Received: September 26, 2018

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Web Architecture for Monitoring Field using Representational State Transfer Methods

Krishna Kumar Palavalli Radharamana[1]* **Chandra Mouli[2]** **Udaya Kumar[1]**

[1] *Department of Computer Science & Engineering, Cambridge Institute of Technology, India*
[2] *Department of Computer Science & Engineering, East point College of Engineering, India*
* Corresponding author's Email: Rana.krishnakumar@gmail.com

Abstract: Internet of Things (IoT) provides information services based on daily usage that depend on the sensors of devices and network platform. An increase in the number of connected devices through internet, increases the demand for the number of low-latency services. In this research, Open Cloud Computing Interface (OCCI) technique is used in IoT architecture that helps to encompass application level interface. The OCCI-based architecture is proposed to manage and store the data, by introducing the resource analyzer, edge devices and monitoring manager that helps to transfer the data effectively. The monitoring manager, edge devices and broker schedule the data to minimize the traffic in the IoT. Representational State Transfer (REST) methods using Hyper Text Transfer Protocol (HTTP) for communication are presented in this technique. Simulation result showed the effectiveness of the proposed OCCI architecture. Arduino and Raspberry Pi 3 are the two major hardware used in this technique. The result of the OCCI-based architecture uploaded to the ThingSpeak server, which is the external server. Several parameters such as Temperature, Round Trip Time (RTT), latency, Clock difference and frequency are evaluated in this work. Round Trip Time reduced to 0.96 seconds by reducing the delay in the system.

Keywords: Arduino, IoT architecture, Open cloud computing Interface, Raspberry Pi 3, ThingSpeak.

1. Introduction

Internet of Things (IoT) has been growing in a wide range of domains, those are, health appliances, entertainment electronics, wearable gadgets and industrial sensors, those are connected over the Internet. Most of the devices are embedded devices that are portable and powered by batteries. These devices communicate are required to be wirelessly communicated with each other or through some remote IoT gateways [1, 2]. IoT refers to an ecosystem in which devices such as smart phones, sensors, and household appliances are connected to the global network, which helps to monitor and manage the devices. These types of devices are available in a wide range of devices and are mainly supported by hardware micro controllers [3, 4]. New technologies have enhanced the effect of information, communication, control and computation. This ensures that an IoT lends itself as an "ideal" tool to manage agriculture practices [5]. Remote monitoring of factors that affects artworks helps to preserve in the long-term and also promote their value. IoT is one of the main technique in current technology helps in monitoring the artworks [6]. IoT has become an important technique and can be used in object, small embedded system that include computing, networking capabilities and sensing.

The sensed information is collected from the object and uploaded in the network cloud and this can be analyzed to provide new intelligent services [7]. Many industrial communication protocols are available to support Facility Smart Grid Information Model (FSGIM) for energy management in industries and by applying different private protocols will result in poor interoperability. IoT is an alternative tool for operating the devices and it is expected to make the industrial production more intelligent and efficient [8, 9]. Energy efficiency is

becoming a huge research area and it is expected to achieve 20% savings in energy consumption by the year of 2020. The challenge is generally present on multiple levels, namely energy consumption in the physical performance, improved energy data management and automation, and the introduction of better energy consumption practice with human users [10]. In this research, an OCCI is used to encompass the application level interface of the IoT infrastructure, which provides the interface of the single component. This helps to optimize the usage of resources in the IoT environment. The greenhouse field is used as the case study in this method for the evaluation purpose. The sensor device is plotted across various part of the study area and temperature is monitored across the field. The new architecture is OCCI-based architecture in this method that has monitoring manager and Analyzer. These two devices handle the data flow and the data is stored in the cloud. This technique improves the performance of the data flow and provide effective monitoring technique. Monitoring manager handles the resources allocation in the various devices and doesn't violate Service Level Agreement (SLA). Analyzer investigate the data flow and improves the reconfiguration process in the system. This technique is compared with existing method of different method to improve the monitoring system. Experimental result shows that the OCCI-based architecture has better performance compared to existing technique in IoT monitoring technique.

The organization of the paper in the manner of Literature review of IoT monitoring techniques in the section 2, section 3 contains the brief explanation of OCCI-based architecture and experimental result in the section 4.

2. Literature review

Yingfeng Zhang and Shudong Sun, [11] designed the manufacturing system with IoT to provide a new technique of manufacturing with the IoT system. This method monitors the conditions related to the production system such as operators, pallets, materials, machines etc., are connected with the sensor. These were monitored using this technique and optimization were carried out with the real time data to improve the productivity of the shop-floor. This method also helps in reducing the wastage of the manufacturing resources, which decreases the production cost, risk and improve the efficiency in cross-border custom logistics. This senses the changes and parameters in the productions optimized which depends on the monitored real time manufacturing information and also to eliminate the disturbance of the production process.

S. Muralidharan, A. Roy, and N. Saxena [12] aims to reduce the delay and also classify and prioritize the IoT traffic with low latency, that effectively retrieves the data. A Markov Decision Process (MDP) based Interest scheduling proposed in this paper for IoT by priorities and also calculate the performance of the proposed method in various traffic probabilities. The experimental result showed that the operations of prioritizing and scheduling these requests based on their network type helped to reduce the network traffic by 30%, which increased QoS in IoT environment. The scheduling of the request using the Markov Decision Process (MDP)-based IoT reduced the Round Trip Time (RTT) values up to 20%-30% than the state-of-art method in forwarding requests and the incurred delay decreased up to 30%. The flow was not controlled on the path and also in the data with data scheduling of Pending Interest Table.

S. Liu, G. Zhang, and L. Wang [13] established the technique for real time information driven dynamic optimization distribution for logistics tasks by adapting a bottom-up logistics strategy. This method is focused on the objective to design the innovative system for providing sustainable logistics in logistics distribution model. In order to evaluate this technique a real-time IoT-enabled information sensor is developed, which sense and capture the real-time data of logistics resources. These observed values are shared among companies after the value-added processes. The new paradigm was proposed for dynamic optimization of Real-time data of logistics resources to achieve the optimized configuration of logistics cost, logistics resource, distribution distance and energy consumption, and also alleviate the environmental pollution. This method needs to be evaluated for different real-time process to understand the efficiency of the system.

Soobin Jeon and Inbum Jung, [14] presented the method namely Improved MinT (MinT-I) in order to improve the performance of MinT middleware by the real-time adjustment of threads. This method majorly focused on the connection part of the system responsible for processing, analyzing and retransmitting the received packets. The simulation results showed that the MinT-I method increased the average throughput of approximately 25% to 30% compared to the existing methods. This proposed technique not only optimize the resources and memory, this also reduced the latency and power consumption of IoT devices. Sensing devices produced the periodic aggregation, which doesn't

update the information frequently that causes transmission delay and excess energy consumption.

H.R. Arkian, A. Diyanat, and A. Pourkhalili [15] studied existing trends in the optimization of IoT resources and proposed the method based on a fog computing scheme namely, MIST that supports crowd sensing applications in the context of IoT. The investigation performed on the data consumer association, task distribution, and virtual machine placement towards MIST to provide cost-efficient and limited resource in IoT. In this technique, initially the problem was formulated into a mixed-integer non-linear program (MINLP) and this was linearized into a mixed integer linear program (MILP). This technique was evaluated using the data of real world parameters of the Tehran province, which is the capital of Iran. Simulation results showed that the performance of the proposed method MIST increased with the number of applications demanding real-time service increases while compared to the conventional cloud computing. The investigation was not conducted for the mobility of data generators and data consumers and its impact on the performance.

3. OCCI-based architecture method

The overview of the OCCI-based architecture method's architecture in terms of the greenhouse study presented in the Fig. 1. The temperature of the greenhouse recorded by the sensor and the raw data transferred to the environments data collectors. These data, then uploaded to the ThinkSpeak and the Raspberry pi act as a Message Queuing Telemetry Transport (MQTT) Broker. The analyzer analysis the data and temperature is monitored, and provide the optimized value in the monitoring. The aggregator is connected to the several sensors placed across the greenhouse. An OCCI technique used as an interface that interconnect with the different layers in the system. The definitions of Monitor Manager and Analyzer are given below.

- **Monitoring Manager**: This denotes the resources handling SLA and this has the permission to activate the necessary resources for monitoring and reconfiguration functionalities.
- **Analyzer resource**: This process monitors the information and increases reconfiguration action

3.1 An OCCI method in IoT architecture

The OCCI specification depends on the core schema that supports an extension mechanism. The core OCCI method describes two types of core techniques: the resource and the link. A link connects with the two resources: source resource and the target resource. Uniform Resource Identifier (URI) connects with each entity and REST-full interface is presented to the user that helps to interact with the entities on behalf of the management system. A given entity has a defined attribute; whose values represents a specific instance. Ability to perform an activity is the action that is defined by a category, which is often supported by the abstraction of an internal state of the entity. Fig. 1 shows the architecture of the monitoring method.

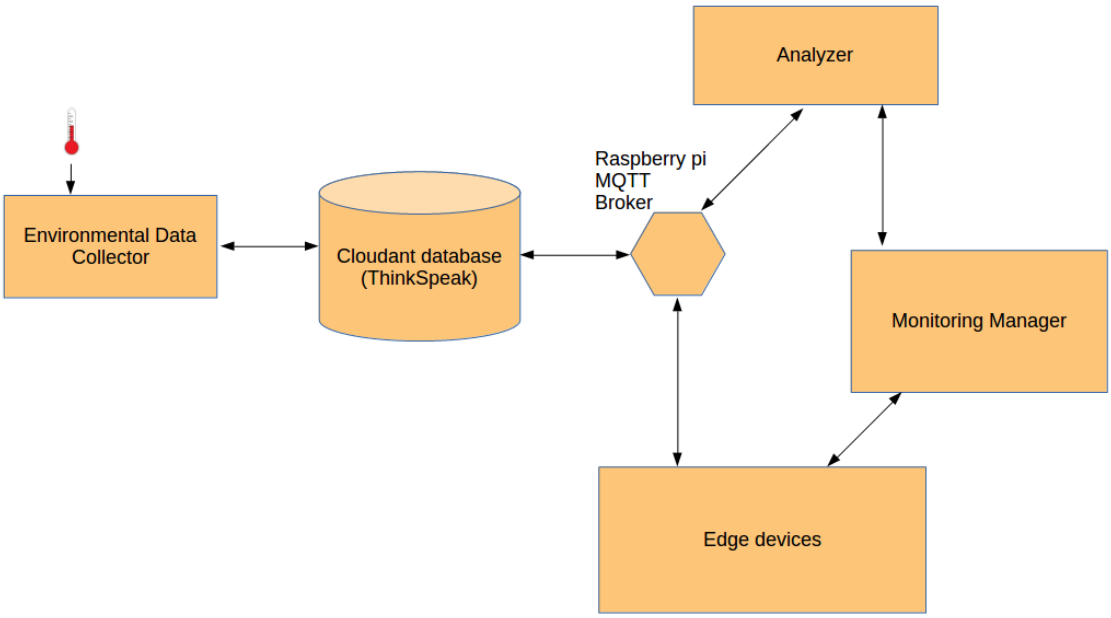

Figure.1 Architecture of the OCCI-based architecture method

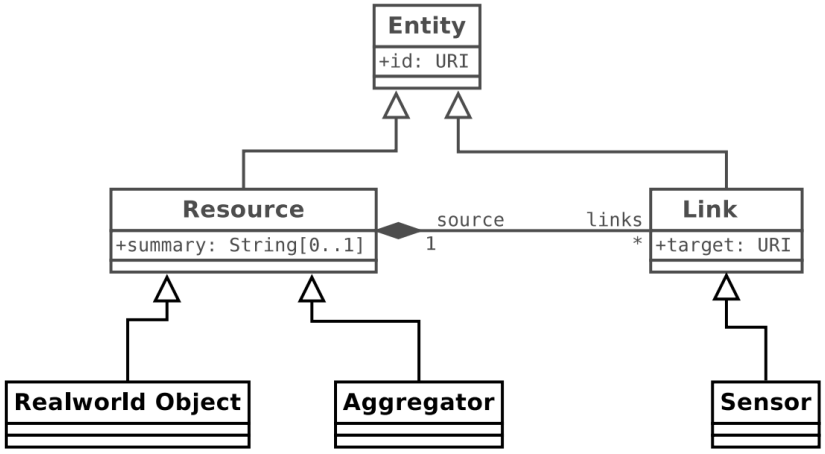

Figure.2 The unified modelling Language (UML) class diagram of the IOT extension (core model light-gray)

OCCI introduces the mixin in order to customize instance of a given entity. While these are associated with one of an entity, instance added a new attribute and action. That complement those entities which were already present in the entity instance. In the commercial point of view, the mixins have the major role of differentiating the offers of distinct providers: entity types are the common denominator that allows basic portability, while mixins are provider-specific. OCCI is simple that can be adapted to a number of distinct environments: historically the first application of OCCI gives the description of IaaS cloud resources, Service Level Agreement, PaaS resources, Cloud Monitoring infrastructures and more. The steps to use the core model for a specific method consists of document writing that gives new sub-types of core entities with related features. In OCCI terminology, the document is an extension of the core schema.

3.2 An OCCI extension for IoT

From the informal model, sensor monitor the real world object on the alternate of an aggregator. In OCCI manners, both the aggregator and real world entity are OCCI resources, and the sensor is an OCCI link. The aggregator is the subcategory of the core resource type and a sensor connect between the resources.

The Fig. 2 shows the class diagram. Specific attributes not introduced for the new OCCI entities, but it expects that suitable mixins are defined by the provider to change an entity for a user's needs. The description of the available mixins provided to the user in response to GET request, which is sent to the OCCI server: available mixins are discoverable. Selected mixins connect with the sensor instance, the user implemented a sensor with the needed functionality.

The link attribute is an array of unique element of Negative Temperature Coefficient (NTC) sensor and the sensor rendering inside an aggregator. NTC mixin is acquired from two attributes: period and OUTlevel, an entity port.

The existing communication maintain in the National Institute of Standards and Technology (NIST) document and these represent by an entity port in these methods. Bi-directional communication channel is a port with matching identifiers shares the same channel. Bi-directional communication is identified by a port, which has a string of attribute. The sensor has the ability to connect with another layer of the communication represented by the null mixin, which connect one of the lower level layer to another port of the upper level layer aggregator.

3.3 Temperature monitoring

On one side, it shows that the formal schema is enough to describe by an expression and guide the deployment of the simple use case. The performance of a Representational state transfer (REST)-based deployment is compared with the other technique. The prototype is required to focus on the important aspects of this proposal.

- OCCI-IOT data structure guides the deployment
- Websockets use in the infrastructure for data transport

The other aspects are deliberate especially, in order to neglect the introduction of spurious variables in the networking. This prototype is a typical smart agriculture use case stated as follows.

IoT is deployed in the field that collects the ambient measures, which controls air condition and watering in the greenhouses. These devices are fixed across the field to possible distance of the plants. IoT devices organize as three layers, greenhouses

present in the real world objects layer and this connects to the upper layer by sensors, which sends raw data to the devices on the intermediate layer. This layer consists of aggregators that convert the raw data into temperature and filter is applied to the sensor data. The experimental result of this method is uploaded to a ThingSpeak server, which is the external server meeting RQ5 and this collects and displays the data from all aggregators and may activate alarms and also schedules the events. It is an aggregator in the OCCI-IOT paradigm.

To develop the above system, six types of mixin require.

- In order to describe the temperature meter, three sensor mixins and NTC are required, null connect the intermediate aggregator to the centralized server. Dummy is not connected to a real measurement device and used for stress the aggregator;
- The two aggregator mixins are NTC2Degrees and ThingSpeak. NTC2Degrees for the intermediate aggregator and ThingSpeak are connected to the ThingServer;
- A real world object mixin for the greenhouse.

This method focused on the temperature meter and the intermediate aggregator. The Table 1 presents the temperature meter mixin. NTC thermistor is an inexpensive temperature sensor, which is used as an input devices and reveals by the term attribute. The provider defines the mixin definition of the sensor scheme, which is based on the sensor mixins subtype denoted in an OCCI extension for IoT. It can be related only with a sensor link defined in the same IoT document. The mixin attributes are prefixed with the provider domain: com.example.

- The time interval to configured between successive evaluation is in milliseconds.
- level_out indicates the id of the output channel related with the OUTlevel port: the value comes from the output of a A/D converter connected to the NTC device.
- Uri is the URI of the remote measurement device hosting a WebScoket server.

Table 1. The json rendering of the temperature sensor

```
{" term " : "NTC" ,
" scheme " : " h t t p : / / example . com / s e n s o r # " ,
" depends " : " h t t p : / / schemas . ogf . org / o c c i / i o t # s e n s o r m i x i n s " ,
" a p p l i e s " : " h t t p : / / schemas . ogf . org / o c c i / i o t # s e n s o r " ,
" a t t r i b u t e s " :{
"com . example .NTC . p e r i o d " :{ " t y p e " : " number " } ,
"com . example .NTC . l e v e l o u t " :{ " t y p e " : " s t r i n g " } ,
"com . example .NTC . u r i " :{ " t y p e " : " s t r i n g " }
} ,
" t i t l e " : "NTC" ,
" l o c a t i o n " : " / s e n s o r /NTC"}
```

Table 2. The json rendering of the aggregator mixin

```
{" term " : " NTCtoDegrees " ,
" scheme " : " h t t p : / / example . com / a g g r e g a t o r # " ,
" depends " : " h t t p : / / schemas . ogf . org / o c c i / i o t #
a g g r e g a t o r m i x i n s " ,
" a p p l i e s " : " h t t p : / / schemas . ogf . org / o c c i / i o t # a g g r e g a t o r " ,
" a t t r i b u t e s " :{
"com . example . NTCtoDegrees . n t c i n " :{ " t y p e " : " s t r i n g " } ,
"com . example . NTCtoDegrees . d e g r e e s o u t " :{ " t y p e " : " s t r i n g " } ,
"com . example . NTCtoDegrees . g a i n " :{ " t y p e " : " number " } ,
"com . example . NTCtoDegrees . p e r i o d " :{ " t y p e " : " number "}
} ,
" t i t l e " : " NTCtoDegrees " ,
" l o c a t i o n " : " / a g g r e g a t o r /NTC"}
```

The aggregator mixin processes the temperature data and creates the collaborate successive value which is shown in the Table 2. The raw data collected from the input channel NTC-in and convert into degrees using a logarithmic interpolation of the characteristic function of the NTC. Data in the history combined using an exponentially weighted moving average and, at regular intervals of seconds, the current value is transferred across the degrees_out channel.

Using the above mixins, it is possible to define an architecture with an arbitrary number of sensors: the system smoothly scales, when the number of aggregators in the intermediate layer increases the capacity of the ThingSpeak aggregator and further layer can also be added.

4. Experimental study

The prototypes that implemented in the devices are the eUtilities, according with NIST terminology, which are an Arduino Duemilanove for the measurement device and Rashberry Pi for the aggregator. These are two low cost devices that help to implement the study and the Arduino contains an Ethernet shield. The switched Ethernet network used in this study because it provides minimal interference on measurements that requires to function. The wireless devices are most commonly used devices in this green field, but it barely introduces uncertainty into experiments, without technical added value.

WebSocket server executed by the Arduino that offers the measurements of the temperature at a constant rate. The low level interrupts used in these experiments to initiate data production due to timing issues. The temperature sensor is the voltage divider consists of one fixed resistor and a negative temperature coefficient (NTC) resistor. Raspberry Pi 3 used for hosting aggregator device, which is in credit card-sized single-board computer depend on a powerful processor of ARM that is quad core at 1.2 GHz. The operating system used in this implementation powered by the Linux. This connects through the public Internet across a NAT router in order to access the public ThingSpeak server.

According to Center for Applied Internet Data Analysis (CAIDA) the dashboard ThingSpeak server hosted on the Amazon Web Services (AWS) autonomous system. The numerical data can be uploaded in the ThingSpeak that allowed by a REST Application Programming Information (API), these can be used to monitor and initiate the actions. The PC connected to the intranet sends the data to the aggregator, from the request of spawn's dummy measurement device. The prototype of the device is a pipe consists of sensor link (ntc), an aggregator (to measure the temperature), another sensor link (s1), and another aggregator (TS). In Fig. (3), every component indicates the instance identifier, its OCCI-IOT type, and the associated mixins: the syntax is outlined in the caption. Every component consists of OCCI-IOT represents in JSON. The temperature aggregator representation is shown in the Table 3. The outgoing NTC sensor link connection depends on the OCCI technique.

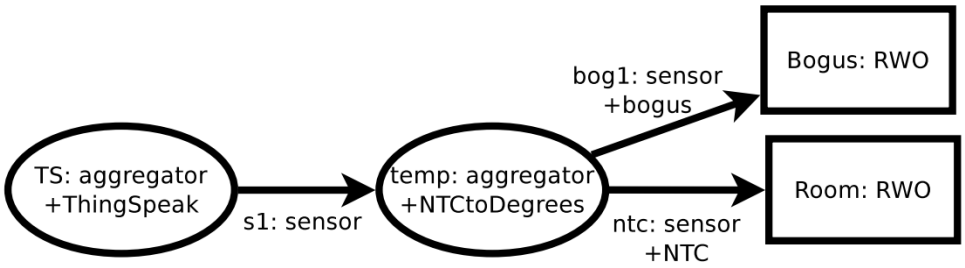

Figure.3 Graphical representation of the system deployed in the prototype. Syntax: < id >:< type > [+ < mixin >]∗

Table 3. OCCI-json rendering of the temp aggregator with the outgoing ntc sensor

```
{
"kind":"http://schemas.ogf.org/occi/iot#aggregator",
"mixins":[
"http://example.com/aggregator#NTCtoDegrees"],
"attributes":{
"com.example.sensor.NTCtoDegrees":{
"ntcin":"channel1",
"degreesout":"channel2",
"gain":16,
"period":30gg,
```

```
"id":"temp",
"links":[
f"kind":"http://schemas.ogf.org/occi/iot#sensor",
"mixins":["http://example.com/sensor#NTC"],
"attributes":{
"id":"ntc",
"target":{
"location":"/room/MyRoom",
"kind":"http://schemas.ogf.org/occi/iot#rwo"},
"source":{
"location":"/aggregator/temp1",
"kind":"http://schemas.ogf.org/occi/iot#aggregator"
},
"com.example.sensor.NTC":{
"OUTlevel":"channel1",
"period":1,
"uri":"ws://192.168.113.177"}
gg]}
```

5. Experimental result

Different input parameter is given to the system and the output is uploaded to the ThingSpeak, that helps to monitor the field. The output has been calculated in the form the various parameters and these parameters are represented in the graphical terms. A brief explanation about the simulated result of this method presented in this section with the different types of output. The greenhouse is taken as the case study, in order to evaluate the performance of the OCCI-based architecture. OCCI technique is applied and REST-ful interface is provided to help the user to interact with the system. The Arduino and Raspberry Pi 3 are the two important devices used in this method. The aggregator connects with the sensor, which sends the raw data to the device and the temperature is monitored using this technique. The experiment result of this method uploaded in the ThingSpeak server. There are various parameters such as temperature, RTT, latency, clock difference, and frequency evaluated in this technique. The formula for measuring RTT and latency can be measured from the following Eqs. (1) and (2).

$$RTT = (\alpha . Old\_RTT) + ((1 - \alpha) . New\_Round\_Trip\_Sample) \quad (1)$$

Where α is constant weighting factor ($0 \leq \alpha < 1$).

$$latency = clock\ cycle\ time \times number\ of\ clock\ cycles \quad (2)$$

The temperature is measured by the sensor and uploaded to the ThingSpeak server, which can be accessed to monitor the field area parameter. The graphical representation of the greenhouse is shown in the Fig. 4. This shows that the temperature varies across the time and these are recorded by the sensor. The greenhouse temperature can be constantly monitored by this method with less delay in the signal transfer. The jitter shows the delay in the server and the latency gives the delay by the Raspberry Pi 3. The temperature is monitored for 100 seconds and these plotted in the graph with time as X-axis and Temperature as Y-axis.

The RTT is measured to analyze the efficiency of the system and this gives the time taken for sending the signal and acknowledge for receiving signals. The time required to send the signal and acknowledge is within one second. The RTT is shown in the Fig. 5. The maximum time taken by the system to send and acknowledge is 0.96 seconds. This gives the performance of the method in the cloud and edge devices. Various numbers of correct operation rounds are sending across the WebScoket and receiving the acknowledgement and the measurement device stops sending the data.

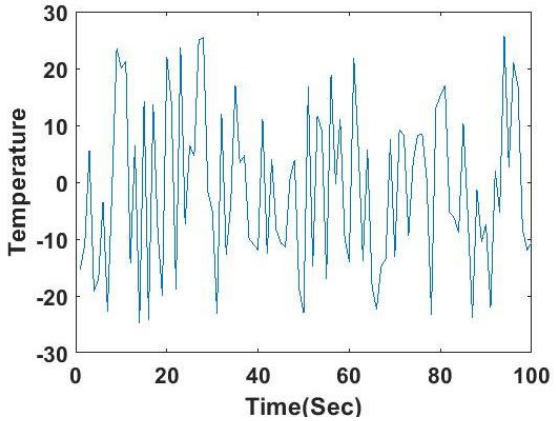

Figure.4 Temperature recorded in the Greenhouse

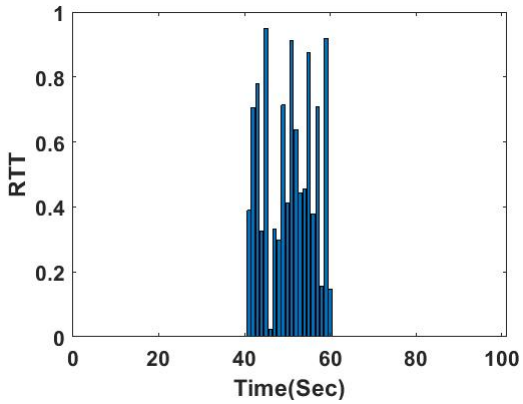

Figure.5 Round trip time for the system

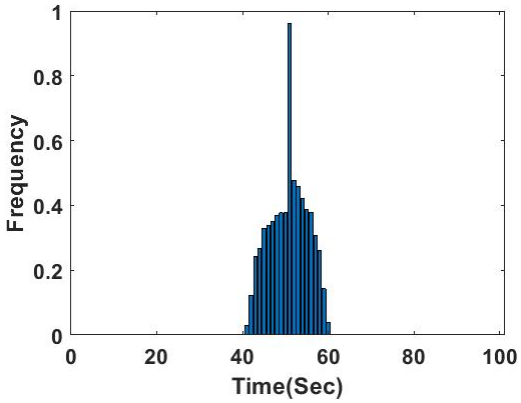

Figure.6 Jitter of event on ThingSpeak server

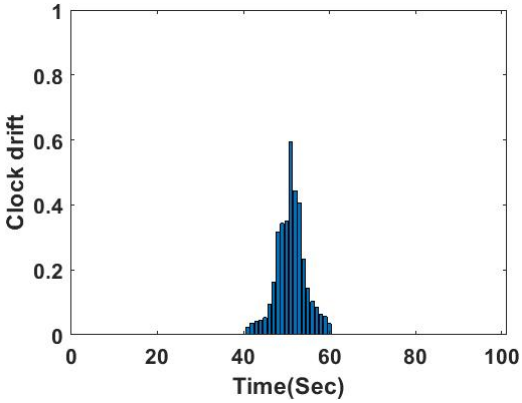

Figure.7 Clock drift of the method

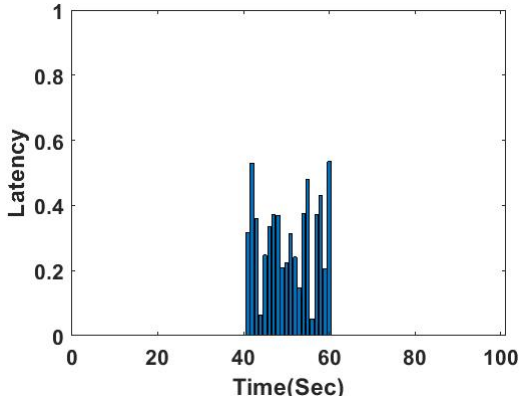

Figure.8 Latency parameter observed in the technique

The graphical representation of jitter is shown in the Fig. 6, which gives the delay in the ThinkSpeak server. Evaluating the packet, it is observed that the Arduino breaks the connection. The problem caused due to packet loss and WebScoket hang, which are difficult to compensate and also causes different result in the different Ethernet shields.

Conventional Named Data Networking (NDN) with heterogeneous traffic technique transfers the data in IoT system and involves in delay. In the research [12], prioritize the IoT traffic data by using MDP technique and data has been retrieved. This allow the data to transfer to the system with lower latency. Traffic class and the Timestamp are added in the data and interest packet helps to prioritize the traffic. The data scheduling technique helps to achieve the higher performance in this function. The research has been compared with the OCCI-based architecture to analysis the performance.

The one-way delay can be measured by using a significant clock drift and clock drift occurred between the time period of 40 to 70 seconds. The clock drift occurred in 0.6 Sec and this shows the effectiveness of the system. The clock drift with respect to time is plotted in the graph which is shown in the Fig. 7. The latency gives the delay occurs in the Raspberry Pi 3 while updating the parameters in the cloud (ThingSpeak). The latency in related to time (sec) of the given system presented in the graphical representation of Fig. 8. This shows the effectiveness of the system using the REST method in the OCCI. In Table 4, RTT of this method is compared with an existing methods and this technique achieved less RTT than existing methods. The OCCI-based architecture manages the resources using the devices to schedule the data, while existing method schedule the data based on the MDP technique in same environment. The existing methods and proposed method are simulated in the same data and compared with each other. The monitoring manager allocate the resource to the devices and data are passed from sensor devices to ThingSpeak.

The OCCI-based architecture is compared with existing method [16] and [17] in terms of response time. The optimization technique has been applied in the cloud services to improve the performance of the cloud management [16]. The two types of optimization techniques are used in the research [16] for identifying optimal number and optimal placement in service management application and in cloud respectively. The optimization technique has been applied to Automatic managers (AMs). AMsOPTIMIZATION is proposed for finding optimal number in service based application, which

has the capacity to run in parallel and the set of waiting service will run after receiving data transmission. AMsASSIGNMENT algorithm is proposed for assigning the services in the cloud and this algorithm also runs in parallel. In order to avoid the bottleneck problem, this technique will assign the task in the cloud while two service running in parallel. The extension has been proposed to the OCCI to support the different aspect of autonomic computing [17]. Automatic Manager has been introduced to allocate the resource to enable automatic management in cloud. Initiate of the method involves in monitoring the data and applies the analysis rules. The ActionLink has been used the manager to reconfigure the actions and applies them on Resources. The OCCI-based architecture method has the different architecture compare to the research [16] and [17] in same environment, to increase the performance by proper assignment.

Table 4. RTT with existing system

Iteration Number	Optimal technique [16]	Automatic manager and Resource manager [17]	MDP-IoT [12]	OCCI-based architecture
20	26	26	24	22
40	30	28	27	24
60	35	32	28	25
80	38	37	32	28
100	42	39	34	31
120	64	58	48	42

5.1 Comparative analysis

The OCCI-based architecture tends to reduce the data traffic in the IoT environment and effectively transmit the data between the devices. Comparison between the OCCI-based architecture and existing helps to understand the effectiveness of the OCCI-based architecture. The important aspect to consider in the resource management method is response time and other parameters are compared with existing system which is shown in Table 5. The different parameters are simulated and calculated for the 500 client request.

The parameters that are measured showed that the OCCI-based architecture is effective compared to the other existing methods in monitoring greenhouse. Different measures are made and evaluated in this method, which gives the efficiency of this method.

Table 5. Different parameters compared with existing system

Parameters	Optimal technique [16]	Automatic manager and Resource manager [17]	MDP-IoT [12]	OCCI-based architecture
RTT	14104	792	746	621
Latency (average delay in ms)	36	32	20	17
Clock drift	1.4	1.2	0.8	0.6
Jitter (ms)	1.68	1.52	1.26	0.97

6. Conclusion

The aim of this research is to provide the optimized method for the monitoring technique through IoT. OCCI is applied in this technique and this can be easily extendable and it provides an interface between the various layers of the system. The two major hardware used in this method is Arduino and Raspberry Pi 3. In order to evaluate the performance of the proposed OCCI technique, the greenhouse is used as the case study. The sensor is connected to the various part of the study area and this provides the raw data to the system. Using these data, temperature is monitored through the Internet and the optimization made by this technique. Simulation result of the OCCI-based architecture system uploaded to the ThingSpeak server and this is an external server. The RTT achieved up to 0.96 seconds and delay has been reduced. The purpose is to contribute standard production for IoT infrastructures and components, to foster an open competition in a fast growing market.

References

[1] H. Qin, W. Chen, B. Cao, M. Zeng, and Y. Peng, “A cross-interface design for energy-efficient and delay-bounded multi-hop communications in IoT”, *Ad Hoc Networks*, Vol.70, pp.103-120, 2018.

[2] F. Karim and F. Karim, “Monitoring system using web of things in precision agriculture”, *Procedia Computer Science*, Vol.110, pp.402-409, 2017.

[3] J.F. Mendoza, H. Ordóñez, A. Ordóñez, and J.L. Jurado, “Architecture for embedded software in microcontrollers for Internet of Things (IoT) in fog water collection”, *Procedia Computer Science*, Vol.109, pp.1092-1097, 2017.

[4] N. Sahraei, S. Watson, S. Sofia, A. Pennes, T. Buonassisi, and I.M. Peters, "Persistent and adaptive power system for solar powered sensors of Internet of Things (IoT)", *Energy Procedia*, Vol.143, pp.739-741, 2017.
[5] G. Severino, G. D'Urso, M. Scarfato, and G. Toraldo, "The IoT as a tool to combine the scheduling of the irrigation with the geostatististics of the soils", *Future Generation Computer Systems*, Vol.82, pp.268-273, 2018.
[6] A. Perles, E. Pérez-Marín, R. Mercado, J.D. Segrelles, I. Blanquer, M. Zarzo, and F.J. Garcia-Diego, "An energy-efficient internet of things (IoT) architecture for preventive conservation of cultural heritage", *Future Generation Computer Systems*, Vol.81, pp.566-581, 2018.
[7] D.Y. Kim, S. Kim, H. Hassan, and J.H. Park, "Adaptive data rate control in low power wide area networks for long range IoT services", *Journal of Computational Science*, Vol.22, pp.171-178, 2017.
[8] M. Wei, S.H. Hong, and M. Alam, "An IoT-based energy-management platform for industrial facilities", *Applied energy*, Vol.164, pp.607-619, 2016.
[9] F. Al-Turjman, "Information-centric framework for the Internet of Things (IoT): Traffic modeling & optimization", *Future Generation Computer Systems*, Vol.80, pp.63-75, 2018.
[10] A. Fensel, D.K. Tomic, and A. Koller, "Contributing to appliances' energy efficiency with Internet of Things, smart data and user engagement", *Future Generation Computer Systems*, Vol.76, pp.329-338, 2017.
[11] Y. Zhang and S. Sun, "Real-time data driven monitoring and optimization method for IoT-based sensible production process", In: *Proc. of the 10th International Conf. on Networking, Sensing and Control*, pp.486-490, 2013.
[12] S. Muralidharan, A. Roy, and N. Saxena, "MDP-IoT: MDP based interest forwarding for heterogeneous traffic in IoT-NDN environment", *Future Generation Computer Systems*, Vol.79, pp.892-908, 2018.
[13] S. Liu, G. Zhang, and L. Wang, "IoT-enabled Dynamic Optimisation for Sustainable Reverse Logistics", *Procedia CIRP*, Vol.69, pp.662-667, 2018.
[14] S. Jeon and I. Jung, "Experimental evaluation of improved IoT middleware for flexible performance and efficient connectivity", *Ad Hoc Networks*, Vol.70, pp.61-72, 2018.
[15] H.R. Arkian, A. Diyanat, and A. Pourkhalili, "MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications", *Journal of Network and Computer Applications*, Vol.82, pp.152-165, 2017.
[16] L. Hadded, F.B. Charrada, and S. Tata, "Optimization and approximate placement of autonomic resources for the management of service-based applications in the cloud", In: *Proc. of International Conf. on the Move to Meaningful Internet Systems*, pp.175-192, 2016.
[17] M. Mohamed, D. Belaïd, and S. Tata, "Extending OCCI for autonomic management in the cloud", *Journal of Systems and Software*, Vol.122, pp.416-429, 2016.