Smart Solution for Heterogeneous Device Interoperability in IoT

Tooba Maryam¹, Aslam Muhammad², Martinez-Enriquez A. M.³
¹,²Department of CS &, U. E. T., Lahore, Pakistan (tooba_islamic@yahoo.com, maslam@uet.edu.pk)
³Department of CS, CINVESTAV, D.F. Mexico (ammartin@cinvestav.mx)

Abstract: Internet of Things gives an image of future where physical and virtual things can be connected with each through internet and can talk with each other without human intervention. The characteristic of IoT to connect anything, anywhere and anytime enables an ultra large number of devices, things and networks with high level of heterogeneity to connect with each other. This introduces interoperability issue among heterogeneous devices in IoT ecosystem. In this paper, we proposed a solution, Smart System for device Interoperability in Internet of Things, to interconnect devices in an IoT ecosystem. It acts as a central element allowing the heterogeneous devices to communicate through it regardless of difference in their protocols and other communicating technologies. Detailed architecture of the proposed system is presented and each component is discussed in detail. Our proof of concept is presented through a real life case study from health and fitness domain.

Keywords: Interoperability, IoT, Heterogeneous, Device interoperability.

I. INTRODUCTION

Internet of Things(IoT) basic idea is to connect things with each other through internet where things can be sensors, actuators, tags, and mobile phones. Internet of Things is applicable in all the domains of human life e.g. Health, smart home, smart city, energy, and logistics etc. Internet of Things opens a new era of interaction in the physical world where all the things around us can be connected to internet anytime anywhere. A recent prediction presented by Corporation International Data Center (IDC) shows that Internet of Things and its associated ecosystems will be of worth 7 trillion dollars in 2020 [1]. Recent advances in communication technologies, such as lower power consumption, easy data connectivity and decrease cost of devices, make it possible.

Internet of Things vision is to connect all the things in such a way that they can collect, process and share information with each other [2]. All devices must be able to communicate and interact with each other regardless of their communication technologies. One of the main obstacles that IoT faces is the high degree of heterogeneous heterogeneity. Devices have different communication protocols, technologies and hardware. In fact the interoperability issue is one of the major challenges to be addressed while developing and integrating new IoT ecosystem [3]. Interoperability issue can affect up to 40% benefits of the IoT [4].

In IoT, interoperability is ability of the two components or systems to exchange and use information with each other [5]. In IoT context, interoperability can be established at different levels i.e. interoperability of protocols and interoperability of data. Interoperability of protocols means to establish direct connection between different network technologies.

Interoperability of data is associated with the syntax and semantics of the data. The messages exchanged between two connected devices must also be interoperable to process and understand data. All communication protocols have their own data syntax and encoding technologies [5].

Solving the interoperability issue will allow the Internet of Things ecosystem to prosper in right direction and obtain the true meaning IoT that is hidden in transmission and understanding of data. In this paper, we proposed a Smart System for device Interoperability in Internet of Things (SI2oT) that enables interoperability between heterogeneous devices on the network, syntax and semantic level. Interoperability is achieved through conversion of communication protocols, processing and transferring data. Proposed system allows communication between different protocols such as Bluetooth, Zigbee, Wi-Fi, and ANT+ etc at network level. Data interoperability is achieved through translating all the obtained data in a uniform format and ontology are used to add semantic to the data to make it meaningful for devices. Protocol conversion, ontology and transmitting routes are decided by the decision making components that leaned through Naïve Bayesian Classifier. Then data is transferred to the destined device by converting into the destined protocol.

This paper is structured as follows: Section II presents the state of the art of the previous work. In Section III, the architecture of the proposed system is presented. A case study of SI2oT's application is presented in Section IV. Section V briefly defined the interoperability challenges achieved by the proposed system. Section VI discussed the conclusion and future work.

II. RELATED WORK

To resolve the issue of interoperability, multiple solutions are proposed such as middleware, gateways, and hub based solutions etc.

The authors in [6] and [7] Proposed IoT Gateways for Wireless Sensor Networks (WSNs) and mobile communication networks. These gateways are responsible for protocol conversion to facilitate data transmission and device administration. Some researchers proposed gateway solution for specific domains. For example in [8] and [9], Gateways for e-health and home are presented respectively. Gateway for e-health provides interoperability among WiFi, Bluetooth,
6LoWPAN and the underlying protocols through the use of WebSockets. A gateway for home provides interoperability between different network protocols like X10, Insteon, ZigBee and UPnp.

In [10], a very recent and detailed survey on IoT middleware is presented. IoT middleware requirements are divided into three major categories i.e. functional, non-functional and architectural requirements. Functional, non-functional and architectural requirements for IoT middleware are discussed in detail for better understanding. Then, Middleware solutions are categorized on the basis of their design. All the existing solutions are reviewed according to the requirements. Then the existing challenges are discussed for future researcher. Authors reviewed interoperability level in existing middleware solutions as an architectural requirement for IoT middleware. In [5], a hub based solution is proposed to solve interoperability issue in IoT ecosystem. It represents resources on the web using RDF model. RDF uses triple statements to provide information about format and semantics of the data. This enables application developer to access required data and understand the data. In [11], another approach named as INTER-IoT is been proposed to solve interoperability issue among all layers of internet. This approach is based on INTER-Layer, INTER-FW and INTER-Meth. Where INTER-Layer is responsible for granting interoperability among hardware and software in an IoT environment. On the other hand, INTER-FW is a framework for development of interoperable applications and INTER-Meth provides methodology for IoT integration.

Most of present work focused on designing solutions to solve the interoperability of protocols and specific communication technologies without considering data syntax and semantic issues. In existing solutions, network and semantic interoperability are focused to solve while syntax interoperability is completely ignored which is necessary for data communication among devices. A smart middleware solution is needed to have complete solution to make heterogeneous devices interoperable in an IoT ecosystem.

III. ARCHITECTURE OF SI2oT

The main purpose of the this proposed solution is to provide seamless connectivity at network, syntactic(syntax and format) and semantic level between heterogeneous devices. Fig. 1 shows the main architecture of the proposed solution SI2oT.

The SI2oT have five main components: (i) Sender, (ii) Event Handler, (iii) Naive Bayesian Classifier, (iv) Ontology handler, and (v) Receiver. These component are responsible to carry out the sequence of actions while making successful communication among devices in an IoT ecosystem. All the major components of the SI2oT are discussed in detail below:

A. Sender

Sender is the resource discovery component or the publisher of the events. In an IoT ecosystem, different type of sensors and actuators are there and offer their resources to other IoT applications or devices. Data from the sensors and other devices is initially collected by senders and pass them as events to the event handler for further processing. To achieve connectivity among devices within an IoT ecosystem, set of multiple interfaces are developed in sender and receiver component. Devices in an IoT have different radio technologies e.g. IEEE 802.15.4, IEEE 802.11n, Bluetooth , ZigBee etc. Therefore, module for each technology along with required hardware must be implemented in the sender and receiver component. Figure 2 shows the general architecture of the Sender.

![General Architecture of Sender](image_url)

The purpose of the sender is to receive the frame from an IoT device, sensor or actuator and parse it to the event handler. Sender is also responsible for appending the required frame details, like RSSI values and timestamp, into the JSON payload. The structure of the JSON payload is as below.

```json
{
  payload:
  {
    rssi: float
    SndID: string //sending device MAC address
    SndInterID:string //sending interface ID i.e. WiFi, ZigBee etc.
    timestamp: string
    samples: {
      sensorvalue: integer
    }
  }
}
```

The above given format is used by the modules that are implemented for ZigBee, Wi-Fi and Bluetooth. In case, the connected device is not a sensor the sample section remained empty.
empty. This format is very feasible for sensors as its very simple and lightweight. The sample section only carries the name and the value send by the sensor. Integer is used to represent sensor value and sensors send numeric value. Sensor value or data value of the packet should not be converted as it may damage the information within data.

Wifi and Bluetooth sender are responsible for monitoring traffic that uses IEEE 802.11n and IEEE 802.15.4 frequency band respectively within an IoT ecosystem. Other sending interface modules are responsible to monitor their respective frequency bands to detect and connect with devices when they are range.

B. Event Handler

Events sent by the sender module for further processing are received, processed and send to the destination by the Event handler. Event handler has four sub-component as shown in Figure 3.

![Figure 3. General Architecture of Event Handler](image)

1) Event Receiver (ER)

ER receives the event from the sender. ER checks the JSON and makes sure that the required information is complete. Information required is MAC address of the sender, frame and receiver's address in case of communication among two devices. Integrity of MAC address of the sender and data packet is enough when the sending device in sensor or data is being sent to the server on the receiving end. If the integrity of the MAC address or data frame is compromised then ER will notify the sending module of the Sender. Error message to resend the event will be published by ER to sending module.

2) Event Distributor (ED)

ED is the one that keeps the log of the event movement and its process. ED is responsible for making sure that an event can move from one component or module to another successfully. It is also responsible for keeping the log of event entrance and exist times along with its route. So that an event can be trace back if some failure occurs.

3) Event Controller (EC)

EC is responsible for successful conversion of the protocol and making data meaningful for the receiving end. Initially, EC receives event from ED. It extracts the MAC address of sender and receiver from the payload. EC pass this information to the NB classifier to ask for further action. NB classifier then decides the change in data format or semantics. NB classifier adds special tags in case of change in data format and semantic which helps event controller to convert the protocol and send it to specific ontology manager. For interconnection and interoperability between devices Heterogeneous protocol conversion is necessary. When the event controller receives a data packet through an event, it does not change the message data of the source but maps the packet format according to the communication protocol of destination device. This Functionality is carried out once the data has been processed by the ontology handler. Ontology handler add required semantics, EC sends it to the event distributor so that event is delivered to the destined receiving interface.

4) Policy Database

Controlling access to specific devices and servers is also a job of event handler. Along with that keeping the record of different protocols stacks i.e. rules, data formats, encryption rules etc. are also managed by event handler. Event handler performs this job by calculating the event information using policy database. There are devices that can only be accessed by already declared set of devices. All other devices or sensors are not allowed to access such devices. This is also very important to make sure that all devices can be secured from malicious and unauthorized access. Difference of Data format, encryption and other communication rules is also stored in policy database that ask event handler to make required changing on data to make it interoperable among heterogeneous devices.

C. Ontology Handler

Ontology Handler is responsible for the adding semantic to the given information. For this purpose, ontology are divided in two categories as shown in Figure 4.

1) Abstract ontology

Abstract ontology is the ontology that have common elements of all IoT domains. It also has ability to get extended in order to represent some specific domain in detail. Abstract ontology returns the query within small processing time and consuming small number of resources. Within an IoT ecosystem, mostly actuators and sensors perform same task, regardless of the domain they are working in. Although the decisions to be taken on the basis of their output may vary from domain to domain. For example, sensor that senses the brightness and asks the light management system to bright or
dim the light works the same in smart home, hospital, smart city etc. So the abstract ontology represents such common sensing, actuating and tag devices. There further extension is implemented in their respective domain ontology. It is important to place such objects at abstract level so that processing time and resources can be reduced.

2) Distributed ontology

Distributed ontology represents some specific domain in an IoT environment. These can also be called as domain specific ontology e.g. smart home ontology, smart hospital ontology, smart city, smart shopping mall etc. Distributed or domain specific ontology presents the detailed semantics of that domain so that the true semantic interoperability can be achieved. Such ontology have specific domain based knowledge database to add semantics to the data belong to that domain. A device that is designed for irrigation of a farm is an agriculture domain specific device. To design semantic ontology for such device needs a lot of details and in depth knowledge of its domain. In such case, distributed ontology must be implemented so that user and other devices can get better understanding of the device when interacting with it.

D. Naive Bayesian Classifier

In SI2oT, rules are developed on the base of ECA(Event-Condition-Action) model. In Naive Bayesian Classifier, ECA is a rule mechanism that is not dependent on any rules repository. It is used in artificial intelligence to trigger some predetermined action on occurrence of some event and satisfaction of specified condition. The Naive Bayesian is applied and trained on the basis of available rules and decisions regarding heterogeneous devices are also achieved. So the decision about the destined device if not determined by the sender, format and semantic decision are all on the base of the highest probability of the events in the past. Rules are designed by using highest probability of the available device types on the basis of their protocols and their presence in IoT domain for network and semantic related decisions respectively.

The format of the rule for interoperability among devices in given below: 

On event ‘l’ if condition ‘l’ do action ‘l’; Rule: On e1 if c1 do a1

E. Receiver

Receiver has the same functionality as sender. It discover the receiving end devices and establish a successful communication with them so that data can be shared without any hinders.

Along with sending interfaces as shown in Fig. 5 i.e. WiFi, Zigbee, Bluetooth etc, receiver has also modules that receive data packet for CoAP, MQTT, HTTP etc servers. These modules receive events from the event handler and forward it to the respective servers in an IoT ecosystem.

IV. CASE STUDY

There are different application domains of the IoT such as health care, education, industry and environment etc. In this paper, we describe a case study from health and fitness domain where SI2oT is applied and tested. Now-a-days people are very concerned about their health. Heart patients and sports persons required to maintain the record of their heart condition on regular basis. For this purpose they use different type of heart rate monitors that interact with other devices and assisting applications to update the user’s record. In this scenario, there is a wireless heart rate monitor that provides instant feedback about user’s heart rate. It tracks heart beats and sends this information to the fitness or healthcare application for further processing. In this case study, three main components are used: i) Heart rate monitor having ANT+ technology, ii) SI2oT middleware and iii) Assisting application stored on an android cell phone having Bluetooth. Heart rate monitor uses ANT+ technologies and can only interact with ANT+ compatible devices. But SI2oT helps it to interact with Bluetooth device that is an android cell phone. ANT+ receiving interface discovers the ANT+ heart rate monitoring sensor and receives the information from it. Then generate an event and add a JSON payload having detail about heart rate monitor and data send by it. This event is then sends to the event handler which check its authentication through rules repository. As this device is not blacklisted, the event handler processes it according to the rules directed by Naive Bayesian rules repository. Any ANT+ compatible devices is not present in the this IoT environment, Naive Bayesian rule is to launch it through Bluetooth sender and no further changing are made to data. As receiving data is supposed to send to a device that is expecting a data from sensor. So no semantic changing are required in this scenario. Data is then launch through Bluetooth sender by adding Bluetooth header to the sending data packet. The android device receives the data through its Bluetooth and pass it on to the assisting application.

V. ACHIEVEMENTS

In this section, we briefly list the key challenges achieved by our proposed system.
A. Device Discovery

A middleware system must detect the device when it appears in the range of an IoT ecosystem. It must be capable to identify the source that can be a device, sensor, or actuator etc and then notify the interested source about its presence. Proposed system is capable enough to detect a new device when it appears in the range of specific gateway. WiFi, Bluetooth, ANT+, and Zigbee etc. are the specific gateways or interfaces that are implemented on sender and receiver ends.

B. Device Management

Once a device is discovered successfully, next comes retrieving its data and sending it to the destination. For effective device management, this module needs to have repositories for storing information regarding devices i.e. types of devices, their enabling technologies, data formats etc. Device description and device control are all the responsibility of device management module.

In SI2oT, once the device is discovered by the specific gateway, gateway interface e.g. WiFi sender, Bluetooth sender etc. recognize the type of the device and retrieve its data. SI2oT has policy database that stores the record of devices and their enabling technologies that is also be used for authentication purpose.

C. Event Handling

All events must be handled in an effective way so that the sending devices can be discovered and interested parties can be notified about the presence of their required resource. But handling event data is also an important requirement for a middleware. Data representation of event data in the programming language is a must. For this purpose, data must be naturally mapped from event data to programming language instances. In SI2oT, all events are handled in an effective way so that sending and receiving devices can be discovered when needed. SI2oT also keeps record of the events, the route log and acknowledgment reports.

D. Interoperability

Interoperability among devices is provided at network, syntactic and semantic level. All communication protocols are interoperability is achieved by JSON. For semantic implemented at sender and receiver ends and protocols is converted for successful transmission.

<table>
<thead>
<tr>
<th>Systems</th>
<th>Network Interoperability</th>
<th>Syntactic Interoperability</th>
<th>Semantic Interoperability</th>
</tr>
</thead>
<tbody>
<tr>
<td>HYDRA [10]</td>
<td>ZigBee, Bluetooth, RFID, Wi-Fi, HTTP, SOAP</td>
<td>• Ontology, OWL</td>
<td>• Ontology, OWL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• No proof</td>
<td>• Large Vocabulary</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Domain Specific</td>
<td>• Domain Specific</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Lack of Standards</td>
</tr>
<tr>
<td>Ubiware [10]</td>
<td>RFID, Wi-Fi, Pull/push, HTTP</td>
<td>• Metadata</td>
<td>• Ontology, OWL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• No proof</td>
<td>• Large Vocabulary</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Too many standards</td>
<td>• Domain Specific</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Lack of Standards</td>
</tr>
<tr>
<td>HYPERCAT [5]</td>
<td>ZigBee, Bluetooth, RFID, Wi-Fi, HTTP, SOAP</td>
<td>• Not Supported</td>
<td>• Web Semantic, RDF</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Lack of expressiveness</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• More response time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Free of world</td>
</tr>
<tr>
<td>IoT Gateway System [6]</td>
<td>ZigBee, WSNs</td>
<td>• No Support</td>
<td>• No Support</td>
</tr>
<tr>
<td>SI2oT</td>
<td>ZigBee, Bluetooth, RFID, Wi-Fi, ANT+, ...</td>
<td>• JSON</td>
<td>• Abstract and Distributed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lightweight &amp; Faster</td>
<td>Ontology</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Easy to understand</td>
<td>• Large vocabulary</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Limited set of Standards</td>
<td>• Less response time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Lack of expressions</td>
<td>• Lightweight</td>
</tr>
</tbody>
</table>

Table 1. Comparison between existing and proposed system
Syntactic or syntax interoperability, abstract and domain specific ontology are used. Interoperability among components of proposed system is also achieved using JSON. As JSON is an interoperable format with all programming languages. In other words, all devices present within an IoT ecosystem are able to interact through SI2oT regardless of their technologies.

E. Extensibility

Middleware systems evolve with the passage of time. Extensibility is the most important challenge while designing Middleware solutions. In IS2oT, all components are loosely coupled and have minimum dependency on each other. So, new component can be added easily with minimal human effort. In case new protocol needs to add in present design, sender and receiver interfaces for the protocols are just need to be added.

F. Security and Privacy

Confidentiality, authentication and non-repudiation are the responsibilities IoT middleware have to fulfill. Security is maintained by just letting the authorized devices become a part of IoT ecosystem. For this purpose, policy database is used to avoid the un-authorized devices being a part of IoT ecosystem.

G. Comparison with existing systems

Table 1 presents a small comparison among existing proposed systems and SI2oT. There are many proposed systems to provide interoperability in IoT environment. We choose the one’s which are providing high level of interoperability support. In the presented table network interoperability presents the protocols supported by the proposed system. Syntactic interoperability presents techniques used to support syntax interoperability and its limitations. Semantic interoperability again presents the methodology used for semantic support and its limitations. As mentioned in related work, existing systems are solving network interoperability issue along with domain specific semantic and syntactic interoperability. But IoT is not a domain restricted environment. Multiple devices from all kind of domains are supposed to be interacting with each other in an IoT ecosystem.

SI2oT has achieved interoperability at network level where devices having any kind of communication protocol can interact with each other. At syntactic level, data can be transferred among devices using JSON without worrying about its syntax or format as JSON is understood by all programming languages. And Semantic interoperability without any hinder of domain can be achieved with the help of distributed ontology.

VI. CONCLUSION

Existence of heterogeneous devices within an environment is the main hurdle in the adoption and deployment of IoT. In this paper, a Smart System is proposed to ensure interoperability between devices at protocol, data and semantic level. SI2oT acts as a central element allowing the heterogeneous devices to communicate through it regardless of difference in their protocols and other communicating technologies. Detailed architecture of the proposed system is presented and each component is discussed in detail. Network interoperability is achieved by protocol transformation, syntactic and semantic interoperability is achieved through JSON and ontology respectively. The SI2oT is applied to the health and fitness domain although it can be applied to any domain of IoT to enhance interoperability among heterogeneous devices. In the end, key middleware challenges achieved through proposed system are discussed. In future, we propose to extend the domains of its implementation so that it can be tested for wide variety of devices along with semantics as well. And other machine learning techniques can be used to achieve more and more learning capabilities of the system.

REFERENCES