Energy Preserving Remote Supervised Room - An Improved Room Automation System

Gaurav Tripathi[1], Damanpreet Singh[2]

[1] PG Scholar, Department of Computer Science & Engineering, SLIET, Longowal, India

[2] Associate Professor, Department of Computer Science & Engineering, SLIET, Longowal, India

Email: [1]gt50@live.in, [2]damanpreets@sliet.ac.in

Abstract

Smart environment represents intelligent automation of things by processing contiguous sensor data. The main challenge of smart environment is the availability of a contiguous data in the implemented area. Wireless Sensor Network (WSN) is formed to avail this contiguous data from distributed sensor nodes. Smart environment is an implementation of Internet of Things (IoT), that is a networked interconnection of quotidian objects. Various IoT applications such as smart city, smart home and room automation has a prime focus on minimization of electricity consumption. In this paper the idea of Energy Preserving Remote Supervised Room Automation System (EPRS-RAS) is proposed as the application of IoT to minimize the electricity and to provide remote access on the room. The proposed EPRS-RAS consists of WSN, Microcomputer and web application. The architecture and methodology of EPRS-RAS is also discussed in this paper. Further this paper concludes with the discussion on results and limitations of the proposed system.

Keywords: *Automation, electricity, home, IoT, remote, room, smart, supervised, WSN.*

1. Introduction

The electricity consumption is increasing roughly in an exponent pattern [1] [2]. It is observed that due to lackluster attitude by the society, in general some of the used electricity is wasted due to negligent used. Countless energy efficient systems are introduced to minimize energy consumption in different sized places such as city, home, office, room, etc. Among these systems IoT offers some of the best solutions such as smart city, smart home, smart industry, etc. These smart environments provide intelligent automation of objects as an implementation of IoT.

IoT is a network of quotidian objects. IoT consists of three phases, a WSN, Data Processing and Cloud [3]. The applications of IoT is raising the living standard of people and solving real world and daily life problems. Smart Home [4], Smart city [5], Smart grid [6], [7], Health Monitoring [8], Smart Agriculture [9], Smart Transportation [10], etc. are the widely implemented applications of IoT. Most of the research works of IoT are ongoing in these fields [11].

The most popular application of IoT nowadays is a Smart Home, basically introduced for personal in-home usage. Smart Home is where every entity of the house is connected to a common network to work simultaneously. For example, your alarm clock is connected to your geyser, your alarm starts ringing at 6:00 AM and when you wake up your geyser automatically switched on so that, by the time you reach to the shower warm water is ready for you. This example shows the improved lifestyle by saving the time and minimization of electricity wastage.

As a segment of Smart Home, Room Automation System (RAS) provides the automation of in-room appliances to achieve increased comfort and energy efficiency. The existing applications of RAS provide an architecture where automation is achieved by using Heat/Light/PIR sensors; also provide remote access using Bluetooth, Local Area Network (LAN) and Thingspeak platform.

The proposed EPRS-RAS provides improved comfort and energy efficiency by creating an automation environment in a room. The WSN provides data to the EPRS-RAS that is processed by a microcomputer for the automation of electric appliances in the room. Concurrently, a web application is served by the system to provide remote access over the internet for Real-time data visualization and remote switch to control the appliances.

The system is designed by the installation of WSN in the room, the WSN provides the sensed data of the room environment to the microcomputer placed in the same room. The microcomputer is responsible for providing an online web interface to the user and to control the relays connected to the switches of the room. WSN is using Adhoc On-Demand Distance Vector (AODV) routing protocol for data transmission, one node is the head of WSN which is the destination of all other sensor nodes. The microcontroller of the head node processes received data from all the nodes and sends it to the microcomputer with a proper data structure. The system is programmed using python and C defining the rules for automation of appliances. The database is maintained by the insertion of sensor and switch data.

The rest of the paper is organized into four sections. Section 2 Prototyping presents the design and development methodology of EPRS Room prototype. Section 3 Methodology presents the design strategy and development phases of the EPRS-RAS containing the implementation methodology in software level and hardware level. Section 4 Result and Discussion presents the discussion on the changes observed after the implementation of the EPRS-RAS and its comparison with the other existing RASs. Finally, this paper concludes with section 5, also with a discussion on possible future work.

2. Design and Emulation

For the detection and prevention of possible errors and faults before implementation of EPRS-RAS, the prototype is designed. The same methodology is used for the prototyping of EPRS-RAS in small-scale. Limited hardware and software resources are used to develop the prototype to minimize the loss in case of any failure occurred. The EPRS Room prototype is equipped with a microcomputer, a relay board, and two sensor nodes. There are following phases of the development of prototype.

2.1 Hardware Components

EPRS-RAS consists of three levels of hardware input, processing and output. Input is provided by the sensors, microcomputer processes the input data, and the output is realized by the relay in real environment as shown in Figure 1.

2.1.1 Sensor and Microcontroller: The sensor is a tiny node which is used to detect and record the changes in environmental conditions. SENSEnuts module is embedded with sensors and a microcontroller. SENSEnuts is a stacked module of sensors, gateway, and a microcontroller. This module supports IEEE 802.15.4 ZigBee that makes it a better option for WSN because of the less energy consumption [12].

2.1.2 Microcomputer: A small sized, fully functional and inexpensive computer is introduced in 2013 for educational purpose in the United Kingdom became popular worldwide as Raspberry Pi [13]. It is used in EPRS-RAS as it can handle 28 relays at a time and the requirement here is to control 24 appliances [14].

2.1.3 Relay Board: Relay is an electromagnetic device works as a switch to electric appliances. Microcomputer gives signal to relay and on the other side it is connected to the electric appliance that is controlled according to the signal provided by the microcomputer [15].

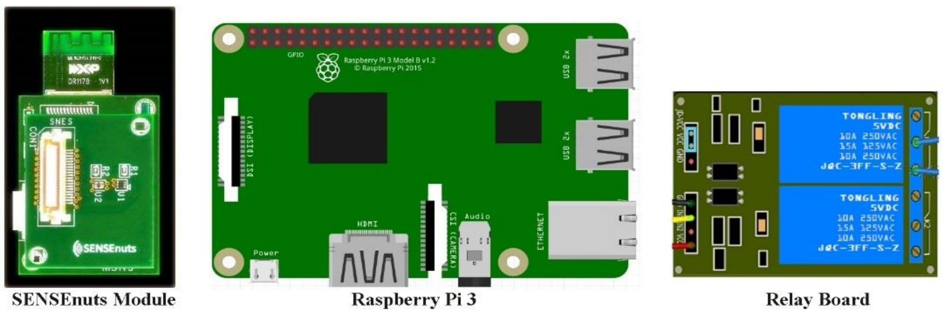


Figure 1 Hardware components of EPRS-RAS

2.2 Software Components

2.2.1 C Language: C is considered as the most suitable programming language for embedded systems. In EPRS-RAS C language is used to program the microcontroller attached with the sensors insures the working of sensors and the data transmission in WSN [16]. The functions used in C program are as follows:

- **startNode():** The Program execution starts from this function.
- **userRcvDataPkt(payload, length, prevAdd, linkQuality):** This function is called when a data packet from the network is received.
- **addTask(taskAdder, taskType, time):** This function adds a user-defined time-bound task which is executed at the end of the time specified by “time” in the parameters.
- **routingSendData(*data, len, destAddr):** Send data packet pointed to destination address via route found by the routing protocol.
- **tmpInit():** This function initializes the communication interface between the temperature sensor and Radio Module.
- **lightSensorInit():** This function initializes the communication interface between the light sensor and Radio Module.
- **macInit():** Initializes the MAC layer for wireless communication.
- **readTmp():** Reads the temperature data from the temperature sensor and returns it in 8-bit format.
- **readLux():** Reads the light intensity in lux and returns in 16-bit format.

2.2.2 Python: Python is known for its simplicity and ability for its vast usage area [17]. Python has a very rich library, out of which four packages are used in EPRS-RAS. Packages serial, struct, and sqlite3 are used to perform communication with serial port, define and decode the data structure, and to manage the database respectively. RPi.GPIO is used to send signals to the GPIO pins.

2.2.3 SQLite 3: It is very simple RDBMS since it doesn't need to run any separate server. It is also possible to make a prototype using SQLite3 and then porting the code to a bigger database like Oracle or PostgreSQL [18].

2.2.4 Node.js: Node.js is very popular server-side scripting language based on JavaScript. Node supports a very special feature of non-blocking code [19]. Because of its simplicity it is used to serve the web application over the internet. From its rich library NPM packages such as HTTP, FS and SQLite3 are used to handle HTTP request and response, interaction with file system, and to manage the database respectively.

2.2.5 Socket.io: Socket.io is used to provide Realtime data to the client side. In EPRS Room web application socket.io is used to send the switch value and to receive the sensor data in real-time that enables the full functionality of the web application without reloading it even once [20].

2.3 Architecture

There are three layers in the architecture of EPRS-RAS as shown in Figure 2. Perception layer consists of the hardware installed in real environment such as sensors, microcomputer and relays. Network layer consists of communication mediums such as ZigBee, Wi-Fi, LAN and Internet. Application layer consists of the output generated by the working of whole system such as automation and web application.

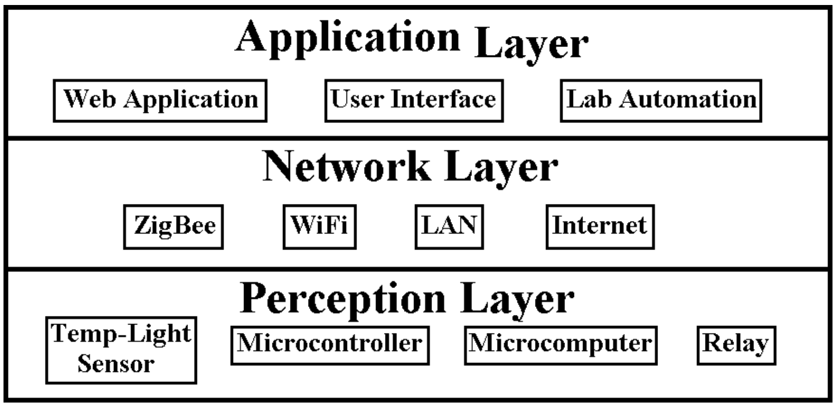


Figure 2 Layered architecture of EPRS-RAS

2.4 Prototype of EPRS-RAS

Before developing the EPRS-RAS a prototype of this system is designed to detect and prevent possible errors in small-scale. The prototype contains two sensors one for sensing and transmitting the data, another for sending this data to the serial port. This system has a small-scale database and a simple web interface for Realtime data display and switch control. The prototype includes all the technologies used in the actual model of EPRS-RAS. The prototype has a WSN, serial port communication and GPIO pin control same as the EPRS-RAS. The circuit diagram and pictorial representation of the prototype is shown in Figure 3.

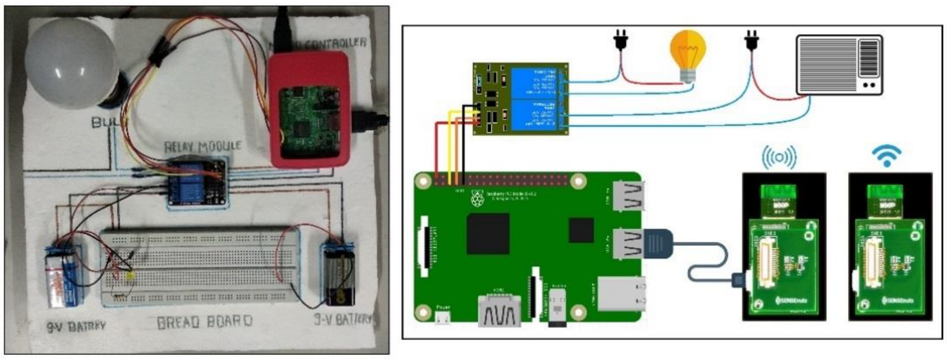


Figure 3 Picture and Circuit diagram of the Prototype

3. Methodology

After the successful implementation of the prototype, its testing and debugging the actual EPRS-RAS is developed. This system is more complex than that of the prototype. The development involves a circuit system of connection between more than 30 hardware, complex programming of the WSN consisting AODV routing protocol and a web application with better User Interface.

3.1 Hardware Setup

EPRS-RAS is equipped with very complex hardware circuit design including five nodes in WSN wirelessly connected to each other using ZigBee, pan-coordinator being head of the WSN is receiving data from all the coordinator nodes and it is associated with the microcomputer through the USB. A microcomputer as a central hub takes control over everything and behaves like a central processing unit for EPRS-RAS. There are two relay boards of size 12 and 8 are connected to the fluorescent light and fans to control it according to the EPRS-RAS and four relays are separately connected to the ACs of the room as shown in Figure 4.

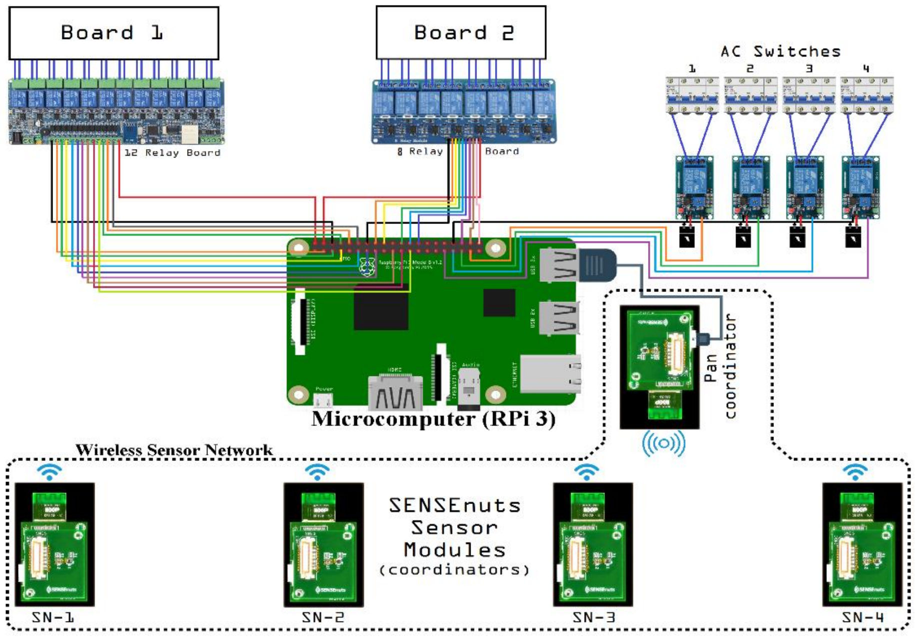


Figure 4 Circuit diagram of EPRS-RAS

3.2 Software Setup

Two main programs run in a synergy to run the EPRS-RAS. One is the *server.js* that is a node.js script works for the interaction between the client and server. The second one is *smartRoom.py* that is a python program works for receiving the input from the WSN and controlling GPIO pins. Software setup includes programming WSN, serial port communication, web application design, GPIO pin control, and AODV routing protocol. The algorithms and pseudo codes are described below.

- **Wireless Sensor Network:** In WSN the coordinator and pan-coordinator nodes are programmed as shown in Pseudocode 1 of *broadcast_coord.c* and Pseudocode 2 of *broadcast_pancoord.c*.

Pseudocode 1: *broadcast_coord.c*

```
1.set packet[4] = {PCKT_TYPE, temp,
                  light_hi, light_lo}
2.send packet to mac layer
3.hold 1 second
4.go to step 1
```

Pseudocode 2: *broadcast_pancoord.c*

```
1. if (data packet from network is received)
2.     check if payload[0] = PCKT_TYPE
3.     temp = payload [1]
4.     light = payload [2]<<payload[3]
5.     send temp, light to gateway
6. end if
7. go to step 1
```

- **Serial Port Communication:** The pan-coordinator node forwards the sensor data to the microcomputer. This communication between USB gateway of sensor module and the microcomputer is performed by the serial port. Pseudocode 3 shows the serial port communication.

Pseudocode 3: read_serial() in SmartRoom.py

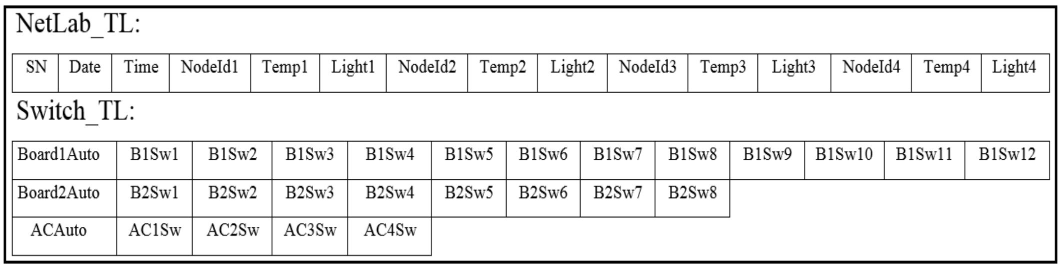
```
1. open serial port '/dev/ttyUSB0'
2. set baud rate = 115200
3. set DTR = false
4. set RTS = false
5. while true:
6.     read 12 bytes value from the serial port
7.     unpack the 12 bytes data
8.     insert temp, light and NodeId into the database
9. end while
```

- **Switch Control by GPIO:** The microcomputer used in EPRS-RAS has inbuilt GPIO pins. The GPIO pin holds the binary signal as 1 or 0. The relay is switched on/off by sending the signal to the associated GPIO pin. Pseudocode 4 shows the pin how the program is controlling the pins in order to control the appliances.

Pseudocode 2: *switch_ctrl()* in SmartRoom.py

```
1.  if switchAuto[0]==0 then                                        // Board1 auto mode
2.      control GPIO(3,5,7,8, 10,11,12,13,15,19,21,23) According to temp/light values
3.  end if
4.  else                                                            // Board1 user mode
5.      control GPIO(3,5,7,8,10,11,12,13,15,19,21,23) According to Switch_TL values
6.  end else
7.  if switchAuto[1]==0 then                                        // Board2 auto mode
8.      control GPIO(16,18,22,24,26,36,38,40) According to temp/light values
9.  end if
10. else                                                            // Board2 user mode
11.     control GPIO(16,18,22,24,26,36,38,40) According to Switch_TL values
12. end else
13. if switchAuto[2]==0 then                                        // Air Conditioners auto mode
14.     control GPIO(37,35,33,31) According to temp value
15. end if
16. else                                                            // Air Conditioners user mode
17.     control GPIO(37,35,33,31) According to Switch values
18. end else
```

- **Database:** The database is created using SQLite3. Tables in the database are NetLab_TL and Switch_TL. NetLab_TL stores the sensor data with sensor id and timestamp, the Switch_TL stores the switch data from the web application of EPRS-RAS. The database structure is shown in Figure 5.



NetLab_TL:

SN	Date	Time	NodeId1	Temp1	Light1	NodeId2	Temp2	Light2	NodeId3	Temp3	Light3	NodeId4	Temp4	Light4

Switch_TL:

Board1Auto	B1Sw1	B1Sw2	B1Sw3	B1Sw4	B1Sw5	B1Sw6	B1Sw7	B1Sw8	B1Sw9	B1Sw10	B1Sw11	B1Sw12
Board2Auto	B2Sw1	B2Sw2	B2Sw3	B2Sw4	B2Sw5	B2Sw6	B2Sw7	B2Sw8				
ACAuto	AC1Sw	AC2Sw	AC3Sw	AC4Sw								

Figure 5 Structure of database

3.3 Communication in EPRS-RAS

There are two types of data flowing in EPRS-RAS, sensor data and switch data. The sensor data is generated by the WSN and switch data is generated by the web application. Both the programs (server.js and smartRoom.py) are using switch and sensor data at the same time. The flow in EPRS-RAS is shown in Figure 6.

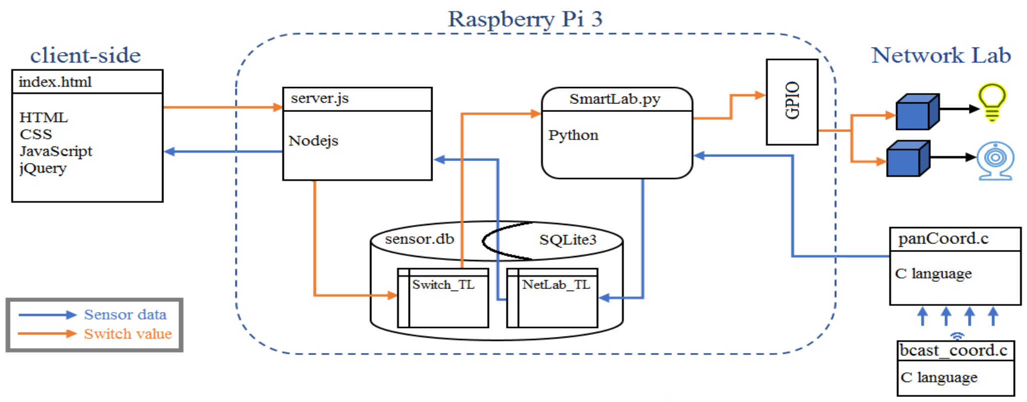


Figure 6 Data Flow in EPRS-RAS

4. Result and Discussion

The main goal of the EPRS Room was to save electricity in public sector buildings and provide remote control to the administrator. After its installation in the room, it started controlling electric appliances smartly, which results in saving in the wastage of electricity. After the calculations on existing data, it is concluded that the EPRS Room is saving electricity as well as the administration cost for everyday use of this room. The room has 15 fluorescent tube lights, 5 fans and 4 Air conditioners are sucking electricity uselessly even when there is no need of light and ACs at some corners of the room.

4.1 Power Consumption

A comparison of electricity usage is also performed in which the total usage of electricity in the room before and after the implementation of automation is calculated. According to Table 1, it is concluded that a total of approximately 12500 units of electricity is saved annually by the implementation of EPRS Room in one room. The electricity usage is reduced to about 45% from the normal room, when fully automated. By minimizing electricity consumption EPRS Room is indirectly reducing CO_2 emission by 2.8 tonnes annually.

Tabel 1 Electricity saving in room

Number of Appliances	Normal Room usage (per week)	EPRS Room usage (per week)	Total Saving (per week)
5 Fans (70 W each)	11 Units	7 Units	4 Units
15 Tube-lights (80 W each)	36 Units	20 Units	16 Units
4 Air Conditioners (3.9 KW each)	491 Units	273 Units	218 Units
Total Electricity usage	538 Units	300 Units	238 Units
Charges (₹7 per KWh)	₹3766 ≈ $56	₹2100 ≈ $31	₹1666 ≈ $25

4.2 Remote control

The web application of EPRS-RAS provides a user interface as shown in *Figure 7*. This web application displays the Real-time sensor data with one-second timeout and control switches to handle the room environment.

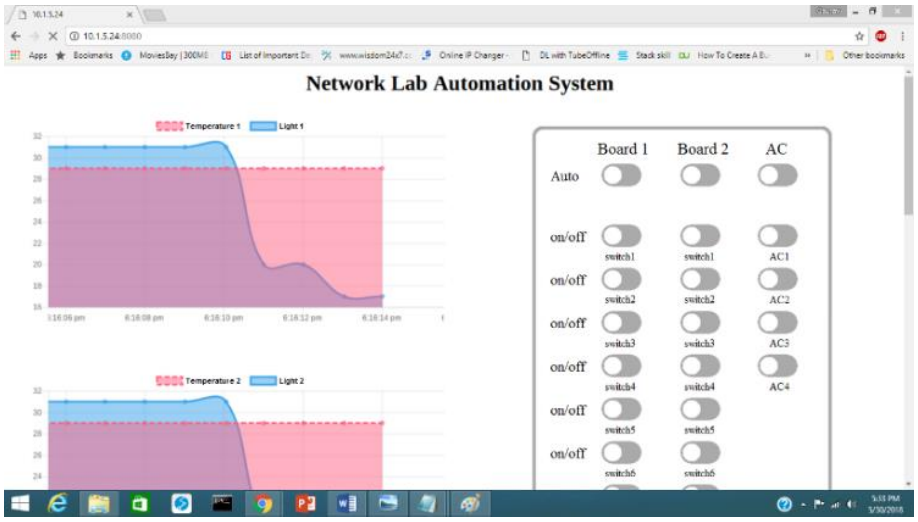


Figure 7 User Interface of the web application

4.3 Comparison with existing systems

Comparison between existing RAS and EPRS-RAS is shown in Table 3.

Table 2 Comparison between existing RAS and EPRS-RAS

	Vibhuti and Shimi [21]	Singh and Shimi [22]	Dementjev and Kabitzsch [23]	Proposed System (EPRS-RAS)
Sensor used	Heat, Light, PIR	Heat, Light, PIR	Heat, Light	Heat, Light
Remote Access	No	Visualization	Visualization	Control, Visualization
Microcontroller	Arduino Uno	ARM-7 Controller	NA	Raspberry Pi 3
Database	No	No	Yes	Yes
Data on Energy Consumption	No	Yes	Yes	No
Cloud	No	Thingspeak	No	Independent
Refresh Rate	NA	15 sec	NA	1 sec
Energy Consumption	Reduced	NA	32-36% Reduced	45% Reduced

The above comparison shows that the proposed EPRS-RAS provides Remote access for RAS control and data visualization and the refresh rate of data visualization is 1 second (customizable), that is more frequent than the other existing RAS. The EPRS-RAS provides more energy saving than other existing systems by the reduction of 45% energy consumption. The EPRS-RAS provids remote control on the room from a remote location through the internet that shows this system better than other in these terms.

5. Conclusion

This paper presented the EPRS Room architecture as a small-scale RAS, the role of IoT for energy efficiency, and the comparison with existing RASs. Various technologies and hardware such as C programming, Python, Node.js, microcomputer, sensors, and relays are used in the development of EPRS-RAS. Being a basic automation system EPRS Room has the possibility of many more improvements. The work can further be improved by using PIR Motion sensors and GPS positioning. Since these sensors are of very small memory and processing capacity the security is a very important aspect of this automation. Moving one step forward, security of whole automation system can be improved. Another work that is possible after this project is the use of machine learning which can more smartly control the appliances. Since the data of a long time is available in the database that data can further be processed to identify a pattern in the usage of the room, this pattern will help to improve the system. Opensource Home Automation can also be developed so that a user can design his own type of Smart Home. Since now every time a Smart Home is designed from bottom to top, open sourcing will make it use just like a plug and play device with a little use of programming. In short, the EPRS-RAS has many possibilities for future research and project works.

6. *References*

[1] Sustainable Energy Authority of Ireland, "Annual Report 2015 on Public Sector Energy Efficiency Performance," ***(2015)****.*

[2] The World Bank, "Electric power consumption (kWh per capita) | Data," IEA Statistics © OECD/IEA ***(2014)****.[Online].Available:https://data.worldbank.org/indicator/EG.USE.ELEC.KH.PC?locations=IN.*

[3] D. Evans, "The Internet of Things - How the Next Evolution of the Internet is Changing Everything," CISCO white Pap., no. April, pp. 1–11, ***(2011)****.*

[4] C. Fernandes, N. Clark, and T. Karliychuk, "Smart IoT Devices in the Home," IEEE Technol. Soc. Mag., no. June, pp. 71–79, ***(2018)****.*

[5] a Zanella, N. Bui, a Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," IEEE Internet Things J., vol. 1, no. 1, pp. 22–32, ***(2014)****.*

[6] S. M. Amin and B. F. Wollenberg, "Toward a Smart Grid," IEEE power energy Mag., no. october, pp. 34–41, ***(2005)****.*

[7] S. Massoud Amin and B. F. Wollenberg, "Toward a smart grid: power delivery for the 21st century," IEEE Power Energy Mag., vol. 3, no. 5, pp. 34–41, Sep. ***(2005)****.*

[8] S. M. R. Islam, D. Kwak, H. Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for Health Care : A Comprehensive Survey," Access, IEEE, vol. 3, pp. 678–708, ***(2015)****.*

[9] J. C. Zhao, J. F. Zhang, Y. Feng, and J. X. Guo, "The study and application of the IOT technology in agriculture," Proc. - 2010 3rd IEEE Int. Conf. Comput. Sci. Inf. Technol. ICCSIT 2010, vol. 2, pp. 462–465, ***(2010)****.*

[10] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Futur. Gener. Comput. Syst., vol. 29, no. 7, pp. 1645–1660, ***(2013)****.*

[11] *I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," Bus. Horiz., vol. 58, no. 4, pp. 431–440,* ***(2015)****.*

[12] *E. Technologies, "SENSEnuts Hardware Guide." Sensenuts University.*

[13] *"Raspberry Pi 3 Model B+" https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf*

[14] *"Usage - Raspberry Pi Documentation." http://www.raspberrypi.org/documentation/usage/*

[15] *Georgi Dalakov, "The electromechanical relay of Joseph Henry." [Online]. Available: http://history-computer.com/ModernComputer/Basis/relay.html.*

[16] *E. Technologies, "API Reference Guide." Sensenuts University.*

[17] *B. Simon Monk and B. Simon, "Programming the Raspberry Pi: Getting Started with Python,"* ***(2012)****.*

[18] *M. Owens, "Introducing SQLite," in The Definitive Guide to SQLite, Apress,* ***(2006)****, pp. 1–16.*

[19] *G. Pollack and C. Souza, "Real-time Web with Node.js," Code School. [Online]. Available: https://www.pluralsight.com/courses/code-school-real-time-web-with-nodejs.*

[20] *"Realtime application framework - socket.io," github. [Online]. Available: https://github.com/socketio/socket.io.*

[21] *S. L. Shimi and Vibhuti, "Implementation of Smart Class Room Using WAGO," 2018 2nd Int. Conf. Inven. Syst. Control, no. ICISC, pp. 807–812,* ***(2018)****.*

[22] *M. Singh and S. L. Shimi, "Implementation of Room Automation With Cloud Based Monitoring System," 2018 2nd Int. Conf. Inven. Syst. Control, no. Icisc, pp. 717–813,* ***(2018)****.*

[23] *A. Dementjev and K. Kabitzsch, "A CONSULTING MODULE IN ROOM AUTOMATION Alexander Dementjev, Klaus Kabitzsch,"* ***(2001)****.*