Implementation of Effective Heterogeneous IOT Message stability by Time Scheduling

A.D.RANI SIRISHA[1], M. GOPIKRISHNA[2]

[1]*PG Scholar, Dept of ES Vardhaman College Of Engineering(Autonomous) Hyderabad, Telangana,India*

[2]*Associate Professor Dept of ECE Vardhaman College Of Engineering(Autonomous) Hyderabad, Telangana, India*

Abstract- An Internet of Things (IoT) system options integration info from heterogeneous detector devices, permitting them to deliver a spread of perceived info through networks. Associate in Nursing IoT broker within the system acts as Associate in Nursing info exchange center, relaying periodic messages from heterogeneous detector devices to IoT shoppers. As additional devices participate within the IoT system, the service scale of entire system will increase. to beat the limitation of the amount of direct links to Associate in Nursing single operating component, the complete IoT system ought to be divided into several subsystems, referred to as IoT units, to create the system hierarchy. moreover, applying the service-oriented design (SOA) conception to comprehend the IoT service will facilitate to adapt the future-proof devices to the IoT system. This paper proposes Associate in Nursing IoT system skeleton and a shortest time interval (SPT) algorithmic rule for planning web-based IoT messages. The enforced planning theme supported by a priority queue model will effectively stablize the response messages from the scattered IoT sensors per every shopper request.

Keywords- Heterogeneous,Iot,SOA,SPT

I. INTRODUCTION

With the rapid development of network technologies over the past decade, a broader range of electronic devices can now support various network protocols with different physical connections to exchange data via the Internet. The Internet of Things (IoT) concept is based on these emerging technologies, in which everything is connected to a common network by an IoT platform to improve human communication and convenience. For example, the authors of implemented an IoT system which uses various sensors to identify food-related information and provide cooking suggestions.

In the traditional two-tier architecture, all IoT devices are connected to a single server. However, physical limitations of the server will prevent such a scheme from effectively scaling up or handling inputs from diverse devices. Thus, some studies have proposed three-tier architecture, adding a broker between the server and the devices. Usually, a broker is developed as a coordinator to connect different types of IoT components. Hence, the primary challenge lies in developing a broker capable of managing and accessing a variety of devices. To successfully blend with various associated elements, the broker should support multiple physical connections and protocol stacks. Furthermore, all brokers in the system are connected to a common network so that the IoT system scale can be enlarged by decentralizing these broker-centric subsystems, called IoT units, which aggregate information from many sensors.

A. OVER VIEW:-

To make the IoT system flexible and extendable, we adopt the Service Oriented Architecture (SOA) concept to develop the system. From the perspective of the developer-service-provider, SOA includes five features modularization, reuse, efficiency, loose coupling and division of responsibility. By using the SOA concept to develop the IoT broker, each sensor device is deemed as a service with a common communication interface. All devices are simply classified as input or output elements to become a Sensor or Actuator objects. Each device is encapsulated by a corresponding software component based on its functionality

B. PROBLEM STATEMENT:-

SOA based applications are often built using web based techniques such as HTML, CSS, JavaScript, Asynchronous JavaScript and XML (AJAX), allowing programmers to develop user interfaces that can be executed on most of end-user devices. Therefore, the web-based application development platform, like Apache, is good to be used for creating the IoT broker. Next, IoT clients send HTTP requests to the IoT broker to solicit information toward all frontend sensors. On the other hand, the Open Services

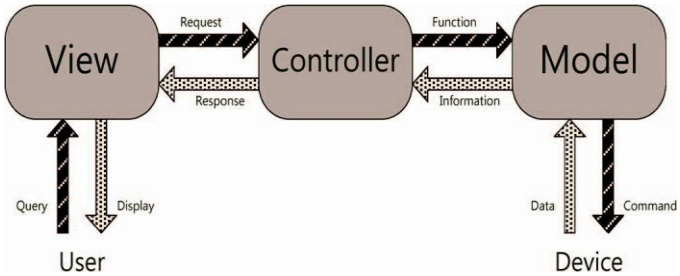


Fig.1: MVC Design Pattern

Gateway initiative (OSGi) is a well-developed SOA frame-work that is suitable for use as a broker to bridge all components in the service architecture. All service components are developed as bundles over the OSGi framework, and the embedded web container (called the Jetty Server in OSGi) can help in the publishing of all services and resources in web services. With the extendibility, flexibility and portability of OSGi, the bundles for different corresponding devices can be easily and dynamically added or removed to adapt to different application scenarios.

C. MOTIVATION

The overhead and complexity of web service technology are still high for developing an IoT system. Especially, poor HTTP transmission efficiency results in unstable service quality for displaying a large volume of heterogeneous IoT messages in a busy or low bandwidth network. Hence, maintaining stable service quality is an inevitable challenge. In this paper, we first analyze the periods of different sensor messages. Based on queue theory, we use a priority queue to check the traffic intensity and adjust the polling frequencies for different types of messages. Finally we rearrange the message request order by the proposed scheduling algorithm. Evaluation results show the implemented system effectively improves message stability.

D. OBJECTIVE

Recently, wireless communication and embedded system technologies have increasingly been applied to environmental or biomedical sensing. Many electronic appliances equipped with wireless links and sensing capabilities have gradually been introduced in the IoT environment (e.g., an RFID card reader for access control, a Wi-Fi based IP camera near the garage, a Bluetooth sphygmomanometer in a clinic). However, these off-the-shelf devices may come from different manufacturers and may support different kinds of communication protocols, thus complicating integration among various devices.

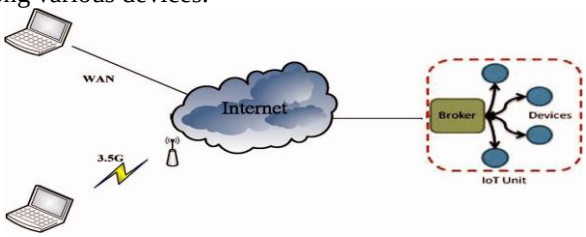


Fig.2: IoT System

II. DESIGN AND IMPLEMENTATION

A. HETEROGENEOUS MODEL OF IOT SYSTEM:

The authors introduced message scheduling in a real-time system, noting algorithm feasibility can be increased by increasing the predictability of message parameters including period, deadline and length. In a real-time message, period and deadline mean the same thing. Moreover, some messages have inflexible time constraints and exceeding these time constraints could have serious consequences for the system. On the other hand

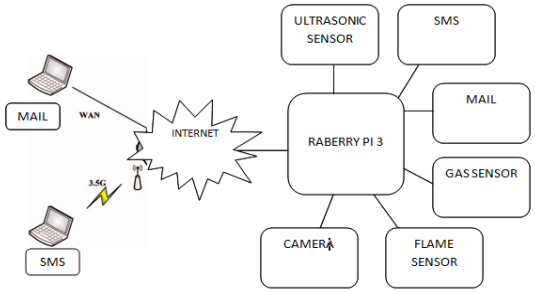


Fig.3: Proposed IoT System

in soft real-time communication, the non-observance of certain events does not necessarily have serious consequences, meaning that messages can sometimes miss their deadlines. Few studies have addressed scheduling in an application layer network such as HTTP due to the inefficiency of such networks. The present work proposes a scheduling algorithm over an HTTP network, using a web-based design to produce an improved IoT platform.

In a real-time IoT system, different messages from a variety of IoT devices are regularly sent back to the IoT client per requests from the IoT client. If the message delivery misses its deadline, the message may lose its importance. In a HTTP based request/response model, the transmission is synchronized such that every periodic request from a client should wait till the server respond to the previous response. HTTP request/response pairs fail to provide efficient communication for real-time message transmis-sion. Furthermore, when the number of requests increases in an uncontrollable network like Internet, many messages may exceed their validity deadlines. Thus, message requests should be well scheduled to secure the reliability of response messages. In this section, we used the queue theory to model the system and propose our scheduling algorithm to improve message stability from the IoT client's perspective.

B. BLOCK DIAGRAM:

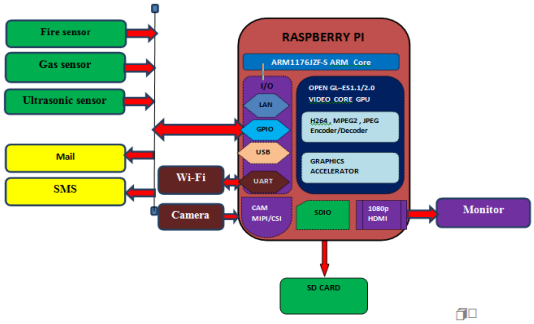


Fig.4: Block Diagram

The block diagram includes three different types of sensors namely ultrasonic, flame and gas sensors and we have two types of output through mail and sms.

III. RASPBERRY PI 3 INTRODUCTION

A Raspberry Pi is a thirty five dollar, credit card sized computer board which when plugged into an LCD and attachment of a keyboard and a mouse, it is able to complete the functions of any regular PC can. Like a PC, it has RAM, Hard Drive (SD Card), Audio and Video ports, USB port, HDMI port, and Ethernet port. With the Pi, users can create spread sheets, word-processing, browse the internet, play high definition video and much more. It was designed to be a cost friendly computer for users who needed one. Here we are using Raspberry pi 3 model B. it uses 1GB LPDDR RAM and inbuilt Wi-Fi is there when compared to earlier versions.

Fig.5: Raspberry pi 3 Model

A. ULTRASONIC DISTANCE SENSOR

Fig.6: Ultrosonic Sensor

The Ultrasonic Sensor sends out a high-frequency sound pulse and then times how long it takes for the echo of the sound to reflect back. The sensor has 2 openings on its front. One opening transmits ultrasonic waves, (like a tiny speaker), the other receives them,(like a tiny microphone).

B. GAS SENSOR

Gas sensors are available in wide specifications depending on the sensitivity levels, type of gas to be sensed, physical dimensions and numerous other factors. This Insight covers a methane gas sensor that can sense gases such as ammonia which might get produced from methane. When a gas interacts with this sensor, it is first ionized into its constituents and is then adsorbed by the sensing element. This adsorption creates a potential difference on the element which is conveyed to the processor unit through output pins in form of current.

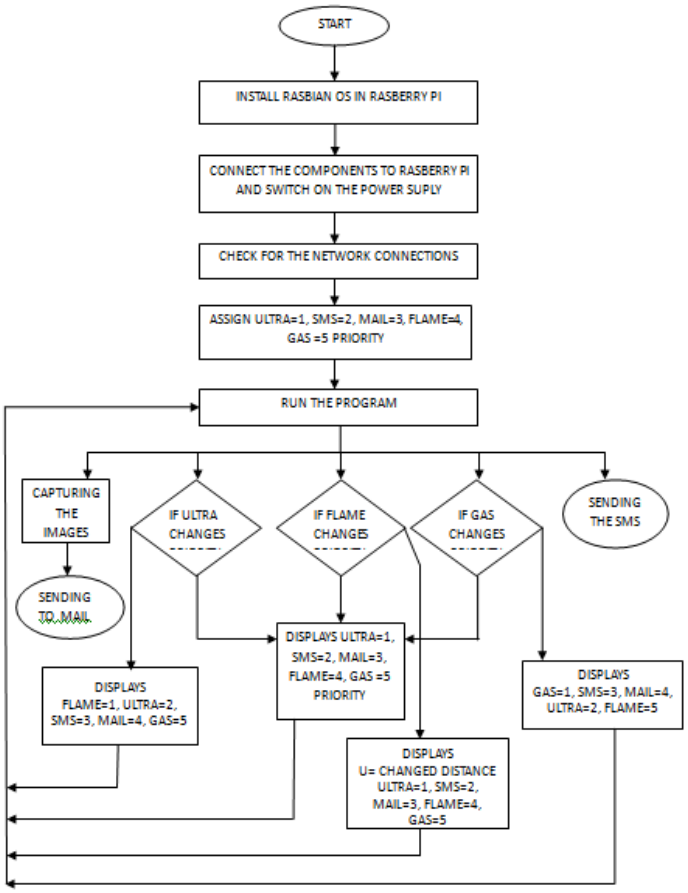
Fig.7: Gas Sensor

C. FLAME SENSOR

Fig.8: Flame Sensor

We use thermistor as flame i.e temperature sensor here. Like an RTD, a thermistor changes resistance as temperature changes. The thermistor offers higher sensitivity than RTDs, meaning that the thermistor resistance will change much more in response to temperature changes than an RTD.

D. FLOW CHART



IV. RESULTS

A. PROJECT SETUP

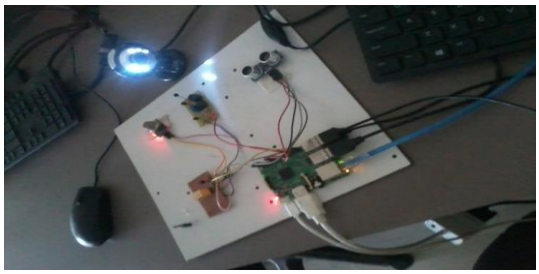


Fig.9: Project Setup

V. PROJECT RESULTS

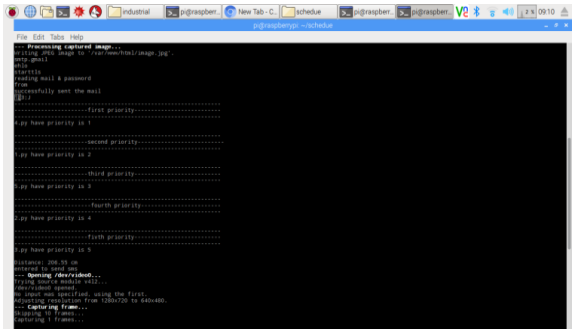


Fig.10: When flame sensor is activated

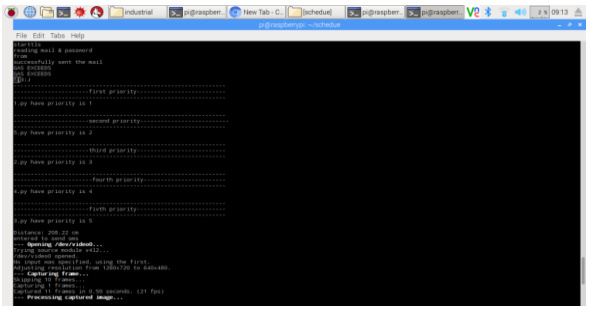


Fig.11: When Gas sensor is activated

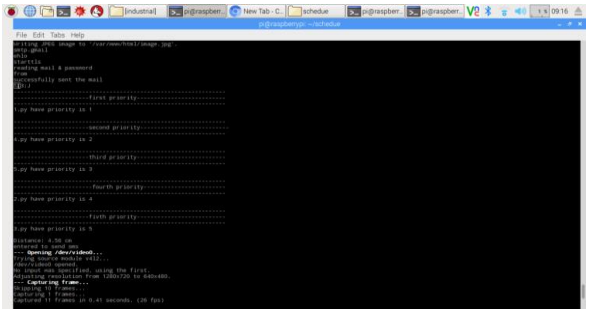


Fig.12: When Ultrasonic sensor is activated

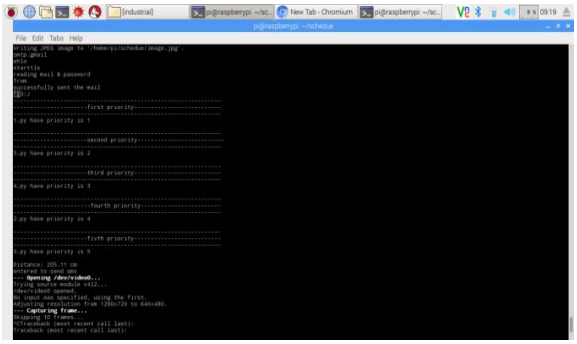


Fig.13: In general execution mode

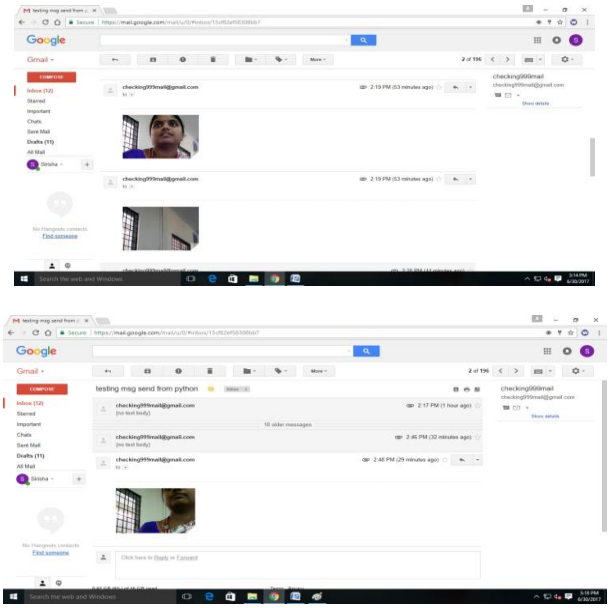


Fig.14: The Output in the mail giving the information of field

VI. CONCLUSIONS

We are making all the heterogeneous things together and scheduling them according to the priority and shortest processing time.We will have information and control over the field form the remote areas.

VII. FUTURE SCOPE

The construction of an IoT environment is based on HTTP techniques. Developers can follow the design concept to build the IoT system. The proposed integrated interface can aggregate all messages from each IoT unit including a broker and several sensor devices. . However, if the user interfaces for a real-time IoT system are built using common web technologies, HTTP-based polling will inevitably result in transmission inefficiency. Especially in an unstable network, the polling request may result in a serious delay without effective scheduling. To prevent such a situation, we first use a queuing model to analyze system traffic intensity. Using the priority queue model, each IoT message is considered as an

arrival message that in the server's system queue. Next, the proposed scheduling algorithm evaluates traffic intensity to determine the order of requests by following the SPT rule. Evaluation results validate the proposed skeleton and proposed scheduling scheme for an IoT system

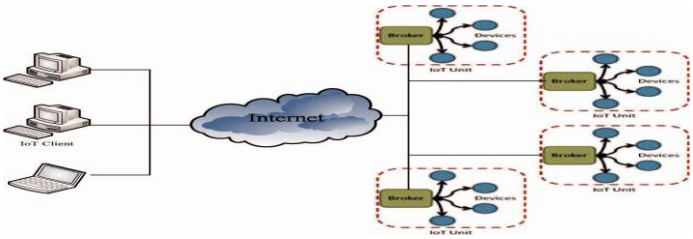


Fig.15: Future IoT System

VIII. REFERENCES

[1]. Jenq-Shiou Leu, Chi-Feng Chen, and Kun-Che Hsu,"Improving Heterogeneous SOA-Based IoT Message Stability by Shortest Processing Time Scheduling" IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 7, NO. 4, OCTOBER-DECEMBER 2014.

[2]. A. Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, and T. Razafindralambo, ''A Surveyon Facilities for Experimental Internet of Things Research,'' IEEE Commun.Mag., vol. 49,no. 11, pp. 58-67, Nov. 2011.

[3]. H. Ning and Z. Wang, ''Future Internet of Things Architecture: Like Mankind Neural System or Social Organization Framework?'' IEEE Commun. Lett., vol. 15, no. 4, pp. 461-463, Apr. 2011.

[4]. M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, ''From Today's INTRAnet of Things to a Future INTERnet of Things: AWirelessand Mobility-Related View,'' IEEE Wireless Commun.Mag., vol. 17, no. 6, pp. 44-51, Dec. 2010.

[5]. Z. Shelby, ''Embedded Web Services,'' IEEE Wireless Commun., vol. 17, no. 6, pp. 52-57, Dec. 2010.

[6]. M. Kranz, P. Holleis, and A. Schmidt, ''Embedded Interaction: Interacting with theInternet of Things,'' IEEE Internet Comput., vol. 14, no. 2, pp. 44-53, Mar./Apr. 2010.

[7]. E. Newcomer and G. Lomow, Understanding SOA with Web Services (Independent Technology Guides). Reading, MA, USA: Addison-Wesley, 2008, pp. 27-28.

[8]. D. Gross, J.F. Shortle, J.M. Thompson, and C.M. Harris, Fundamentals of Queueing Theory, 4th ed. Hoboken, NJ, USA: Wiley, 2008.

[9]. M.P. Papazoglou and W.J. Heuvel, ''Service Oriented Architectures: Approaches, Technologies and Research Issues,'' VLDB J., vol. 16, no. 3, pp. 389-415, July 2007.