IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 20, Issue 4, Ver. II (Jul - Aug 2018), PP 32-37
www.iosrjournals.org

A Network Port for the Internet of Things Using Cloud Computing

Mohammed Al-Rawe[1], Sefer Kurnaz[2]

Abstract : *- The high availability of internet connections as well as the huge number of services provided on the internet has encouraged the use of this network to connect devices of many types, other than computers, in order to communicate information or make use of these services. These devices are known as Things, andthis phenomenon has created the Internet of Things (IoT), where most of these devices are of limited resources to maintain smaller sizes that achieve the main feature of these devices, which is mobility. As computers have always been used to support devices with fewer resources, and according to the benefits of using cloud computing, a novel method is proposed in this study to reduce the resources required from the IoT devices to achieve their intended tasks. The proposed method consists, mainly, of three components, which are a web application to provide the required configurations to support each task, a data management server, and a service that communicates with the IoT devices, in order to support their operation using the configurations stored in the data management server. This service uses a plain TCP/IP protocol, instead of the more complex protocols used in the existing services, to minimize the processing and data communication of the IoT device. The results show that the proposed method has been able to reduce energy consumption to 75.41%, memory consumption to 88.32%, time consumption to 78.74% and data communications to 49.03%.*
Keywords: *-Cloud Computing, Efficiency, Internet Services,Internet of Things.*

Date of Submission: 23-07-2018 Date of acceptance: 07-08-2018

I. INTRODUCTION

The peripherals connected to a computer, which enable it from interacting with the environment and external devices, may be divided into two categories. The first category is the sensors, which have the ability to measure one or more variables and convert these values into computer standards. The second category receives commands from the computer and executes these commands on other devices. The use of these peripherals led to the revolution of using computers in automation [1], where computers became capable of interacting with other devices in order to acquire the necessary input data, process them to make a decision, and then execute that decision using the output peripherals. The benefits of using such systems enabled the evolution of these systems from a single-duty device, which is usually capable of interacting with only one kind of inputs or outputs, to the embedded systems that are capable of interacting with multiple inputs and outputs mountainously.

These devices are usually of limited capabilities. Thus, they are usually connected to a computer in order to make use of the relatively huge capabilities of the computer. To make these devices more useful in different applications, they are designed to be able to use the existing computer ports in order to exchange information with the computer. Earlier devices used the computer's parallel port to send and receive data [2]. Later, the serial port is also used to exchange data between these devices and computers [3]. The use of serial port gives more flexibility to the distance between a device and the computer. Then, with the help of microcontrollers or some specialized converters, these devices are connected to computers using Universal Serial Port (USB). This enableshigh-speed data transfer between a device and the computer [4].

The rapid growth of Wireless Sensor Networks (WSN) usage in different fields of application and with the need to communicate to these sensors from different places leads to the urgent need of connecting these sensors to the internet, which created the internet of things (IoT), [5]. These applications may vary from saving lives by monitoring patients' vitals [6] to monitoring the environment for the plants in agricultural applications [7]. These devices are connected to the internet using a different interface such as wireless internet (WIFI module) or using mobile networks (GSM module). Connecting these devices to the internet enables communications among them, to a centralized device or to monitoring and control devices.

II. LITERATURE REVIEW

The high availability of the internet connections encourages the attempt to monitor and control as many devices as possible. These devices may vary from as simple as measuring an environment variable, such as temperature, to as complex as remote controlling machines, [8-10]. Earlier, peripherals are connected to computers in order to collect data from the external world, send this data to a computer, then dedicated software is used in the computer to process the collected data and produce a suitable command. This command is then sent to the corresponding peripheral in order to execute it in a suitable way, by converting the requested command from computer language to the suitable output according to the environment that this peripheral is connected to, [11]. Moreover, the rapid growth in computer usage also affects the technologies related to it. These peripherals are also affected by this growth, where the connection scheme to the computer has evolved to enable faster data transfer and more peripherals per each computer. Furthermore, these devices have earned the capability to collect data, make decisions, and execute these decisions, which enables the use of standalone systems that can run without the need for a computer. These standalone systems have started to grow rapidly, as well, which imposes the need to interchange data between these systems. Thus, wireless communications are introduced to these systems, which created the Wireless Sensor Networks (WSN).

As these devices are, usually, of small volume and are used in different environments, different power sources are used to energize these devices. Some of these power sources have a very limited amount of energy, or the energy consumed from these sources may be expensive. Thus, many methods are proposed to optimize the power consumption of these devices. Where consumed power is computed per distance unit per every bit of digital data transmitted. Thus, to reduce the consumption of the provided energy, most of the existing methods emphasis on finding the shortest path between the transmitting and receiving points, [12-14]. These devices are then used with special models to connect to the internet, which created the Internet of Things. These compact standalone devices enable connecting almost everything in the world to the internet, so that, it is possible to monitor and interact with these devices remotely. These connections are also used to interchange data between these devices, as well as connecting these devices to different online services.

Moreover, the existing online services are not targeted to be used by these compact devices. These services are mainly targeted toward computers, which have relatively higher performance than the compact device of the internet of things. Even the new services that are designed to serve the internet of things devices are using the same connection schemes and protocols that are used by other services. For example, the well-known IoT services, provided at thingspeak.com [15], which allows interchanging data among IoT devices and logging data for further analysis, uses the hypertext transfer protocol. Although this protocol is very easy to achieve using a computer, it is a memory and power consuming method to be used with the IoT devices. The consumed memory is used to store some extra information that is used to encapsulate the original data being transferred from and to the IoT device. As well as the complexity of requesting a value from that service using the HTTP protocol. Thus, as these devices are connected to the internet, and are required to use the services available online to achieve their goals, and with the fact that these devices have always used the assistance of computers to achieve better performance, using different physical ports to connect to computers. A new method is required to assist these devices to reduce the effort to reach the internet services using a network port to connect to, instead of the classical physical ports. The service provided on this network port should be custom made to meet the needs of the IoT devices.

The high availability of cloud servers, as well as the high flexibility regarding the power provided for each cloud server, it is preferred to use such servers to provide this service. In addition to the limitation of using web hosting plans, which reduces the available options of protocols that can be used to communicate with the IoT device to a limited number of the existing protocols. This limitation is the main reason behind using the HTTP protocol in most of the existing services.

III. PROPOSED METHOD

In order to achieve the same tasks required for the IoT devices using less amount of data, users are required to create predefined applications that can be accessed by the IoT devices to complete any required data in order to execute that application. To achieve that, the hierarchy shown in Fig. 1 is used to implement the proposed model. Using such a model, the user uses the website to enter all the required data to create a predefined application. The basic configurations that should be provided for every new application defined in the system are the following

- Application ID. A random unique ID number generated automatically by the system.
- Application Password. Set by the user to control access to that application.

- Application Type. One of multiple provided commands.
- Execution Trigger. Whether to execute the application as soon as data received, or by comparing a certain part of the data to a predefined condition.
- Mandatory configurations for each type of applications.
- Predefined data and data expected from the IoT device.

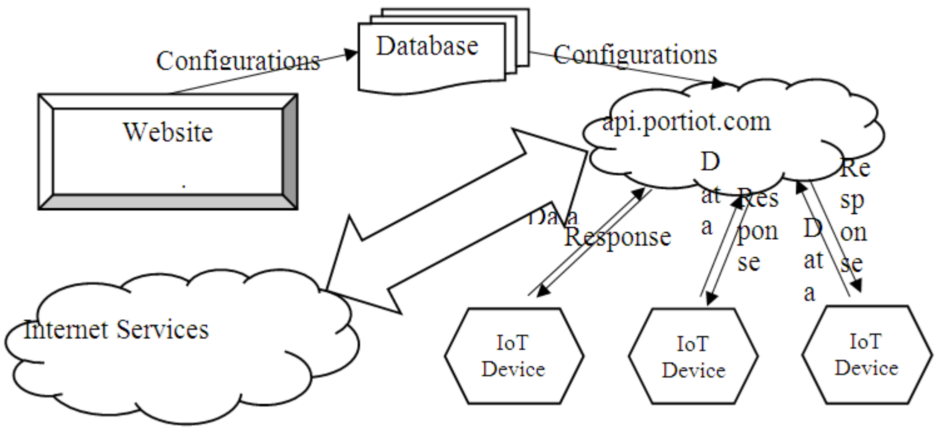

Figure III1: An illustration of the hierarchy of the proposed system.

Some of the commands that are integrated into the system are:

- Write. To store a value of a predefined type, such as integer, double or string.
- Read. To read the value stored in for a specific point.
- Smail. To send an email using a predefined mail server and login details.

The 'Read' and 'Write' commands have different possible ways to read or write the required point. These points may be:

- Local. Where points are stored locally in the database, in order to exchange data among multiple IoT devices.
- HTTP. Where points are stored on remote servers and require specific HTTP methods, such as request or post, to read or write these methods.
- XML. Where the required points are included in an XML response. So that the required value is retrieved in the system and sent to the IoT device, instead of sending the entire response, which causes more traffic and requires more processing at the IoT device.

All these configurations in addition to some extra configurations, related to every type of commands, must be set by the user, for each application before they are used by the IoT devices. For example, to set up an application with a Smail command, the user is required to provide the mail server, login credential, the address of the receiver of the email, the subject of the email, and the body of the email. It is possible to predefine any of these configurations, except the mail server and login credentials, or leave the entire, or a part of, a specific field to be filled using data incoming from the IoT device.

Upon storing the above configuration in the database, it becomes possible for the service running at the 'api.portiot.com' to use these configurations in order to execute the required commands using the data incoming from the IoT devices. The algorithm used to process the incoming data, execute the command, and send responses back to the IoT device is shown in algorithm 1.

```
Algorithm1: Execute IoT Applications.
1    AppID = Application name received from the IoT device.
2    AppPW = Authentication password received from the IoT device.
3    If database has (application with name=AppID and password=AppPW)
4            AppData = Retrieve data relevant to that application from the database.
5            IoTData = Receive data from the IoT device
6    Else
7            Send message 'Authentication Failed!' to the IoT device
8            Close connection
9    Concatenate AppData and IoTData as defined in the configurations of the application
10   If (application has response)

11           Response = execute application
```

```
        If (application require response process)
            IoTRes = process Response according to application configuration
        Else
            IoTRes = Response
        Send IoTRes to the connected IoT device
    Else
        If (execution is successful)
            Send '1' to the IoT device
        Else
            Send '0' to the IoT device
    Close the connection with the IoT device
```

IV. Experimental Results

In order to evaluate the performance of the proposed method, and compare it to existing techniques, to achieve some of the popular tasks required from such devices, the time, energy, memory and amount of data transferred from and to the IoT device are measured during the execution of these tasks. For this purpose, the circuitry shown in Fig. 2 is implemented, where the power information collected by the INA219AIDR power meter are transmitted to a computer using an Arduino UNO device through a Universal Serial Bus (USB) port. The other USB port is used to program the well-known ESP8266-12E IoT device, in order to implement a different task per each experiment. To produce accurate measurements, the voltage level on one of the pins on the ESP8266-12E device is set to high during the setup of the IoT device, where this voltage level is changed to low just before the execution of the required task is started. When the execution of the task is finished, the voltage level on that pin is set back to high, so that, only data logged while the level is low are considered. Moreover, the average of repeating the same task ten times, in a loop, is used instead of a single measurement, to avoid any biased results.

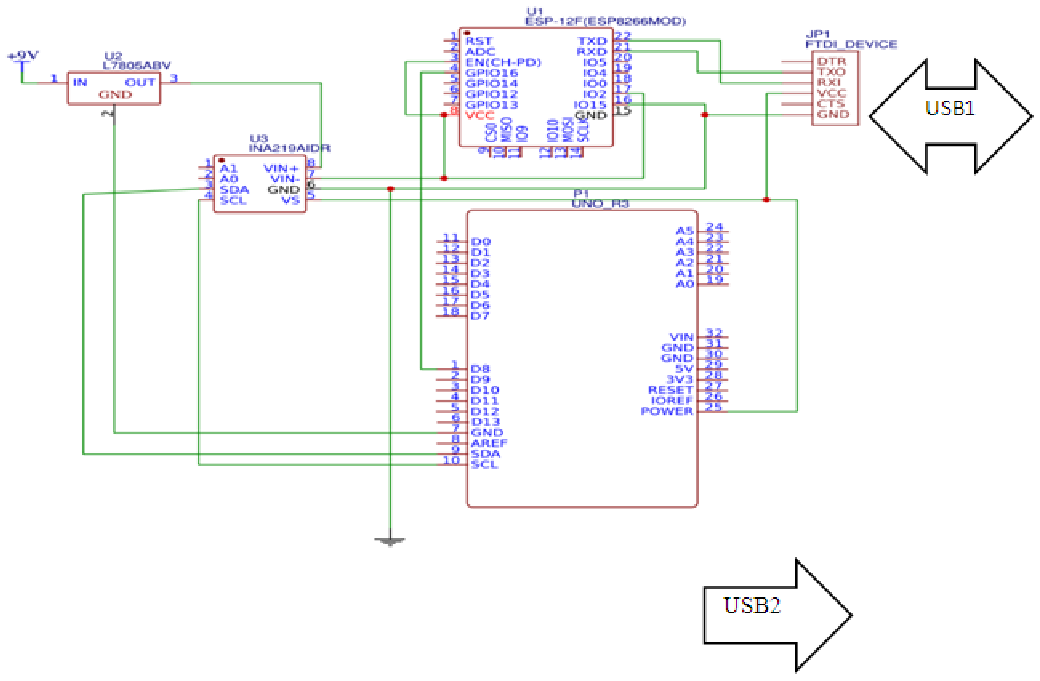

Figure 2: Experimental setup.

The tasks used for the evaluation are data read and write, sending emails and extracting information from XML response retrieved from a remote server. Reading and writing data is compared to the widely currently used method, through the ThnigSpeak services, while sending emails is compared between using Google's Gmail service directly, or through the proposed method. Moreover, information is extracted from Yahoo weather XML responses using directly, and through the proposed method, as well. The power consumption results are summarized in Table 1, where these results show that the IoT device has been able to achieve the same task consuming only 75.41% of that required to achieve the same tasks without the aid of the proposed method. The results also show that the highest difference in power consumption is achieved during extracting information from XML responses, which illustrate the reduction of the processing power required from the IoT device to process the entire response and extract the required information.

Table 1: Summary of the IoT device's energy consumption.

	Energy Consumption (mWH)	
	Proposed Method	**Existing Methods**
Writing Data	16.86	26.36
Reading Data	16.37	24.88
Sending Emails	16.64	24.07
Retrieving XML	16.55	28.39
Average	**16.605**	**25.925**

The memory required from the IoT device to store the instructions required to achieve the intended task, with and without the assistance of the proposed method, are shown in Table 2, where the memory required from the IoT device while executing the required tasks using the proposed method is around 88.32% of that required to achieve the same tasks without the assistance of the method. The results also show that the amount of memory required from the IoT device to achieve most of the tasks is constant, where only the mandatory parameters are sent to the cloud services, which is responsible forexecuting the required task and sending the response back to the IoT device. The only task that requires different memory size is writing data, where the difference in the required memory is caused by the different data sizes used in the experiments, which are stored in the memory of the IoT device for transmission.

Table 2: Summary of the IoT device's memory consumption.

	Memory Consumption (Bytes)	
	Proposed Method	**Existing Methods**
Writing Data	258192.3	261025.7
Reading Data	258175	261471
Sending Emails	258175	322566
Retrieving XML	258175	324194
Average	**258179.3**	**292314.2**

The times required by the IoT device to achieve the intended tasks are summarized in Table 3, where the use of the proposed method has been able to reduce the time consumption to 78.74% of the total time consumed by the IoT device to achieve these tasks solely. However, the more important observation in these results is the similar time required to achieve different tasks with the support of the cloud server, regardless of the complexity of the task, while more complex tasks require longer time without the support of the proposed method. Such behavior agrees with the hypothesis of this study that these task tasks are very easy to be handled by the cloud server, according to the huge difference of the resources available on the cloud server, compared to those available at the IoT device.

Table 3: Summary of the IoT device's time consumption.

	Time Consumption (mSec)	
	Proposed Method	**Existing Methods**
Writing Data	176.25	232.69
Reading Data	175.08	225.6
Sending Emails	175.07	202.88
Retrieving XML	174.43	228.89
Average	**175.2075**	**222.515**

As some of the IoT devices are connected to the internet through channels with limited capacity, as well as the power required to transmit and receive the data in some cases, it is important to measure the amount of data transferred to and from the IoT device to achieve the required task. Thus, Table 4 illustrates the number of bytes sent from and received by the IoT device to achieve the tasks included in the evaluation. The results show that the use of the proposed method has been able to reduce the amount of data communications to 49.03% of the amount of data communicated when the IoT device executed the tasks without the assistance of the proposed method. This huge reduction is caused by the elimination of any unnecessary data communication, where actions that require no responses are acknowledged using only one byte of data, unlike other techniques where larger responses are sent to the IoT device to describe the state of the execution. Moreover, storing any data required to achieve the intended task that are always the same and not generated at the IoT device, such as most of the email body, has reduced the amount of data transmitted by the IoT device, where only variables measured by that device are sent to the server in order to append it with the stored data, according to the configurations provided by the developer.

Table 4: Summary of the sent and received data.

	Transferred Data (Bytes)					
	Proposed Method			**Existing Methods**		
	Sent	**Received**	**Total**	**Sent**	**Received**	**Total**
Writing Data	41.33	1	42.33	186.33	635	821.33
Reading Data	33	5.33	38.33	182	639.33	825.67
Sending Emails	38	1	39	105.33	27	132.33
Retrieving XML	33	3706.33	3739.33	247	5844	6091
Average	**36.3325**	**928.415**	**964.7475**	**180.165**	**1786.333**	**1967.583**

V. CONCLUSION

The rapid growth of the use of IoT device requires the implementation of new techniques to accommodate such growth, where most of the existing techniques and services available on the internet are targeted to computers that have, relatively, higher resources that IoT devices. Thus, in this study, a novel technique to support the operation of IoT devices accessing internet services as well as communicating information among them, is proposed. The proposed technique consists of three main components, a web application that IoT developers use to provide configurations for the APIs that the IoT devices use, a data management server to store these configurations, as well as users' information, and a service that communicates with the IoT device in order to execute the stored tasks, using the configurations stored in the data management server. Unlike existing techniques, the service of the proposed method uses a plain TCP/IP protocol to communicate with the IoT device, in order to eliminate any unnecessary information added to the payload data, similar to most of the existing network protocols, such as HTTP. The results show a huge reduction in the resources required from the IoT devices to achieve the exact same tasks, when the proposed method is used to assist IoT devices achieving the required tasks. The average amount of required energy, from the IoT device, is reduced to 75.41% using the proposed method, which is one of the most important factors to improve the performance of the IoT devices, as most of these devices rely on limited energy sources. The amount of memory required to store the scripts that run on them to achieve their tasks has also been reduced to 88.32% of that required to achieve these tasks without the support of the proposed method. The average time required to achieve these tasks has also dropped to 78.74% of the total time required by the IoT device to achieve these tasks, while the amount of data communicated by the IoT device has dropped to 49.03% of the original amount of data communicated with the remote services, directly. Thus, the proposed has been able to improve the performance of the IoT device, which improve the efficiency of these devices, so that, the cost can be reduced by using devices with less resource to achieve the same task, or by using the same IoT device to achieve more tasks, based on the support that the proposed method provide for these devices.

REFERENCES

[1]. D. Popovic and V. P. Bhatkar, *Distributed Computer Control Systems in Industrial Automation* vol. 66: CRC Press, 1990.

[2]. S. Rosminah and A. Z. M. Ali, "Development of hardware-interfacing learning kit for novice learning programming," *International Journal of Information and Education Technology,* vol. 6, p. 647, 2016.

[3]. V. K. Patel and M. N. Patel, "Development of Smart Sensing Unit for Vibration Measurement by Embedding Accelerometer with the Arduino Microcontroller," *International Journal of Instrumentation Science,* vol. 6, pp. 1-7, 2017.

[4]. E. Popa and V. Popa, "Graphic interface for numerical commands on the USB port of PC compatible computers," in *MATEC Web of Conferences*, 2017, p. 01008.

[5]. J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future generation computer systems,* vol. 29, pp. 1645-1660, 2013.

[6]. S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, *et al.*, "Health monitoring of civil infrastructures using wireless sensor networks," in *Proceedings of the 6th international conference on Information processing in sensor networks*, 2007, pp. 254-263.

[7]. F. TongKe, "Smart agriculture based on cloud computing and IOT," *Journal of Convergence Information Technology,* vol. 8, 2013.

[8]. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials,* vol. 17, pp. 2347-2376, 2015.

[9]. J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal,* vol. 1, pp. 3-9, 2014.

[10]. R. Want, B. N. Schilit, and S. Jenson, "Enabling the internet of things," *Computer,* vol. 48, pp. 28-35, 2015.

[11]. G. Dixey, *Computer interfacing*: Newnes, 2014.

[12]. G. Han, Y. Dong, H. Guo, L. Shu, and D. Wu, "Cross-layer optimized routing in wireless sensor networks with duty cycle and energy harvesting," *Wireless communications and mobile computing,* vol. 15, pp. 1957-1981, 2015.

[13]. Y. Wu and M. Cardei, "Distributed algorithms for a robust topology using reconfigurable radios wireless sensor networks," *International Journal of Sensor Networks,* vol. 24, pp. 264-276, 2017.

[14]. M. Hammoudeh and R. Newman, "Adaptive routing in wireless sensor networks: QoS optimisation for enhanced application performance," *Information Fusion,* vol. 22, pp. 3-15, 2015.

[15]. M. A. G. Maureira, D. Oldenhof, and L. Teernstra, "ThingSpeak–an API and Web Service for the Internet of Things," *World WideWeb,* 2015.

Mohammed Al-Rawe "A Network Port for the Internet of Things Using Cloud Computing "IOSR Journal of Computer Engineering (IOSR-JCE) 20.4 (2018): 32-37.