a standard way from heterogeneous data. SSN ontology is an ontology that contains classes and properties for sensors and observations of sensors. GEO ontology is an ontology for describing the location of a sensor device that consists of longitude and latitude. QU ontology is an ontology for describing quantity kinds and units for sensors. IoT-Lite ontology is a lightweight ontology for describing diverse devices of the internet of things, entities, and services. SOSA ontology is an ontology that consists of classes such as (actuator, sensor, platform, feature of interest, observation), and properties. After applying ontologies as shown in Figure 3 to the data model, we need to add logical inference rules to the inference engine to derive implicit logical concepts and new facts about source data.

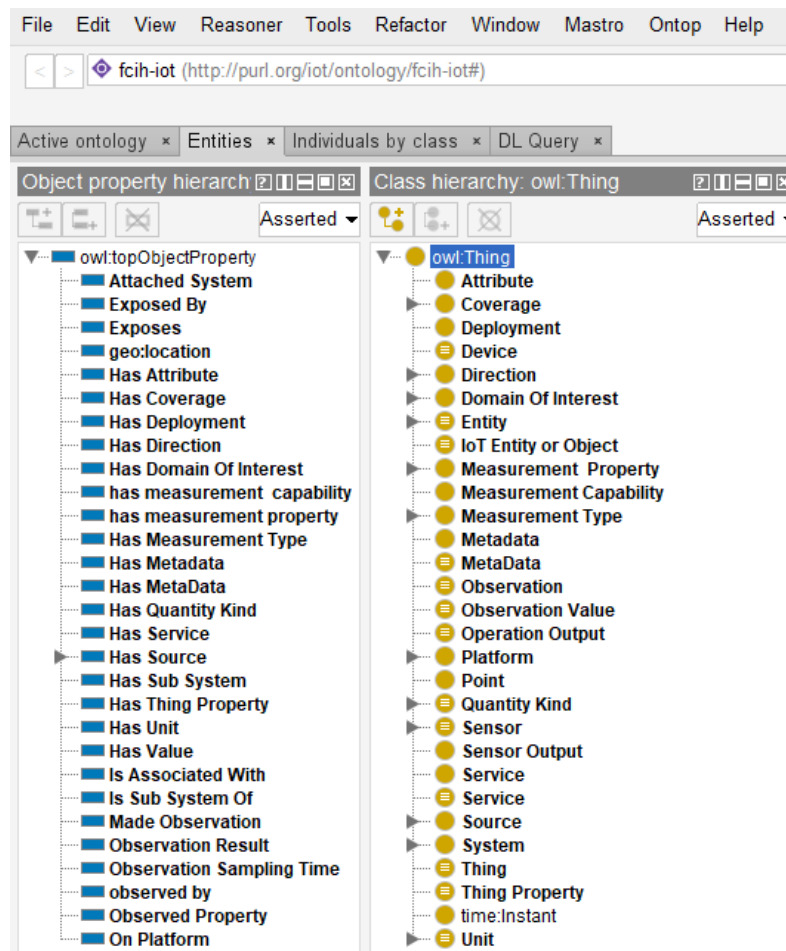


Figure 3. The core classes and object properties in shared ontology schema

2.4. Data saving layer

This layer consists of storing annotated data semantically. After applying the semantic integration layer, we generate the RDF turtle and integrated data. It contains RDF triple stores generated from the semantic annotation model. This layer also contains databases that provide support for the RDF format to store RDF

triples and can retrieve data from them. The stored data must then be retrieved from the RDF triple files. We use SPARQL endpoints to make queries about stored annotated data in RDF triples. SPARQL is an RDF query language for saving data in databases. It allows access to heterogeneous data with a unified format stored in triple stores.

2.5. Data application layer

By using SPARQL endpoints to execute queries, users can access data in various IoT applications, like intelligent transportation systems (ITS), and healthcare. RESTful API must be used to transfer data generated from SPARQL to the IoT application dashboard. The IoT application uses combined data from diverse sources in one view to provide usability for the user in real time.

We used programming technologies like JAVA, PHP, SPARQL, Apache Fuseki, OWL, Protege, Python, and MySQL to create a database management server. We developed a demo application for testing the middleware. The demo used all layers of the ontology model.

Building IoT middleware is challenging because of the many different environments and types of hardware and communication protocols involved. We used the FIWARE platform to combine data from various devices in various formats and store it in the server automatically for every new reading. The demo application can combine heterogeneous data from diverse sources.

Source data is collected by the FIWARE platform in the information gathering layer as raw data and stored in a server. We developed a demo IoT application that uses vocabularies and classes from ontologies like SSN, SOSA, OBOE, IoT-Lite, GEO, and QU. GEO ontology is used for the location of the device, and QU ontology is used for quantities and units. The middleware provides a data model by applying a semantic integration layer to transform data into a unified view. We used the MYSQL database for storing readings continuously from various sources. We also used the Apache JENA Fuseki server to store RDF triples in SPARQL endpoints. The proposed middleware annotates data to store in RDF triples, then publishes it in SPARQL Endpoints using vocabularies from existing ontologies. RDF triples contain URIs that identify the device, location, and reading value. The model uses some devices installed in the Azhar Tunnel as IoT devices and acquires heterogeneous data from diverse sources, like temperature sensors, motion sensors, heating detection sensors, traffic light controllers, and TrafiCam FLIR cameras, as shown in Table 1. These devices generate data in different formats, such as CSV, XML, and JSON.

Table 1. Connected data sources

Device	Generated Format	Data Acquisition	No. of Readings
Temperature sensor	XML	FTP	10000
Motion Sensor	CSV	FTP	30000
Linear Heating Detection	JSON	FTP	65000
Traffic Light Controller	JSON	TCP/IP	35000
TrafiCam FLIR Camera	CSV	TCP/IP	20000

3. RESULTS AND DISCUSSION

3.1. Implementation and experimental results

We prepared a windows-based operating system and set up deep learning environments. We evaluated the ontology model using a FIWARE platform and server running on different machines. The development machine contains an IoT application, a JAVA application that watches the files in the folder, and a semantic integration layer. After the middleware runs on the machine, it generates RDF triples, stores data in triple stores, and inserts data in the MYSQL database. We publish TTL files in SPARQL endpoints and make a RESTful API to view the data in a demo application. Table 2 shows variously connected sources, including a number of readings from connected sources in different formats (XML, JSON, and CSV). Table 2 also shows the number of RDF triples generated per format and per number of readings, the RDF dump file size for every format, and the MYSQL database size per number of readings and format. XML format achieved the best response time on three servers compared to JSON and CSV format because the response time for extracting data schema from XML files is faster than other file formats. The JSON format is lightweight, easy to read and transmission from sensors to the server could be faster than other file formats because less data is transferred. CSV format consumes less memory, is more secure, and is more compact than other file formats but does not support hierarchies of data. The temperature sensor achieved the best response time on three servers because the temperature sensor has generated XML files through FTP protocol.

Import combination of ontology libraries in SPARQL queries as prefixes.

- Initialize prefix for predicate with `<http://www.fcih.net/ >`
- Define namespace for SSN terms from SSN Ontology repository `< http://www.w3.org/ns/ssn/ >`

- Define prefix for OBOE ontology <http://ecoinformatics.org/oboe/oboe.1.0/oboe-core.owl#>
- Define prefix for Iolite ontology < http://purl.oclc.org/NET/UNIS/fiware/iot – lite# >
- Define prefix for GEO ontology < http://www.opengis.net/ont/geosparql# >
- Define prefix for QU ontology < http://purl.oclc.org/NET/ssnx/qu/qu# >
- Define prefix for DUL ontology <http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#>
- Define prefix for SOSA ontology < http://www.w3.org/ns/sosa/>

Table 2. Connected data sources formats

Data Acquisition Formats	XML	CSV	JSON
No. of reading from connected sources	10,000	50,000	100,000
No. of RDF Triples	60,000	300,000	600,000
RDF dump file size	3.6 MB	18 MB	36 MB
MYSQL Database size	0.9 MB	4.5 MB	9 MB

3.2. Evaluation

This research aims to enhance the accuracy of integration and interoperability. Instead of using a simulated dataset, we used real data from the Azhar Tunnel to test the data integration middleware. We used SPARQL queries to evaluate the performance of the middleware, running the middleware on three different machines. This provides an overview of the data integration performance. We used SPARQL queries in an IoT application developed with PHP to query from the MYSQL database and SPARQL endpoints. We developed sample SPARQL queries to assess the performance of the whole application in real time. The SPARQL queries are described below.

We consider the operating system, CPU type, and generation in choosing three different machines to test and run the IoT application. The specifications of machines are described as the following:

- Server 1 with Windows 10 64-bit operating system, 8 GB RAM, and Intel Core i7 2.60 GHz CPU.
- Server 2 with Windows 10 64-bit operating system, 16 GB RAM, and Intel Core i7-8565U 1.80GHz CPU.
- Server 3 with Windows 7 64-bit operating system, 16 GB RAM, and Intel Xeon 3.30 GHz CPU.

The data collected from sources is transferred automatically to the FIWARE platform. After the semantic integration layer is executed, RDF triples are generated, with full information according to the connected sources, in a TTL format. Sample queries were executed on the Apache Jena Fuseki server to check query processing performance and test the accuracy of retrieving data from SPARQL endpoints for use in the IoT application dashboard. We developed the JAVA program to use SPARQL queries and measured both the performance of the whole application in real time and the measured response time of the queries in seconds. Table 3 describes sample SPARQL queries to assess the performance of the whole application in real time.

Table 3. Sample of SPARQL queries

#	Objective	Syntax
Q1	Select the last reading value of temperature sensor	SELECT ?Value ?Timestamp WHERE{{ ?sensor_reading a ssn:Observation; ssn:observedBy ?sensor; ssn:observationResultTime ?Timestamp; ssn:hasValue ?Value. } { ?sensor a ssn:Sensor; iotlite:hasQuantityKind ?property. } FILTER (?property = "Temperature")} ORDER BY DESC (?Timestamp) LIMIT 1
Q2	Retrieve reading values in specific location of sensor	SELECT ?Value ?Timestamp WHERE { ?sensor_reading a ssn:Observation; ssn:observedProperty %%; ssn:observedBy ?sensor; ssn:hasValue ?Value; ssn:observationResultTime ?Timestamp FILTER (?Timestamp < "2020-07-15" && ?Timestamp > "2020-07-01"). ?sensor iotlite:hasSensingDevice ?sensing_device. ?sensing_device geo:lat %%; ?sensing_device geo:long %%. } LIMIT 25
Q3	Retrieve number of reading values of specific sensor	SELECT ?sensor_property COUNT(?Sensor_Reading) WHERE{ { ?Sensor_Reading ssn:observedBy ?sensor. } { ?sensor a ssn:Sensor; iotlite:hasQuantityKind ?sensor_property. } FILTER (?sensor_property = "Temperature")} GROUP BY ?sensor_property ORDER BY DESC (COUNT(?Sensor_Reading))
Q4	Retrieve number of sensors in specific location using GPS coordinates (longitude, latitude)	SELECTCOUNT (DISTINCT(?Sensor_Device)) WHERE { ?Sensor_Device a ssn:Sensor; ?Device geo:long "%%"^^xsd:double; geo:lat "%%"^^xsd:double. iotlite:hasQuantityKind %%; iotlite:hasSensingDevice ?Device. }
Q5	Retrieve Timestamp and location of heating detection sensor when fire is on	SELECT ?Value ?Timestamp ?Lat ?Long WHERE{{ ?sensor_reading a ssn:Observation; ssn:observedBy ?sensor; ssn:observationResultTime ?Timestamp; ssn:hasValue ?Value. } { ?sensor a ssn:Sensor; iotlite:hasSensingDevice ?Device; iotlite: hasQuantityKind ?property. ?Device geo:lat ?Lat ?Device geo:long ?Long. } FILTER (?property = "Fire").FILTER (?Value = ON)

Figure 4 shows the technical diagram of IoT data integration middleware, which consists of four layers. The device layer contains IoT devices and is connected to the edge layer through the fixed gateway and router to monitor the IoT stream from any device connected to the Internet. The back-end layer stores raw data and processes data using semantic integration, then stores the knowledge in SPARQL endpoints. The front-end layer shows the web dashboard of the IoT application. We also tested the middleware on different machines with different capabilities to simulate different environments in the real world, as shown in Table 4. The table shows the execution of every query and the response time per storage size (RDF triples) in seconds by the query.

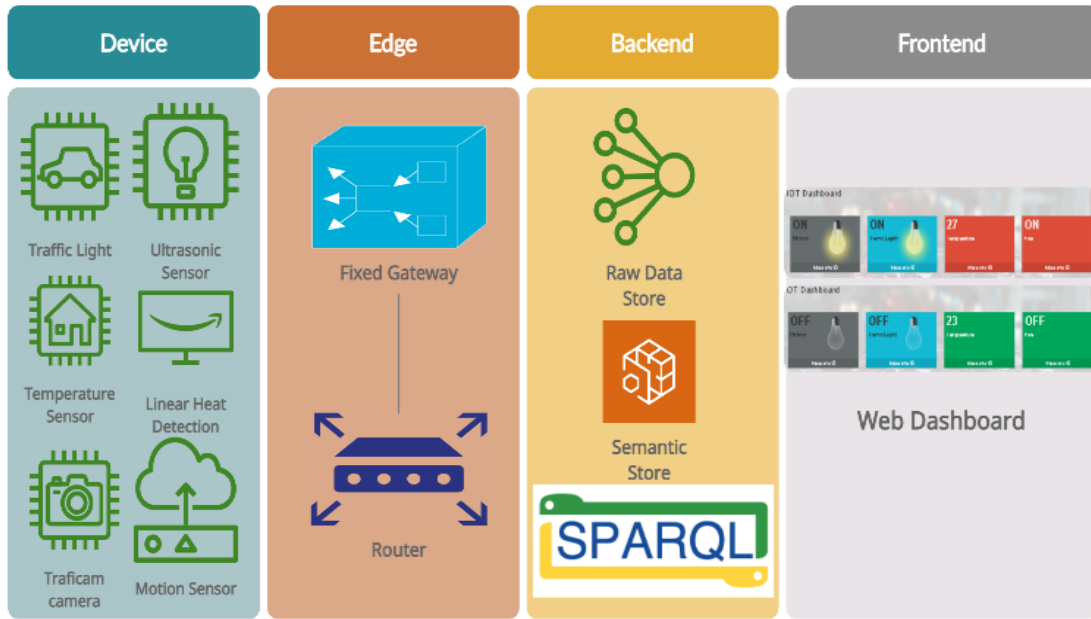


Figure 4. Technical diagram of IoT data integration middleware

Table 4. Running Middleware on three different machines

Server 1			Server 2			Server 3		
60 K	300 K	600 K	60 K	300 K	600 K	60 K	300 K	600 K
0.002	0.003	0.004	0.001	0.001	0.001	0.001	0.001	0.001
0.004	0.004	0.004	0.002	0.002	0.002	0.015	0.002	0.002
0.066	0.2	0.6	0.03	0.1	0.3	0.025	0.1	0.3
0.066	0.27	0.7	0.03	0.13	0.37	0.025	0.11	0.32
0.066	0.27	0.87	0.03	0.13	0.4	0.027	0.12	0.35
0.13	0.33	1.067	0.07	0.17	0.5	0.06	0.15	0.42
0.004	0.004	0.004	0.002	0.002	0.002	0.001	0.001	0.002
0.004	0.006	0.006	0.002	0.003	0.003	0.001	0.002	0.003
0.006	0.006	0.009	0.003	0.003	0.004	0.002	0.003	0.004
0.006	0.008	0.011	0.003	0.004	0.005	0.003	0.003	0.005
0.006	0.006	0.01	0.003	0.003	0.005	0.002	0.003	0.004

After testing the middleware on different machines, we noticed that the average response time of query execution on server 3 is the best performance compared to two servers because of the XEON CPU and generation on server 3. We tested the middleware on three different machines to evaluate the best performance in a real-time environment for high-traffic events and transactions. In the future, we will integrate interactive formats like images and videos in IoT applications such as intelligent transportation systems. Therefore, we need to run the model on a machine with high capabilities performance to visualize and analyse the integrated data from different sources in different formats in real-time to provide smart decisions. Figure 5 shows the experimental scenario for IoT data integration middleware, which consists of multiple layers as described in detail in the ontology-based approach section.

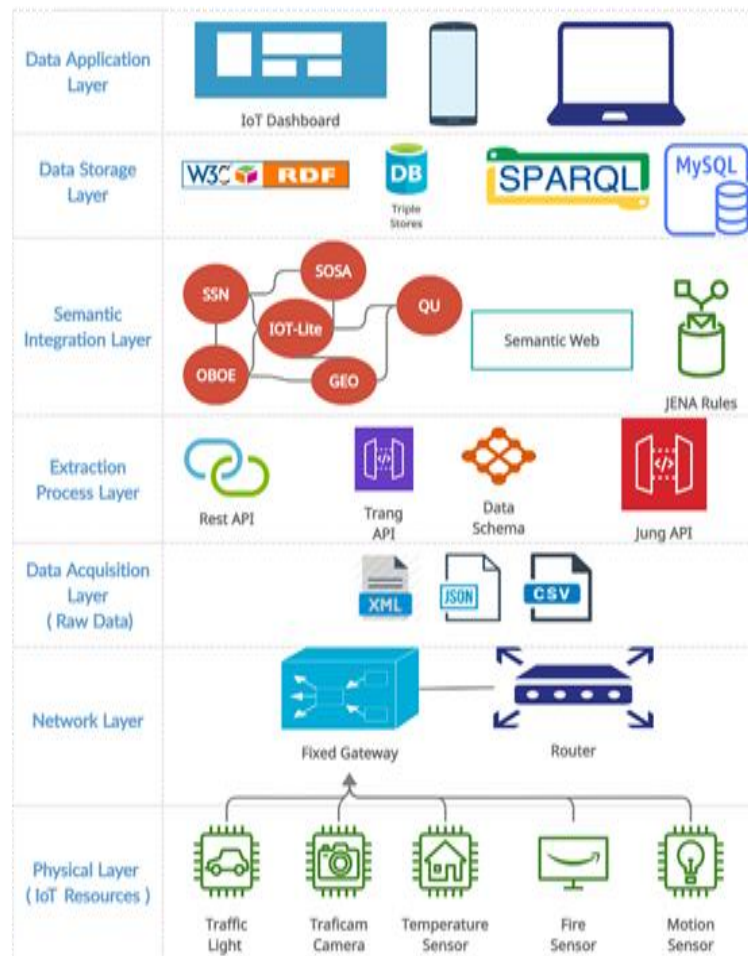


Figure 5. Experimental scenario for IoT data integration middleware

4. CONCLUSION

IoT devices generate heterogeneous data in various formats, like XML, CSV, JSON, and other interactive formats. Data integration allows data from diverse sources to be combined in a standard format. We used real data from the Azhar Tunnel in the proposed solution. The proposed middleware generated data in a unified format and displayed the data in a unified way. In the proposed model, we added a semantic integration layer containing a combination of existing ontologies, like SSN, SOSA, OBOE, IoT-Lite, QU, and GEO, and added inference rules to the inference engine. Then we generated RDF triples, stored them in triple stores, and published the data in SPARQL endpoints. We developed eleven SPARQL queries to assess the performance of the whole application in real-time. We also tested the middleware on different machines with different capabilities to evaluate the best performance in a real-time environment. We noticed that the average response time of query execution on server 3 is the best performance compared to two servers because of the XEON CPU and generation on server 3. Therefore, the average response time for query execution on generated RDF triples from JSON format on the three servers were approximately 0.298 seconds, 0.144 seconds, 0.128 seconds, respectively. In addition, the average response time for query execution on generated RDF triples from CSV format on the three servers were approximately 0.101 seconds, 0.0498 seconds, 0.045 seconds, respectively. Eventually, the average response time for query execution on generated RDF triples from XML format on the three servers were approximately 0.032 seconds, 0.016 seconds, 0.0147 seconds, respectively.

5. FUTURE WORK

Cars play a crucial role in an IoT application such as an intelligent transport system (ITS). This system has emerged in response to the need to reduce congestion, save time, reduce crashes, and reduce fuel consumption. The automated traffic control system for automatic plate recognition is an important factor of automated traffic monitoring. Automatic license plate recognition has become very necessary.

In the future, we will work on improving the model and applying it to interactive formats like images and videos in IoT applications such as intelligent transportation systems. This will help users to access heterogeneous data in middleware from various formats like CSV, XML, JSON, other interactive formats such as JPEG images, and video streaming from cameras.

ACKNOWLEDGEMENTS

Special thanks to the Al Azhar Tunnel Operation Authority for providing real data from various sensors and cameras.




REFERENCES

- [1] A. Ghasempour, "Internet of things in smart grid: architecture, applications, services, key technologies, and challenges," *Inventions*, vol. 4, no. 1, p. 22, Mar. 2019, doi: 10.3390/inventions4010022.
- [2] C. Zhong, Z. Zhu, and R.-G. Huang, "Study on the IOT architecture and access technology," in *2017 16th International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES)*, 2017, vol. 2018-Sept, pp. 113–116, doi: 10.1109/DCABES.2017.32.
- [3] S. Pasricha, R. Ayoub, M. Kishinevsky, S. K. Mandal, and U. Y. Ogras, "A survey on energy management for mobile and IoT Devices," *IEEE Design & Test*, vol. 37, no. 5, pp. 7–24, Oct. 2020, doi: 10.1109/MDAT.2020.2976669.
- [4] M. Burhan, R. Rehman, B. Khan, and B.-S. Kim, "IoT elements, layered architectures and security issues: a comprehensive survey," *Sensors*, vol. 18, no. 9, p. 2796, Aug. 2018, doi: 10.3390/s18092796.
- [5] R. Miarka, "RDF/JSON serialization of knowledge patterns," in *AIP Conference Proceedings*, 2019, vol. 060013, no. July, p. 060013, doi: 10.1063/1.5114048.
- [6] L. M. O. Machado, "Ontologies in knowledge organization," *Encyclopedia*, vol. 1, no. 1, pp. 144–151, Jan. 2021, doi: 10.3390/encyclopedia1010015.
- [7] M. Katsumi and M. Fox, "Ontologies for transportation research: A survey," *Transportation Research Part C: Emerging Technologies*, vol. 89, pp. 53–82, Apr. 2018, doi: 10.1016/j.trc.2018.01.023.
- [8] R. Burse, M. Bertolotto, D. O'Sullivan, and G. McArdle, "Semantic interoperability: the future of healthcare," in *Web Semantics*, Elsevier, 2021, pp. 31–53.
- [9] G. Bajaj, R. Agarwal, P. Singh, N. Georgantas, and V. Issarny, "A study of existing ontologies in the IoT-domain," *arXiv*, 2017.
- [10] K. Taylor *et al.*, "The semantic sensor network ontology, revamped," in *CEUR Workshop Proceedings*, 2019, vol. 2576, pp. 1–9.
- [11] K. Janowicz, A. Haller, S. J. D. Cox, D. Le Phuoc, and M. Lefrançois, "SOSA: A lightweight ontology for sensors, observations, samples, and actuators," *Journal of Web Semantics*, vol. 56, pp. 1–10, May 2019, doi: 10.1016/j.websem.2018.06.003.
- [12] S. Zhong, Z. Fang, M. Zhu, and Q. Huang, "A geo-ontology-based approach to decision-making in emergency management of meteorological disasters," *Natural Hazards*, vol. 89, no. 2, pp. 531–554, Nov. 2017, doi: 10.1007/s11069-017-2979-z.
- [13] N. Maksimov, A. Gavrilkina, V. Kuzmina, and E. Borodina, "Ontology of properties and its methods of use: properties and unit extraction from texts," *Procedia Computer Science*, vol. 169, pp. 70–75, 2020, doi: 10.1016/j.procs.2020.02.116.
- [14] I. Esnaola-Gonzalez, J. Bermúdez, I. Fernandez, and A. Arnaiz, "Ontologies for observations and actuations in buildings: A survey," *Semantic Web*, vol. 11, no. 4, pp. 593–621, Aug. 2020, doi: 10.3233/SW-200378.
- [15] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor, "IoT-Lite: a lightweight semantic model for the internet of things and its use with dynamic semantics," *Personal and Ubiquitous Computing*, vol. 21, no. 3, pp. 475–487, Jun. 2017, doi: 10.1007/s00779-017-1010-8.
- [16] R. M. Keller, S. Ranjan, M. Y. Wei, and M. M. Eshow, "Semantic representation and scale-up of integrated air traffic management data," in *Proceedings of the International Workshop on Semantic Big Data - SBD '16*, 2016, pp. 1–6, doi: 10.1145/2928294.2928296.
- [17] M. M. Rahimi and F. Hakimpour, "Towards a cloud based smart traffic management framework," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-4/W4, no. 4W4, pp. 447–453, Sep. 2017, doi: 10.5194/isprs-archives-XLII-4-W4-447-2017.
- [18] H. Patni, S. S. Sahoo, C. Henson, and A. Shetli, "Provenance aware linked sensor data," in *CEUR Workshop Proceedings*, 2010, vol. 576.
- [19] T. Joseph, R. Jenu, A. K. Assis, V. A. S. Kumar, P. M. Sasi, and G. Alexander, "IoT middleware for smart city: (An integrated and centrally managed IoT middleware for smart city)," in *2017 IEEE Region 10 Symposium (TENSYP)*, 2017, pp. 1–5, doi: 10.1109/TENCONSpring.2017.8070054.
- [20] J. Zhou, L. Hu, F. Wang, H. Lu, and K. Zhao, "An efficient multidimensional fusion algorithm for IoT data based on partitioning," *Tsinghua Science and Technology*, vol. 18, no. 4, pp. 369–378, Aug. 2013, doi: 10.1109/TST.2013.6574675.
- [21] A. Moraru, D. Mladenic, M. Vucnik, M. Porcius, C. Fortuna, and M. Mohorcic, "Exposing real world information for the web of things," in *Proceedings of the 8th International Workshop on Information Integration on the Web in conjunction with WWW 2011 - IIWeb '11*, 2011, pp. 1–6, doi: 10.1145/1982624.1982630.
- [22] D. Le-Phuoc, H. Q. Nguyen-Mau, J. X. Parreira, and M. Hauswirth, "A middleware framework for scalable management of linked streams," *Journal of Web Semantics*, vol. 16, pp. 42–51, Nov. 2012, doi: 10.1016/j.websem.2012.06.003.
- [23] S. De, T. Elsaleh, P. Barnaghi, and A. S. Meissner, "An internet of things platform for real-world and digital objects," *Scalable Computing*, vol. 13, no. 1, pp. 45–57, 2012, doi: 10.12694/scpe.v13i1.766.
- [24] M. Sarnovsky, P. Bednar, and M. Smatana, "Data integration in scalable data analytics platform for process industries," in *2017 IEEE 21st International Conference on Intelligent Engineering Systems (INES)*, 2017, vol. 2017-Janua, pp. 000187–000192, doi: 10.1109/INES.2017.8118553.
- [25] V. Caballero, S. Valbuena, D. Vernet, and A. Zaballos, "Ontology-defined middleware for internet of things architectures," *Sensors*, vol. 19, no. 5, p. 1163, Mar. 2019, doi: 10.3390/s19051163.
- [26] H. Rahman and M. I. Hussain, "Fog-based semantic model for supporting interoperability in IoT," *IET Communications*, vol. 13, no. 11, pp. 1651–1661, Jul. 2019, doi: 10.1049/iet-com.2018.6200.
- [27] W. Kim, H. Ko, H. Yun, J. Sung, S. Kim, and J. Nam, "A generic Internet of things (IoT) platform supporting plug-and-play device management based on the semantic web," *Journal of Ambient Intelligence and Humanized Computing*, Sep. 2019, doi:




- 10.1007/s12652-019-01464-2.
- [28] K. Guo, Y. Lu, H. Gao, and R. Cao, "Artificial intelligence-based semantic internet of things in a user-centric smart city," *Sensors*, vol. 18, no. 5, p. 1341, Apr. 2018, doi: 10.3390/s18051341.
- [29] A. Venceslau, R. Andrade, V. Vidal, T. Nogueira, and V. Pequeno, "IoT semantic interoperability: a systematic mapping study," in *Proceedings of the 21st International Conference on Enterprise Information Systems*, 2019, vol. 1, pp. 535–544, doi: 10.5220/0007732605350544.
- [30] A. F. Skarmeta *et al.*, "IoT-Crawler: browsing the internet of things," in *2018 Global Internet of Things Summit (GIoTS)*, 2018, pp. 1–6, doi: 10.1109/GIoTS.2018.8534528.
- [31] G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati, "Using ontologies for semantic data integration," in *A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*, Springer International Publishing, 2018, pp. 187–202.
- [32] A. Alshehab, N. N. Alazemi, and H. A. Alhakem, "Semantic integration sharing for e-government domains ontology: design and implementation using owl," *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 6, pp. 1820–1831, 2019.
- [33] M. Jayaratne *et al.*, "A data integration platform for patient-centered e-healthcare and clinical decision support," *Future Generation Computer Systems*, vol. 92, pp. 996–1008, Mar. 2019, doi: 10.1016/j.future.2018.07.061.

BIOGRAPHIES OF AUTHORS






Ahmed Swar    received the B.S. from the department of Computer Science of faculty of computers and artificial intelligence in 2013. He is currently worked at the faculty of computers and artificial intelligence as a teaching assistant. He is currently pursuing his MSc at the Department of Computer Science, Helwan University. His research interests include IoT, Data Integration, wireless ad hoc networks, and big data. Email: ahmed.swar@fci.helwan.edu.eg.



Ghada Khoriba    received a B.S. in Computer Science from Helwan University in 2000, and an M.S. from the Helwan University in 2014. She received her Ph.D. in Engineering Computer Science from the University of Tsukuba, Japan in 2010. Her research interests span both distributed systems, Artificial Intelligence, and Data Science. She is an Associate Professor in the Department of Computer Science at Helwan University, where I have been since 2000. Email: ghada_khoriba@fci.helwan.edu.eg.



Mohamed Belal    received a B.S. in Computer Engineering from Alexandria University in 1988, and an M.S. from faculty of Engineering in Cairo University in 1993. He received his Ph.D. in Computer Engineering from faculty of Engineering in Cairo University in 1998. He is a professor in computer science, specific domain: Computational Intelligence, since Nov. 2010. Email: belal@fci.helwan.edu.eg.