







Ontology-Driven IoT System for Monitoring Hypertension

Pedro Lopes de Souza^{1,2}^a, Wanderley Lopes de Souza^{1,2}^b, Luís Ferreira Pires¹^c,
João Luiz Rebelo Moreira¹^d, Ronitti Juner da Silva Rodrigues²^e and Ricardo Rodrigues Ciferri²^f

¹*Semantics, Cybersecurity & Services Group, Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, P.O. Box 217, Enschede, The Netherlands*

²*Ubiquitous Computing Group, Graduate Program in Computer Science, Computing Department, Federal University of São Carlos, P.O. Box 676, São Carlos, Brazil*

Keywords: Internet of Things, Ubiquitous Computing, Cloud Computing, Interoperability, Ontology, Healthcare.


Abstract: Hypertension is a noncommunicable disease (NCD) that causes global concern, high costs and a high number of deaths. Internet of Things, Ubiquitous Computing, and Cloud Computing enable the development of systems for remote and real-time monitoring of patients affected with NCDs like hypertension. This paper reports on a system for monitoring hypertension patients that was built by employing these techniques. This system allows the vital signs of a patient (blood pressure, heart rate, body temperature) to be captured via sensors built in a wearable device similar to a wristwatch. These signals are transmitted to the patient's mobile device for processing, and the generated clinical data are sent to the cloud to be properly presented and analysed by the health professionals responsible for the patient. To deal with semantic interoperability issues that arise when multiple different devices and system components must interoperate, a semantic model was conceived for this system in terms of ontologies for diseases and devices. This paper also presents the semantic module that we developed and implemented in the cloud to perform reasoning based on this model, demonstrating the potential benefits of incorporating semantic technologies in our system.


1 INTRODUCTION


Hypertension is a cardiovascular disease that affects 22% of the world's adult population over 18 years of age, being the cause of death for approximately 9.4 million people yearly. Diagnosis and monitoring of hypertension patients are necessary to maintain blood pressure within levels considered normal, typically 120 mmHg for systolic pressure and 80 mmHg for diastolic pressure in adults (WHO, 2013). However, this task is not always trivial, as for more assertive diagnosis and treatment, it is necessary to consider risk factors and correlate them with the disease progression data. In addition, blood pressure measurements are usually performed by health professionals during consultation and these data are


often stored in an unstructured way and are not always available when needed (Sandi, Nugraha and Supangkat, 2013).


This paper presents the System Based on Internet of Things for Monitoring Patients with Hypertension (SBIoT-MPH) (Rodrigues, 2022), which is a distributed system that allows timely and efficient monitoring of vital signs. SBIoT-MPH has been realised using IoT, Ubiquitous Computing and Cloud Computing technologies to automatically acquire and process clinical data from hypertension patients and make them available to health professionals. SBIoT-MPH allows patient vital signs to be captured and transmitted from the patient's mobile device to the cloud. This allows early warnings to be sent to the


^a <https://orcid.org/0000-0002-4538-7590>

^b <https://orcid.org/0000-0001-8645-4393>

^c <https://orcid.org/0000-0001-7432-7653>

^d <https://orcid.org/0000-0002-4547-7000>

^e <https://orcid.org/0000-0002-4589-7910>

^f <https://orcid.org/0000-0001-5944-8246>

patient, health professionals and other caretakers when critical situations are detected.

One requirement to be addressed by SBIoT-MPH is to transparently support a multitude of different devices. In addition, it should be extensible to cope with other diseases, which means that it should be able to handle heterogeneous data. Furthermore, once we decide to integrate our system with other systems (e.g., IoT platforms), the interoperability issues will get even more stringent. To represent these issues, we developed a semantic model based on ontologies for diseases and devices. This paper then also reports on the semantic module that we developed based on this semantic model and implemented in the cloud to support reasoning based on the collected data.

The objectives of this paper are twofold: (i) demonstrate how a system (hardware and software) can be developed to monitor vital signs and make clinical data available in the cloud for analysis and visualisation and (ii) demonstrate how to extend this system with other devices and applications with support of semantic technologies to cope with semantic interoperability issues.

The remaining of this paper has been structured as follows: Section 2 gives the background to this work; Section 3 introduces the SBIoT-MPH architecture in terms of its layers; Section 4 presents the SBIoT-MPH semantic model and describes the architecture and design of the semantic module; Section 5 discusses our results; Section 6 discusses related work; and Section 7 presents our conclusions and suggestions for future work.

2 BACKGROUND

Hypertension is a prevalent cardiovascular disease and is a strong risk factor for other diseases acquired during life, such as coronary heart disease, left ventricular hypertrophy, cardiac arrhythmias including atrial fibrillation, stroke and renal failure (Kjeldsen, 2018). Hypertension is characterised by sustained levels of systolic blood pressure greater than or equal to 140 mmHg, and diastolic blood pressure greater than or equal to 90 mmHg, is considered a silent killer, with symptoms not visible for many years, and is usually discovered when a complication occurs, or a vital organ is compromised (WHO, 2013).

For a more effective treatment of hypertension, periodic monitoring of the health conditions the patient is necessary. Continuous monitoring enables patients and their health professionals to actively exert disease control. This requires vital signals to be

periodically collected, via equipment or sensors, and the corresponding clinical data to be stored, treated, and made available for analysis.

Internet of Things (IoT), Ubiquitous Computing, and Cloud Computing can be effectively used in combination to build telemedicine systems aiming at providing pervasive healthcare (Husain et al, 2022). Wireless sensor networks and mobile devices have been used to develop mobile applications and smart environments. In this way, it is possible to increase the quality of health services and keep costs down, allowing for less direct but more effective interaction between patient and health professionals, and providing ubiquitous access to health services. Furthermore, data provided by different sensors can be combined to produce more complete reports, allowing more accurate medical diagnoses, and more effective treatments.

IoT technologies are heterogeneous in terms of hardware and software at different levels, with different data formats and semantics, different devices from different manufacturers, different wired and wireless networking technologies, and different communication protocols. Usually, IoT devices produce large amounts of data, which can be first sent to a gateway to be pre-processed, for example, in mobile devices, and then forwarded to the cloud for further processing (Gawanmeh, and Al-Karaki 2021).

IoT heterogeneity can lead to incorrect or inefficient use, in addition to making it difficult for users to access the offered services. To cope with these different levels of heterogeneity, one should strive for semantic interoperability, which aims at providing a common data representation for the meaningful data exchange between different applications and services. This encompasses the meaning of the data and their relationship, and requires common vocabularies to describe the data and ensure that data are unambiguously understood by the devices (Noura, Atiquzzaman and Gaedke 2018). Many semantic models and approaches were proposed recently for IoT semantic interoperability, most of them employing ontologies, middleware, semantic web, and knowledge management systems (Souza, Souza and Ciferri 2022).

3 SBIoT-MPH

Figure 1 presents an overview of SBIoT-MPH and its components, which have been structured according to their locations and functionality in three layers (Zhang and Zhang, 2012): Sensor Layer, Fog Layer and Cloud Layer.

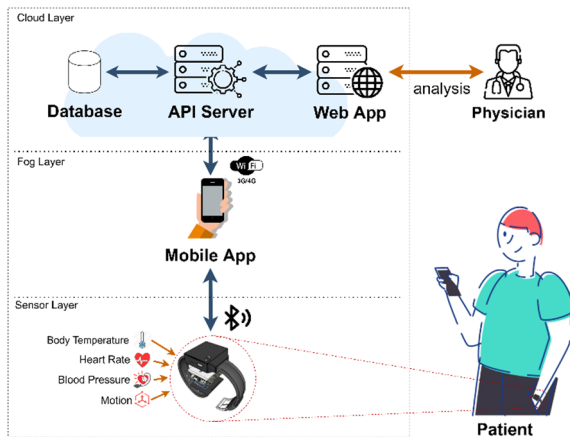


Figure 1: SBIoT-MPH overview.

The Sensor Layer encompasses wearable devices attached to the patient's body, which have different sensors responsible for capturing the patient's vital signs and making them available digitally. In its current implementation, this layer collects these vital signs and automatically transmits clinical data related to these signals to the Fog Layer via the Bluetooth Low Energy (BLE) standard. The Fog Layer contains the Mobile App application, which processes the received clinical data and forwards it to the Cloud Layer via WiFi or 3G/4G networks. The Cloud Layer contains the API Server and Web App applications. The API Server allows other applications to query and store system data, while the Web App provides a User Interface (UI) for health professionals to view and analyse patient clinical data and for system management.

In its current implementation, the Sensor Platform collects, analyses and stores blood pressure, heart rate and body temperature. However, the modular architecture of this system allows other vital signs to be captured, simply by adding new sensors.

3.1 Sensor Layer

The Sensor Layer contains a Sensor Platform, which consists of a Wireless Body Sensor Network (WBSN) built in a bracelet that the patient can wear on her wrist like a watch and captures the patient's vital signs. Clinical data obtained from these signals are transmitted via Bluetooth Low Energy (BLE) to an application that runs on the patient's mobile device (smartphone, or tablet). Figure 2 shows the main hardware modules of this platform.

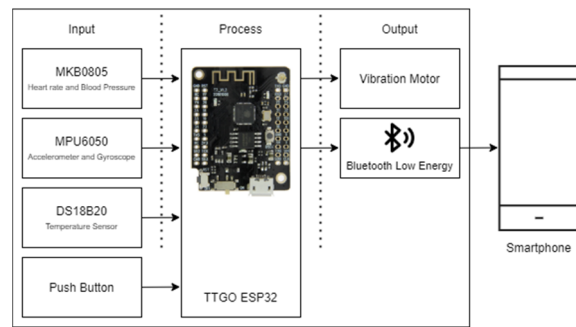


Figure 2: Hardware architecture of the Sensor Platform.

The MKB0805 module is a heart rate and blood pressure sensor that employs the Photoplethysmography (PPG) method to detect changes in blood volume in the microvascular tissue bed. This non-invasive and low-cost method applies a light source to the surface of the skin and measures the variations in light intensity caused by the absorption and reflection of this light by the skin tissue through a photodetector (Castaneda et al., 2018). Since these variations depend on the amount of blood present in the optical path, this sensor reads the signals and estimates systolic and diastolic pressures and heart rate, making these clinical data available to the main module.

The MPU6050 module is a sensor that measures the acceleration of an object according to three coordinate axes (X, Y, Z), and is employed to detect the patient movements necessary to enable the MKB0805 module to read clinical data. The DS18B20 module is a digital temperature sensor that measures temperatures between -55°C and 150°C with an accuracy of $\pm 0.5^{\circ}\text{C}$ and is used to measure the patient's body temperature.

The signals and data obtained by the sensors are processed by the TTGO T7 V1.3 MINI 32 module, a hardware board often used in the development of IoT systems. This board is equipped with a low cost and low energy ESP32 microcontroller from Espressif Systems and offers WiFi 802.11b and Bluetooth v4.2 connectivity. The platform has also the Vibration Motor module, which consists of a small motor capable of generating vibrations to alert the patient when critical situations are detected.

Figure 3 shows two photos of the Sensor Platform prototype: (a) the components of this prototype, which are fixed to a base structure to be inserted into the bracelet; and (b) the bracelet already fully assembled. The plastic frame was produced with a 3D printer from the 3D models that were designed using the FreeCad software.

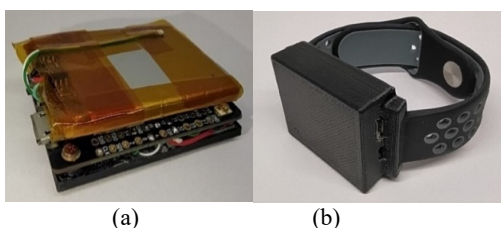


Figure 3: Sensor Platform prototype. (a) Prototype components; (b) Assembled prototype.

3.2 Fog Layer

Mobile App is an application we developed for Android devices that runs on the patient's smartphone or tablet and acts mainly as a gateway, receiving data from the Sensor Platform and forwarding it to the cloud. Two components interact with the Mobile App, namely the Patient Service and the Background Service. The Patient Service represents the hypertension patient and supports the following functionality:

- a) *Authentication*: the patient provides the previously registered username and password and once authorised, she can manage the sensors, and view her clinical data, messages, and risk alerts.
- b) *Sensors Management*: the patient registers the sensor platform and can check the connection status and battery level of the platform.
- c) *Data Visualisation*: the patient consults the data records of the last 24 hours, which are stored in a database on the patient's mobile device.

The Background Service runs in the background in the Android device, and provides the following functionality:

- a) *Physiological Data Collection*: once sensors are registered by the patient, the connection and authentication process of Mobile App with the Sensor Platform is triggered. After this connection is established, clinical data are collected, processed, and sent to the cloud, where they are stored and made available to health professionals.
- b) *Patient Data Analysis*: the received data are analysed and classified according to the recommendations of the American Heart Association shown in Table 1¹ into four health risk levels: NORISK; LOW, indicating that hypertension can be controlled with lifestyle changes, such as more physical activities and healthier eating habits; MODERATE, indicating hypertension needs to be controlled by medication, in addition to lifestyle changes; and HIGH, indicating a critical

situation so that the patient needs immediate intervention of a health professional;

- c) *Alerts Management*: alerts are generated according to the risk levels and sent to the cloud for analysis by a health professional, and the patient automatically receives notifications regarding these alerts on their mobile device and on the Sensor Platform. For MODERATE and HIGH risks, the health professional and an emergency contact immediately receive a text message with information related to the patient's health status; and
- d) *Messages Management*: messages recorded by the health professional on the Web App, such as recommendations for lifestyle changes, treatment adjustments involving changes in drug dosages, or the prescription of new drugs, are received and stored on the patient's mobile device, who in turn receives notifications related to these messages.

Table 1: Blood Pressure Category.

Category	Systolic mmHg		Diastolic mmHg
Normal	Less than 120	and	Less than 80
Elevated	120-129	and	Less than 80
Hypertension Stage 1	130-139	or	80-89
Hypertension Stage 2	140 or Higher	or	90 or Higher
Hypertension Crisis	Higher than 180	and/or	Higher than 120

Figure 4 shows two screenshots of the Mobile App User Interface (UI): (a) the interactive Data Collection screen, which allows the patient to view the sensors and update the data collected by each sensor, and allows the patient to access the screen for viewing the data history related to a sensor; (b) the Data History screen, which allows the patient to view the data history in a graph.

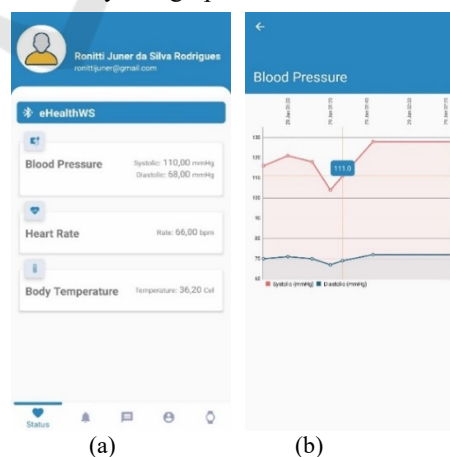


Figure 4: Screenshots of the Mobile App UI. (a) Data Collection screen; (b) Data History screen.

¹ <https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings>

Mobile App was implemented in Java using the API level 21 for Android 5.0 applications, compatible with around 94,1% of the Android-based devices. To safeguard the security of the data stored on the patient's mobile device, the SQLite library with the SQLCipher extension has been used, which provides a database with 256-bit AES encryption. The Apollo GraphQL library 2.4 was also used, which allows Java models for GraphQL queries to be generated (Jeon, Liuhaoyang and Hwang, 2019).

3.3 Cloud Layer

Web App and API Server were deployed in the Cloud Layer allowing cloud resources to be used on demand, as these applications perform tasks that require more computing resources. Cloud deployment also enables access to these applications via Internet. Web App is accessed via a Web browser, by health professionals to view and analyse their clinical data, and by an administrator to manage health professionals and patients.

The following functionality concerns the administrator:

- Patients Management*, to support the registration and maintenance of patient data, such as their personal data and the types of diseases to be monitored.
- Patient to Health Professional Assignment*, to assign a patient to a health professional who will be responsible for monitoring the patient's clinical data.
- Health Professionals Management*, to support the registration of health professionals.

The following functionality concerns the health professional:

- Health Data Analysis*, which allows the health professional to access their patients' clinical data, analyse these data and then make decisions regarding their treatment.
- Message Exchange*, which allows messages to be sent to the patient's mobile device via the Mobile App, with recommendations regarding that treatment or lifestyle. These messages can be categorised by the health professional into 3 priority levels for determining the order in which they are displayed to the patient.
- Alerts Monitoring*, which allows alerts to be automatically shown on the Mobile App once they are forwarded to the API Server, allowing health professionals to analyse clinical data.

Figure 5 shows a screenshot of the Web App UI, which allows the health professional to apply filters, such as the desired clinical data type and the analysis

period. To help the health professional in this analysis, reference lines related to pre-established normal values for each clinical data type are plotted.

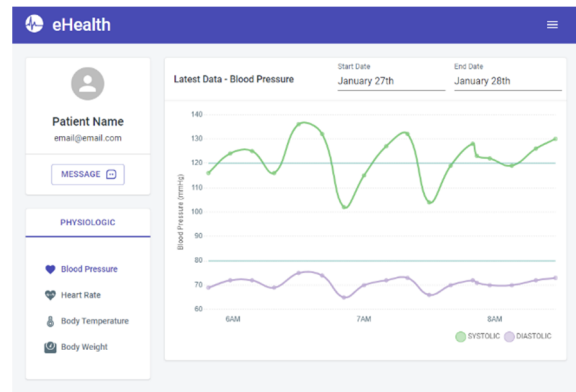


Figure 5: Screenshot of the Web App.

API Server operates as a server, providing a secure API so that Mobile App and Web App can store and retrieve system information. This API employs the GraphQL query and data manipulation language, which uses more parsimonious messages and therefore generates less network traffic than Representational State Transfer (REST).

Web App was implemented in JavaScript, and the React and Material UI libraries were used since they facilitate UI development. Integration with the API Server was carried out via the Apollo Client library, as it supports queries to a GraphQL API, managing the cache and automatically updating the UI data.

API Server was also implemented in JavaScript, and Node.js was used as a server-side runtime environment due to its improved performance when compared to others server technologies (Chitra and Satapathy, 2017). The Apollo Server open-source library version 2.16 was used since it supports GraphQL APIs compatible with any GraphQL client. API Server provides an authentication and authorization mechanism that uses JSON Web Token (JWT), supporting secure communication via the HTTPS protocol.

4 SEMANTIC MODULE

Figure 6 depicts the IoT semantic model that we developed based on a model that has been proposed in (Rahman and Hussain, 2020). In an IoT-based platform, components may not be able to properly exchange and understand the raw data generated by IoT devices (e.g., sensors, actuators, RFID devices) from different manufacturers due to the lack of

common semantics. To solve this problem, the first step can be to send the data to a gateway for pre-processing and aggregation, aiming to increase their quality via an algorithm such as the one presented in (Rahman, Ahmed and Hussain, 2018). These aggregated data can be then stored in the cloud so that semantic operations can be performed on them.

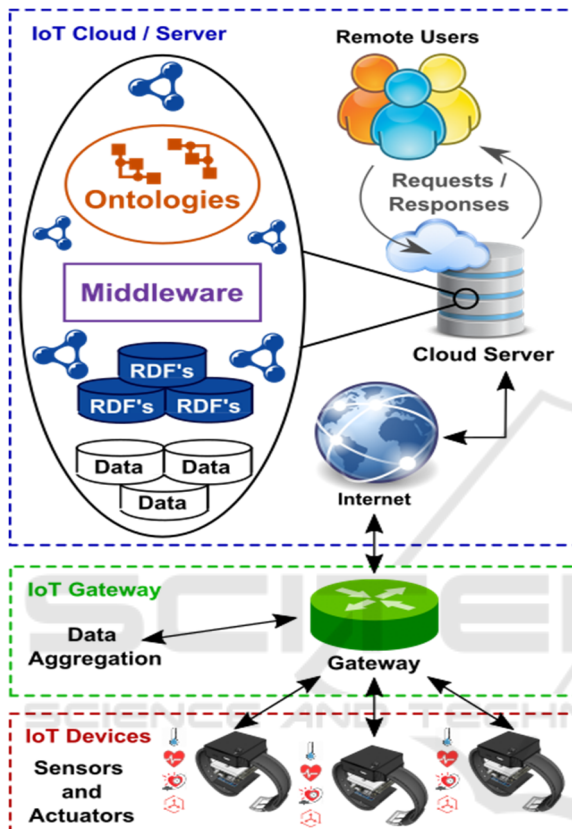


Figure 6: IoT semantic model (adapted from (Rahman and Hussain, 2020)).

In this model, all processing to achieve semantic interoperability is performed in the cloud, which must comprise ontologies to provide semantic annotations to the aggregated data, adapters to convert aggregated data into semantic data, a conventional database, and a triplestore, which is a database in RDF format. The aggregated data are converted to the RDF triples format with the help of ontologies. Some work has already been performed in this direction (Buneman and Staworko, 2016), but further investigation is still necessary.

The semantic module improves SBIoT-MPH by addressing the following requirements of each layer:

- a) *Sensor Layer*, where the sensors have different data formats based on different data types with different semantics, and it is necessary to predefine data according to BluetoothGatt API.

- b) *Fog Layer*, where the interpretation of the data received from the Sensor Layer, with respect to the hypertension risk levels, is restricted to the Patient Data Analysis functionality; and
- c) *Cloud Layer*, where data exchange is not meaningful, and it is not possible to personalise the Patient Data Analysis functionality.

Therefore, we adapted the semantic model shown in Figure 6 to deal with these problems. We worked mainly on the Cloud Layer, in order to add semantic annotations to the aggregated data received from the Fog Layer, as illustrated in Figure 7.

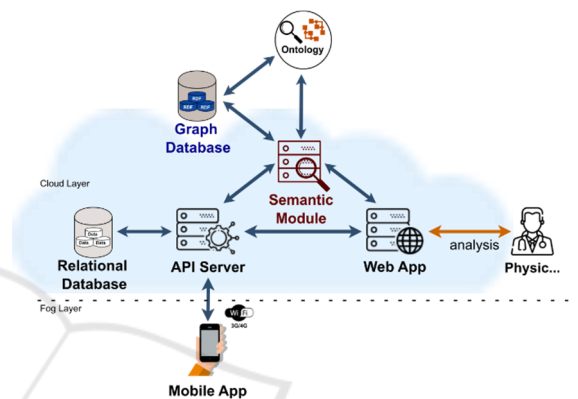


Figure 7: SBIoT-MPH semantic model.

When designing the SBIoT-MPH semantic model, we sought to keep as much as possible the original structure of the system. To this end, the semantic module was introduced in the cloud, communicating with the SBIoT-MPH applications, the RDF database and the ontologies, which are also implemented in the cloud.

Figure 8 shows the components of the semantic model, which are discussed in the sequel.

4.1 API Listener

This component is responsible for capturing aggregated data from API Server, transforming them into RDF format and publishing the result of this transformation. Initially, a GraphQL API has been employed to send to the API Listener all data in JSON format received by the API Server from the Mobile App. API Listener uses the GraphQL subscription operation, which acts like a publisher/subscriber protocol. Whenever API Server receives data from Mobile App, API Listener is notified and receives a copy of those data.

The next step has been to semantically enrich the aggregated data with RDF annotations, where additional knowledge expressed in ontologies. The SBIoT-MPH Semantic Module uses the

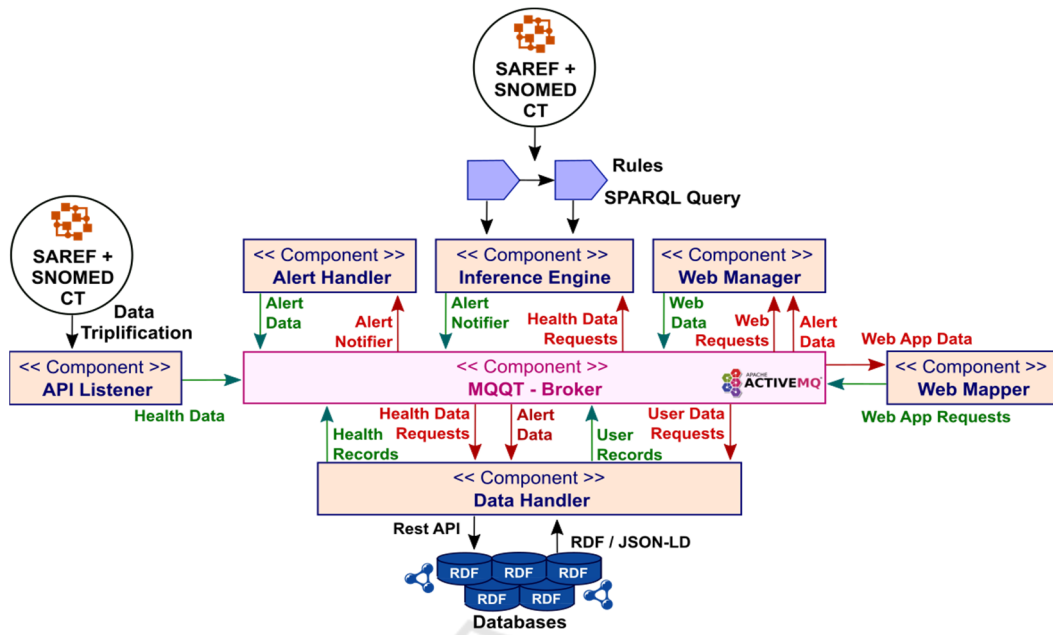


Figure 8: SBIoT-MPH Semantic Module architecture.

SAREF4EHAW ontology² to represent devices, sensors, data, and actors. To incorporate concepts and properties related to hypertension, the *Chronic disease* class of SAREF4EHAW was extended with a subset of the NCDs-related SNOMED CT ontology³.

A conversion algorithm transforms the aggregated data into a set of RDF triplets with the help of the ontology and a mapping file that employs a lightweight JSON-based mapping language. The relationship between the schema and the ontology is stored in this mapping file, which describes how to extract any information from the ontology (Rahman and Hussain, 2020). An MQTT client function connected to the broker publishes the semantically structured data produced by the conversion algorithm as a JSON-RD format to the *FogHeathData* topic. Figure 9 shows the main activities of API Listener.

4.2 Inference Engine

This component is responsible for real-time processing of the patient data published by API Listener, and for detecting possible deviations from normal values based on a pre-defined classification of health data, such as the risk levels of Patient Data Analysis. Furthermore, it is responsible for triggering the Alert Handler component by publishing in the broker topics related to each detected deviation type.

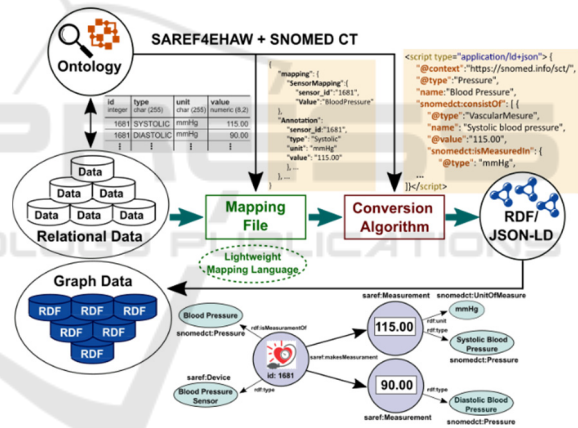


Figure 9: API Listener main activities.

Health data are classified based on acquired knowledge, being used in real health data to assess and predict situations. The Inference Engine deals with two types of knowledge: objective and subjective. Objective knowledge is present in general medical standards, well known by health professionals, easily found in the literature, and widely disseminated by large health organizations such as the WHO. Subjective knowledge is related to the patient’s profile and context, such as medical history, genetic diseases, and personal lifestyle. Objective knowledge can be described in the ontology and, therefore, accessed by the Inference

² <https://saref.etsi.org/saref4ehaw/v1.1.1/>

³ <https://bioportal.bioontology.org/ontologies/SNOMEDCT>

Engine. Subjective knowledge can be manually described by the user, usually the health professional responsible for the patient, and stored in the subjective rules database.

When a *FogHealthData* topic is received, the Inference Engine identifies the data type and the patient associated with these data, and the following steps are applied involving subjective rules:

- a) The Inference Engine verifies if there is any rule in the subjective rules database for this patient regarding this data type.
- b) For each rule found, it checks whether the data fit the rule.
- c) If a deviation from normal values is detected, a specific topic describing this deviation is published, which triggers an alert.

Afterwards, the same steps are applied involving objective rules. The only difference is in the first step, where the Inference Engine verifies in the ontology for these rules regarding this data type.

Objective rules can be skipped if a subjective rule is found that explicitly states that. Some health data have variations by default, which invalidates standard analysis. This may occur due to the patient’s profile and context, which must be informed by the health professional when storing a rule in the subjective rule database. Figure 10 illustrates these steps.

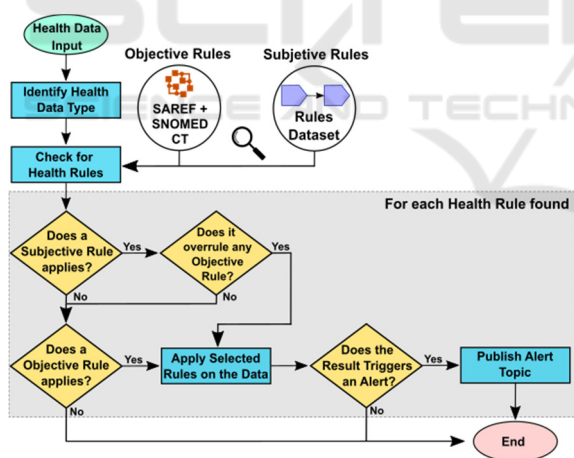


Figure 10: Steps of the Inference Engine.

Rules are described as Semantic Web Rule Language (SWRL) expressions of the form IF health-data-preconditions THEN variants-effects. SWRL expressions can be evaluated by description logic reasoners, such as Pellet, for the evaluation of health data, inferring at runtime new knowledge based on the ontology and rules database. For example, the SWRL rule below defines the Patient Data Analysis normal risk level:

```
Blood_Pressure(?systolic_mmhg,
?diastolic_mmhg), Patient(Pedro),
hasMeasurement(Pedro,?systolic_mmhg),
lessThen(?systolic_mmhg, 120) ^
hasMeasurement(Pedro,?diastolic_mmhg
), lessThen(?diastolic_mmhg, 80) ->
hasHypertensionStage (Pedro, normal).
```

Each classified deviation from normal health data values triggers a topic-related alert message, which is published and consumed by Alert Handler.

4.3 Alert Handler

This component is responsible for processing real-time alerts related to topics published by the Inference Engine, and for contacting the patient and their health professional. Three alerts related to the topics blood pressure, body temperature, and heart rate were respectively implemented in SBioT-MPH. Alerts of these topics provide the following information: the user identifiers (patient and their health professional); the sort of data and their values, and the timestamp of the data and of their sort; and a set of actions to be executed. The actions include sending an email to notify the patient, their health professional, or both.

The user identifier is used so that a request to the Data Handler can be published for retrieving all required information to execute indicated actions (e.g., patient email address). Data are used to create a historic log of all patient alerts, which can be useful for improving pervasive healthcare by helping health professionals to define more accurate subjective rules for each patient. Historic data can be also explored by data mining and knowledge discovery algorithms. An alert log is published as *HealthAlertLog*, and stored in the RDF database by the Data Handler.

4.4 Web Manager and Web Mapper

These components interact with Web App by capturing and redirecting all requests and responses to Data Handler. Furthermore, they provide additional information not available in the relational database to Web App (e.g., alert log).

Web Mapper uses an interface similar to the API Server for receiving a Web App request described in the Apollo Client definition language for GraphQL. Web Mapper converts the received request to a JSON-LD request and publishes it under the *HTMLRequests* topic. After receiving the converted request, Web Manager formulates the corresponding SPARQL queries, and publishes them for being processed by the Data Handler, which in turn returns a response with the processing result. After receiving

the response from the Data Handler, the Web Manager structures the received data for being published in the HTMLData topic. After receiving a response with these data, Web Mapper converts it into a GraphQL response for sending to the Web App.

4.5 Data Handler

This component processes any direct request to the RDF database that is made by other components. It is an endpoint channel that isolates the persisted data, and simplifies mutations and queries management, in accordance with the SOA architectural principles. SPARQL was adopted for requests and responses because it is a structural query language designed for querying RDF data. Since SPARQL queries can be represented as query graphs, they can be answered by performing graph pattern matching over RDF graphs.

5 DISCUSSION

Among the main lessons learned is the use of a holistic approach from the system developer perspective, who needs to be knowledgeable in many technologies that play different roles in the solution. For message exchange, JSON was chosen instead of XML due to the simplicity of its structure and its minimal syntax, which makes it lightweight, and easier to learn, use and read. For the query language and data manipulation, a benchmark was performed between GraphQL and REST using Apache JMeter, in which we concluded that GraphQL has a better performance in terms of response time and a lower average data size rate (bandwidth) than REST (Rodrigues, 2022).

We conceived a semantic model to enable SBioT-MPH to handle heterogeneous devices, data and services. This has been realised in the Cloud Layer, involving ontologies, RDF database (triplestore) and a semantic module. This module was designed keeping as much as possible the original SBioT-MPH structure and, for this purpose, it has an interface that translates all the messages received by the API Server from the Fog Layer and the Web Server. The SAREF4EHAW ontology, which stands out for representing several aspects of health sensors (Moreira, 2020), was extended to describe NCDs especially hypertension, with SNOMED-CT ontology concepts, which is another prominent ontology in the health domain.

RDF was adopted for dataset representation to encode the data with semantic relationships to the ontology, and SPARQL was used to express queries on this data source, since it is capable of querying mandatory and optional graph patterns along with

their conjunctions and disjunctions. The Patient Data Analysis functionality was incorporated to the ontology as a set of SWRL rules employed by the reasoner to determine the hypertension risk level of a patient, turning into an accessible resource to all services in the Cloud Layer.

Furthermore, the proposed rule-based solution is dynamic and adjustable to meet possible changes related to the patient profiles. The semantic module allows the reuse of existing services, and facilitates debugging, update, and maintenance of this module. The messages were encoded in JSON-LD to link the properties of an object represented in a JSON document to concepts defined in the ontology, thus providing additional mappings from JSON to an RDF model. ActiveMQ is the MQTT Broker used to handle the message exchange between services as a publish-subscribe message broker.

The original SBioT-MPH design presents some problems related to interoperability that gives directions for future work. At the Sensor Layer, our sensor platform is the only IoT device able to communicate with the Mobile App in the Fog Layer at the moment. A solution to allow different IoT devices in the Sensor Layer to transmit their data to the Fog Layer is to develop a set of drivers describing multiple authentication protocols.

In the Cloud Layer, the semantic model needs refinement to improve reusability, specialisation, and independence of its components. All services need the Data Handler to access information from the RDF database, which creates a communication bottleneck and deteriorates query performance as the database grows. A solution is to migrate from a single data storage that is shared by all services in the application to a microservices architecture, in which each microservice has its own database (Database-per-service pattern). To achieve this, we still need to investigate how to decompose an RDF dataset without compromising its integrity.

Recent studies have shown that IoT semantic interoperability can be improved if implemented not only in the Cloud Layer but also in the Fog Layer (Rahman and Hussain, 2019). Since this will require the redesign of the SBioT-MPH architecture, a tool for modelling and simulating IoT computing environments, such as the one presented in (Mahmud et al., 2022), will be used to investigate this problem.

6 RELATED WORK

This section discusses some relevant developments that employ IoT, Ubiquitous Computing, and Cloud

Computing in different domains, while also dealing with semantic interoperability.

An IoT-based large-scale SOA (IoT-LSS) ontology that extends the SSN ontology for addressing dynamic service creation, composition and adoption is presented in (Mishra and Sarkar, 2022). The benefits of this ontology have been illustrated in the healthcare domain by mapping the elements of Clinical Decision Support System (CDSS) ontology with IoT-LSS, supporting interoperability, scalability, extendibility, flexibility, manageability and heterogeneity among different entities. Although this work is interesting from the point of view of developing ontologies aimed at IoT semantic interoperability, the application of the proposed ontology in a realistic system like SBioT-MPH was not reported.

An Internet of Medical Things (IoMT) framework called Medical Data Interoperability through Collaboration (MeDIC) is presented in (Jaleel et al., 2020). MeDIC provides services, such as registration, subscription, probing, translation, and publishing, and employs translation resources by means of probing and translating agents at the network edge, where medical data originate. MeDiC is distributed, scalable, and extendable to other IoMT dimensions, such as protocol interoperability, and to other IoT applications, and it will be extended to support protocol and semantics compatibility. Although this work enriches clinical data and translates them from one format to another, it does so by employing agents and not ontologies like in our solution.

An IoT-based health system to monitor and report patients' health conditions at real-time is presented in (Bhuiyan, 2022). This system can transmit health information such as blood pressure, body temperature, heart rate and oxygen saturation, from anywhere, to medical centres and caregivers. It tracks the patient's location using different sensors, transmit data online and offline to mobile apps, and provides alert signals to caregivers once are identified critical health conditions. According to the authors, this system is potentially suitable for rural and urban areas in developing countries. Although this system has some similarities with SBioT-MPH, it does not deal with semantic interoperability.

A healthcare system based on ontological reasoning to monitor patients with chronic diseases called Do-Care is presented in (Elhadj, 2021). Do-Care infers medical information and recommendations based on IoT data, subjective and objective knowledge, and a dynamic rule-based approach. This system employs a modular ontology that integrates three different ontologies: ICNP

medicine ontology, SSN/SOAS sensor network ontology, and FOAF personal profile ontology. The efficiency of Do-Care was tested as well as its ontology, and they intend to integrate the Decision Tree Learner algorithm to this system to predict diseases and provide additional support to physicians in recommending preventive medications. Although this work is similar to ours, some differences are: the Fog Layer in Do-Care only performs data aggregation and transmission and, therefore, the patient can only visualise their data in the cloud; Do-Care employs an ontology built upon different ontologies than the ones we used; and Do-Care does not have a relational or a graphical database, requiring additional functionality when using the semantic module.

7 CONCLUSIONS

In this paper, we reported on SBioT-MPH, a three-layer IoT-based system to monitor hypertension patients. To enable semantic interoperability in this system, a semantic model was designed to incorporate reasoning by using disease and device ontologies. This semantic module was deployed in the Cloud layer, and communicates with the SBioT-MPH applications, the RDF database, and the ontologies, which are also deployed in the cloud.

A possible drawback of our solution is that we replicated the original persisted data in an RDF triplestore. Although data duplication enforces resilience, it also requires mechanisms to guarantee data integrity. For the SBioT-MPH, this mechanism may be rather complex since the databases of this system use different technologies. To investigate this problem, different scenarios will be simulated to check the trade-off of keeping two databases.

As future work, we also intend to extend the sensor platform for collecting other types of clinical data, and to adapt/reuse the SBioT-MPH applications for monitoring patients with other types of NCDs, such as Diabetes, Asthma and Obesity. In addition, WHO recommends the practice of physical activity for preventing NCDs, and since the scenarios for monitoring physical activity are similar to the ones for monitoring NCDs patients, we believe SBioT-MPH can be extended to be used for this purpose.

In the literature, several fog-based approaches that provide solutions for heterogeneity problems have been reported, but semantic approaches are usually implemented in the cloud. Since the SBioT-MPH fog layer is suitable for pre-processing of delay-sensitive data at the network edge, in future work we intend to investigate and propose a semantic approach in which

semantic support is partially deployed in the fog and partially in the cloud.

ACKNOWLEDGEMENTS

This study was financed in part by the ‘Coordenação de Aperfeiçoamento de Pessoal de Nível Superior’ - Brazil (CAPES) - Finance Code 001. We also thank the Brazilian National Council of Technological and Scientific Development (CNPq) and the São Paulo Research Foundation (FAPESP) for sponsoring our research in the context of the Brazilian National Institute of Science and Technology in Medicine Assisted by Scientific Computing (INCT-MACC).

REFERENCES

- Bhuiyan, M. N. et al. (2022). Design and Implementation of a Feasible Model for the IoT Based Ubiquitous Healthcare Monitoring System for Rural and Urban Areas. *IEEE Access*, IEEE, Volume 10, pp. 91984-91997.
- Buneman, P., Staworko, S. (2016). RDF Graph Alignment with Bisimulation. In: *Proceedings of the VLDB Endowment*. Volume 9, Issue 2, pp. 1149-1160.
- Castaneda, D., et al. (2018). A review on wearable photoplethysmography sensors and their potential future applications in health care. *International Journal of Biosensors & Bioelectronics*, Volume 4, Issue 4, pp. 195-202.
- Chitra, L. P., Satapathy, R. (2017). Performance comparison and evaluation of Node.js and traditional web server (IIS). In: *Proceedings of 2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*. IEEE, pp. 1-4.
- Elhadj, H. B. et al. (2021). Do-Care: A dynamic ontology reasoning based healthcare monitoring system. *Future Generation Computer Systems*, Elsevier, Volume 118, pp. 417-431.
- Gawanmeh, A. and Al-Karaki, J. N. (2021). Disruptive Technologies for Disruptive Innovations: Challenges and Opportunities. In: *Proceedings of 18th International Conference on Information Technology: New Generations (ITNG 2021)*. Springer, Advances in Intelligent Systems and Computing, Vol. 1346, Cap. 55, pp. 427-434.
- Husain, M. S. et al. (2022). Pervasive Healthcare - A Compendium of Critical Factors for Success. *EAI/Springer Innovations in Communication and Computing (EAISICC)*, Springer, 379 pp.
- Jaleel, A. et al. (2020). Towards Medical Data Interoperability Through Collaboration of Healthcare Devices. *IEEE Access*, Volume 8, pp. 132302-132319.
- Jeon, D.-c., Liuhaoyang, Hwang, H. (2019). Design of Hybrid Application Based on GraphQL for Efficient Query for PHR. In: *Proceedings of 2019 International Conference on Information and Communication Technology Convergence (ICTC)*, IEEE, pp. 381-383.
- Kjeldsen, S. E. (2018). Hypertension and cardiovascular risk: general aspects. *Pharmacological Research*, Elsevier, Volume 129, pp. 95-99.
- Mahmud, R., et al. (2022). iFogSim2: An Extended iFogSim Simulator for Mobility, Clustering, and Microservice Management in Edge and Fog Computing Environments. *Journal of Systems and Software*. Volume 190, 17 pp.
- Mishra, S. K. Sarkar, A. (2022). Service-oriented architecture for Internet of Things: A semantic approach. *Journal of King Saud University - Computer and Information Sciences*. ScienceDirect, Volume 34, Issue 10, pp. 8765-8776.
- Moreira, J. et al. (2020). SAREF4health: Towards IoT Standard-based Ontology-driven Cardiac E-health Systems. In: *Applied Ontology*, Volume 15, Issue 3, pp. 385-410.
- Noura, M., Atiquzzaman, M. and Gaedke, M. (2018). Interoperability in In-ternet of Things: Taxonomies and Open Challenges. *Mobile Networks and Applications*, Springer, Vol. 24, pp. 796-809.
- Rahman, H., Ahmed, N., Hussain, M. I. (2018). A QoS-aware hybrid data aggregation scheme for internet of things. In: *Annals Telecommunications*. Springer, Volume 73, pp. 475-486.
- Rahman, H., Hussain, M. I. (2019). Fog-based semantic model for supporting interoperability in IoT. *IET Communications*. The Institution of Engineering and Technology, Volume 13, Issue 1, pp. 1651-1661.
- Rahman, H., Hussain, M. I. (2020). A comprehensive survey on semantic interoperability for Internet of Things: State-of-the-art and research challenges. *Transactions on Emerging Telecommunications Technologies*. Volume 31, Issue 12, 25 pp.
- Rodrigues, R. J. S. (2022). “SBIDC-MPH: Sistema Baseado em Internet das Coisas para o Monitoramento de Pacientes com Hipertensão”. *MSC dissertation in Portuguese*. Graduate Program in Computer Science (PPG-CC), Computing Department (DC), Federal University of São Carlos (UFSCar), 110 pgs.
- Sandi, G., Nugraha, I. G. B. B., Supangkat, S. H. (2013). Mobile health monitoring and consultation to support hypertension treatment. In: *Proceedings of International Conference on ICT for Smart Society*. IEEE, pp. 1-5.
- Souza, P.L.; Souza, W.L; Ciferri, R.R. (2022). Semantic Interoperability in the Internet of Things: A Systematic Literature Review. In: *ITNG 2022 Proceedings of 19th International Conference on Information Technology: New Generations*. Springer, Advances in Intelligent Systems and Computing, Vol. 1421, pp. 333-340.
- WHO (2013). A global brief on hypertension: silent killer, global public health crisis: World Health Day 2013. World Health Organization, 39 pp. Available at <https://apps.who.int/iris/handle/10665/79059> [last access: 14/11/2022].
- Zhang, H-t., Zhang, Y-k. (2012). Architecture and Core Technologies of Internet of Things. *Journal of Changchun University of Technology*. Natural Science Edition, Volume 2, pp. 176-181.