AN OVERVIEW ON IOT ENABLING TECHNOLOGIES

Nupur Tyagi

Computer Science, Delhi University
New Delhi, India

ABSTRACT:

This paper provides an overview on Internet of Things (IoT) enabling technologies (various protocol, standards etc.). The recent adoption of a variety of enabling Wireless communication technologies such as RFID tags, BLE, ZigBee, etc. and embedded sensor and actuator nodes, and various protocols such as CoAP, MQTT, DNS etc. have made IoT to step out of its infancy. Now smart sensors can collaborate directly with machine without human involvement to automate decision making or to control a task. This paper starts by providing a horizontal overview of the IoT. Then, we give an overview of some technical details that pertain to the IoT enabling technologies, embedded systems, protocols, and applications.

Keywords: IoT, CoAP, REST, HTTP, RPL, 6LoWPAN, ZigBee, WSN

[1] INTRODUCTION

The Internet of Things is about connecting devices, vehicles, buildings etc. creating opportunities for direct integration between the physical world and machine enabling objects to collect and communicate data and thus resulting in intelligent decision making. This new wave in the era of computing is going outside the realm of the traditional desktop. Many of the objects, having invisible(embedded) information and communication system, surrounding us will be on the network in one form or another, gathering and forwarding data on our day to day activities. This results in the production of huge amounts of data which have to be stored, processed, analyzed and presented in an efficient, and easily interpretable form. In this model services are provided and delivered like traditional commodities. Cloud computing can provide the virtual infrastructure for such utility computing which integrates monitoring devices, storage devices, analytics tools, visualization platforms and client delivery. The cost based model that Cloud computing offers will enable end-to-end service provisioning for businesses and users to access applications on demand from anywhere. IoT can play a significant role in various domains and can improve our quality of living. These applications include healthcare, emergency responses, transportation. Employing embedded technologies and advanced set of protocols made a breakthrough in IT sector by making this concept of IoT successful. Many IoT standards have been proposed to simplify the task of application developers’ and service providers’. Numerous protocols are formulated in support of the IoT by several groups including efforts led by the World Wide Web Consortium.
AN OVERVIEW ON IOT ENABLING TECHNOLOGIES

(W3C), IEEE, Internet Engineering Task Force (IETF) and the European Telecommunications Standards Institute (ETSI).

In this paper, we will look into various protocols, standards and technologies involved in IoT. However, it is not necessary that all of these protocols mentioned in the above figure have to be bundled together to build a given IoT application. According to the nature of an IoT application, some standards may not be required to be supported. In the following sections, we will give an overview of one of the commonly used protocols i.e. CoAP and its functionality.

**Remarks:** HTTP, is a renowned protocol for many devices. A simple 8-bit controller device can also create simple GET and POST requests for HTTP. However, it has some drawbacks. Mainly with HTTP like protocol there are two drawbacks firstly, the power requirements of the devices and secondly the memory size of the program can be an issue on small devices. A binary protocol is a solution for the aforesaid issues. Various protocols which make implementation of IoT, a real time application are discussed in the following sections.
[2] Building blocks of IoT

Here are three IoT components which enables seamless ubiquitous computing: (a) Hardware—made up of sensors, actuators and embedded systems(b)Middleware—on demand storage and computing tools for data analytics and (c)Presentation/user interface—easy to understand visualization and interpretation tools which can be widely accessed on different platforms and which can be designed for different applications. These three classifications are then further divided into the following categories.

1. **Sensors/Actuators**: They are the devices as simple as a solar cell or a temperature measuring device (thermometer) which generally senses the change or occurrence of an event, collects the data and forward it to the corresponding devices/applications over the Internet.

2. **Computational unit**: Processing units (e.g., microcontrollers, SOCs, FPGAs, microprocessors) and software applications signify the “brain” and the computational ability of the IoT. Various hardware platforms have been developed to run IoT applications such as Arduino, ARM, Intel Galileo, Raspberry PI, BeagleBone, T-Mote Sky etc. Furthermore, many software platforms are utilized to provide IoT functionalities. Among these platforms, Operating Systems are the most significant since they run for the whole activation time of a device, however not all systems have operating system installed in them like Arduino may not require an operating system to do its assigned tasks. Below you can see pictures of Arduino Uno board and its IDE.
Some of the operating systems are listed in the table.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Language Support</th>
<th>Minimum Memory (KB)</th>
<th>Event-based Programming</th>
<th>Multi-threading</th>
<th>Dynamic Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>TinyOS</td>
<td>uesC</td>
<td>1</td>
<td>Yes</td>
<td>Partial</td>
<td>Yes</td>
</tr>
<tr>
<td>Contiki</td>
<td>C</td>
<td>2</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LiteOS</td>
<td>C</td>
<td>4</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Riot OS</td>
<td>C/C++</td>
<td>1.5</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Android</td>
<td>Java</td>
<td>-</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Contiki OS has been used widely in IoT applications as it has a simulator called Cooja which allows easy simulation and emulation of IoT and wireless sensor network (WSN) applications. TinyOS, LiteOS and Riot OS are lightweight OS well suited for IoT environments.

3. Addressing schemes: Giving an id to the communicating device is important. Identification methods such as electronic product codes (EPC) and ubiquitous codes (uCode) are available. In addition, addressing methods of IoT objects include IPv6 and IPv4. 6LoWPAN provides a compression mechanism over IPv6 headers that makes IPv6 addressing appropriate for low power wireless networks. Thus 6LoWPAN is used for addressing. Identification methods are used to provide a clear identity for each object within the network. A device can be distinctly identified with the help of its Id and address.

4. Communication technology: Typically, the IoT nodes should operate using low power in the presence of lossy and noisy communication links. To accomplish the above said requirement numerous communication protocols are introduced and are being used such as BLE, WiFi, Bluetooth, IEEE 802.15.4, Z-wave, and LTE-Advanced. Some specific communication technologies are also in use like RFID, Near Field Communication (NFC) and ultra-wide bandwidth (UWB). RFID was the initial technology used to realize the M2M idea (RFID tag and reader). The RFID tag represents a simple chip or label attached to provide object’s
identity. The RFID reader transmits a query signal to the tag and receives reflected signal from the tag, which in turn is passed to the database. The database connects to a processing center to identify objects based on the reflected signals within a (10 cm to 200 m) range [2] [27]. RFID tags can be active, passive or semi-passive/active. Active tags are powered by battery while passive ones do not need battery. Semi-passive/active tags use board power when needed [2].

5. Services: IoT services can be categorized under four classes: Identity-related Services, Information Aggregation Services, Collaborative-Aware Services and Ubiquitous Services [2]. Identity-related services are the elementary and important services as they are used in further other types of services as well. Every object has to be uniquely identified by the applications to do the needful processing. Information Aggregation Services collect and summarize raw sensory measurements that need to be processed and reported to the IoT application. Collaborative-Aware Services act on top of Information Aggregation Services and use the obtained data to make decision and react accordingly. Ubiquitous Services, however, aim to provide Collaborative-Aware Services anytime they are needed to anyone who needs them anywhere. With this categorization, we review some applications of the IoT in the following paragraphs. The ultimate goal of all IoT applications is to reach the level of ubiquitous services. However, this end is not achievable easily since there are a lot of difficulties and challenges that have to be addressed. Most of the existing applications provide identity related, information aggregation, and collaborative-aware services. Smart healthcare and smart grids fall into the information aggregation category and smart home, smart buildings, intelligent transportation systems (ITS), and industrial automation are closer to the collaborative-aware category [2]. There is a need to create data and service sharing infrastructure which can be used for addressing several application scenarios. A conceptual framework integrating the ubiquitous sensing devices and the applications is shown in Figure 4. In order to realize the full potential of cloud computing as well as ubiquitous Sensing, a combined framework with a cloud at the center seems to be most viable. This not only gives the flexibility of dividing associated costs in the most logical manner but is also highly scalable. [3]

6. Semantics to extract knowledge and finding out pattern. Semantic in the IoT refers to the ability to extract knowledge smartly by different machines to provide the required services. Knowledge extraction includes discovering and using resources and modeling information. Also, it includes recognizing and analyzing data to make sense of the right decision to provide the exact service [62]. Thus, semantic represents the brain of the IoT by sending demands to the right resource. This requirement is supported by Semantic Web technologies such as the Resource Description Framework (RDF) and the Web Ontology Language (OWL). In 2011, the World Wide Web consortium (W3C) adopted the Efficient XML Interchange (EXI) format as a recommendation [63]. EXI is important in the context of the IoT because it is designed to optimize XML applications for resource-constrained environments. Furthermore, it reduces bandwidth needs without affecting related resources such as battery life, code size, energy consumed for processing, and memory size. EXI converts XML messages to binary to reduce the needed bandwidth and minimize the required storage size. [2]
Technological advancements in low power integrated circuits and wireless communications have made it possible to use low cost, low power miniature devices in getting connected to the Internet. Embedded systems, wireless sensor networks, new communication technologies have improved the viability of utilizing a sensor network consisting of a large number of intelligent sensors, enabling the collection, processing, analysis and dissemination of valuable information, gathered in a variety of environments [7]. Sensor data are shared among sensor nodes and then sent to a distributed or centralized system for data analytics. WSN hardware is typically a node which contains sensor interfaces, processing units, transceiver units and power supply. Almost always, they comprise of multiple A/D converters for sensor interfacing and more modern sensor nodes have the ability to communicate using one frequency band making them more versatile [7].

Walter Colitti et al. work Integrating Wireless Sensor Networks with the Web is cited here: 6LoWPAN enables the use of IPv6 in Low-power and Lossy Networks (LLNs), such as those based on the IEEE 802.15.4 standard [10]. In addition to 6LowPAN, IETF Routing over Low-power and Lossy networks (ROLL) Working Group has designed and specified a new IP routing protocol for smart object internetworking. The protocol is called IPv6 Routing Protocol for Low-power and Lossy networks (RPL). One of the major benefits of IP based networking in LLNs is to enable the use of standard web service architectures without using application gateways. As a consequence, smart objects will not only be integrated with the internet but also with the Web. This integration is defined as the Web of Things (WoT). The advantage of the WoT is that smart object applications can be built on top of Representational State Transfer (REST) architectures. REST architectures allow applications to rely on loosely coupled services which are shareable and reusable. In a REST architecture a resource is an abstraction controlled by the server and identified by a Universal Resource Identifier (URI). The resources are decoupled by the services and therefore resources can be arbitrarily represented by means of various formats, such as XML or JSON. The resources are accessed and manipulated by an application protocol based on client/server request/responses. REST is not tied to a particular application protocol. However, the vast majority of REST architectures nowadays use Hypertext Transfer Protocol (HTTP). HTTP manipulates resources by means of its methods GET, POST, PUT, etc. [6]. REST architectures allow IoT and machine-to-Machine(M2M) applications to be developed on top of web services which can be shared and reused. The sensors become abstract resources identified by URIs,
represented with arbitrary formats and manipulated with the same methods as HTTP. As a consequence, RESTful WSNs drastically reduce the application development complexity. The use of web service in LLNs is not straightforward as a consequence of the differences between Internet applications and IoT or M2M applications. IoT or M2M applications are short-lived and web services reside in battery operated devices which most of the time sleep and wakeup only when there is data traffic to be exchanged. In addition, such applications require a multicast and asynchronous communication compared to the unicast and synchronous approach of standard Internet applications [11]. The Internet Engineering Task Force (IETF) Constrained RESTful environments (CoRE) Working Group has done major standardization work by introducing the web service paradigm into networks of smart objects. The CoRE group has defined a REST based web transfer protocol called Constrained Application Protocol (CoAP). CoAP includes the HTTP functionalities which have been re-designed taking into account the low processing power and energy consumption constraints of small embedded devices such as sensors. In order to make the protocol suitable to IoT and M2M applications, various new functionalities have been added [5][6]. CoAP is a web transfer protocol based on REST architecture, optimized for resource constrained networks typical of IoT and M2M applications [5]. In a REST architecture resources are server-controlled abstractions made available by an application process and identified by Universal Resource Identifiers (URIs) [5].

Fig 7: HTTP and CoAP protocol stacks
CoAP uses the same methods like HTTP such as GET, PUT, POST and DELETE for accessing the resources but it is not a blind compression of HTTP. A subset of HTTP functionalities has been re-designed and certain mechanisms have been added taking into account the low processing power and energy consumption constraints of devices of a WSN. The use of an IP based communication and a REST based Web architecture in LLNs facilitates the integration of WSNs with Internet based Web applications [5].

An end to end IP based architecture to integrate WSN and web
An application gateway which acts as an interface between the device and the web service needs to have full knowledge of the functionalities of each connected device thus reducing the architecture flexibility and system scalability. This is one of the major problems of non IP based WSN communication standards such as ZigBee. ZigBee does not have a standard IP networking layer which implies that a standard web service architecture cannot be implemented on top of ZigBee. Besides hampering the interoperability, the lack of a web service architecture requires the use of application gateways when interconnecting ZigBee WSNs to the Internet [5]. If the sensor (or device) can itself expose its resources to the cloud(web) with the help of an application protocol such as CoAP the design of gateway can be simplified significantly as compared to the case where application gateway is required to expose the device’s resources. In this work, Walter Colitti et al. used the gateway integrating the CoAP based WSN with the HTTP based Web...
application consists of a Linux Ubuntu machine with a Contiki-gateway attached via USB port. Gateway’s building blocks are the web server, the database and the CoAP client. For simplicity, the first gateway prototype includes all the building blocks in the same box. In a second phase the application will be deployed on an application server. Therefore, the application logic and the data collection functionality will reside into two different machines. This clearly reduces the complexity of the gateway and improves the system scalability. The web server includes a set of services which are used to retrieve data either from the database or directly from the CoAP client. When the Web server sends the Web client historical data already available in the database, the web server directly accesses the database without communicating with the CoAP client and sends the data back to the Web client. When the Web Server needs to send fresh data coming from the WSN (upon client request or upon changes of the WSN resources), the web server bypasses the database and directly communicates with the CoAP client. Since the Web application has been developed with Google Web Toolkit (GWT), the web server at the moment is the built-in GWT’s server called Jetty. When the application will be deployed on an application server a different web server will be chosen. The database stores data coming from the CoAP client and makes them available to the web server. The database chosen is Apache CouchDB. Apache CouchDB is a document-oriented database which can be queried and indexed with MapReduce technique using JavaScript. It stores JSON documents and provides a RESTful API. Since the CoAP client receives WSN data already in JSON documents, the storage operation is rather simple and does not require any intermediate data manipulation. However, the system has not yet been tested under high frequency measurement conditions and therefore the database scalability has not yet been evaluated. If a high number of stored documents results in slow database access, an extra data processing layer might be needed in order to reduce the data access latency. The libcoap CoAP client module is responsible for communicating with the WSN. In the current prototype the gateway-WSN data exchanges are always initiated by the CoAP client. This is a consequence of the fact that Contiki does not yet support observations. We are currently adding this functionality so that the WSN can spontaneously send the CoAP client data upon resource status change. Once retrieved the JSON data from the WSN, the CoAP client add a time stamp and stores them into the database. The time stamp is needed when providing the web server with historical data. For simplicity, the current gateway implementation does not include proxy functionality between HTTP and CoAP. Therefore, there is not translation between the HTTP request and the CoAP request and vice versa. Upon receiving the HTTP request, the web server invokes the CoAP client with the parameters included in the HTTP requests (IPv6 address and port of the mote and the resource of interest). This implies that the gateway is not completely transparent to the application and to the WSN. A proxy module able to do the HTTP-CoAP translation and vice versa needs to be implemented in order to increase the transparency of the gateway. This will also facilitate the gateway in handling more complicated operations such as observations. In this case for example, a mechanism that translates an HTTP subscription (e.g. long-polling) needs to be translated in a CoAP observation relationship [5].

[4] USE CASE
In the use case discussed below we are considering the activity tracker (smart watch) as the sensor/device collecting the data and BLE as the communication technology.
Since IPv6 support requires the use of the 6LoWPAN protocol, so according to an IETF draft the 6LoWPAN protocol is added directly on top of the Bluetooth low energy L2CAP protocol layer. With this approach, IPv6 can be used end-to-end from the local Bluetooth low energy device to an Internet service. In this use case, there are devices connected to the gateway that implement their applications as Internet applications. Typically, CoAP is used as the method to send data to the gateway in order to minimize overhead. The gateway is completely unaware of the application and conversion takes place between CoAP and HTTP. There might also be use cases where HTTP is used all the way down to the device although this will be more resource consuming than CoAP. This use case is quite flexible and can be used independently of the connected device type. In particular, this use case is useful for devices that want to comply with an IP-based standard such as when devices using the Smart Energy 2.0 standard want to use Bluetooth low energy instead of 802.15.4 which is the current standard [7]. Through IPv6 on Bluetooth low energy, an Internet device (computer, phone or tablet) or cloud service can access the Bluetooth low energy device application transparently via standard Internet functionality e.g. using an HTTP REST API or other IP-based protocols. Although not the main intention, an application in a portable device can also access the Bluetooth low energy device when connected point-to-point and still fully utilize the Internet functionality [7]. The cloud can be a public cloud or a private cloud.
[5] CONCLUSIONS:
In summary, it can be foreseen that IoT will enter in every walk of our daily lives thereby increasing the quality of living. IoT can play a notable role in improving the quality of our lives and improving world’s economy. Areas of applications include transportation, healthcare, industrial automation, and response to real time emergencies (natural and man-made disasters) where human decision making is difficult. For example, home garages may open on their own when the residents reach home, coffeemaker make the coffee automatically, delivering good care to people in remote locations by monitoring their activities and their vitals. For IoT to be a success a framework enabled by a scalable cloud is important. The framework allows networking, computation, storage and visualization themes as separate entities thereby allowing independent growth in every sector but complementing each other in a shared environment. New research problems may arise due to the number of devices shooting up, the connection between the physical and cyber worlds and lingering privacy and security issues. Necessary measures and researches have to be taken place continuously to keep the issues in check.

CITATIONS:
[1] Integrating Wireless Sensor Networks with the Web. Authors are Walter Colitti, Kris Steenhaut, Niccolò De Caro
[2] Use case possibilities with Bluetooth low energy in IoT applications white paper Author Mats Andersson Senior Director Technology, Product Center Short Range Radio, u-blox

REFERENCES
[2] Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications Ala Al-Fuqaha, Senior Member, IEEE, Mohsen Guizani, Fellow, IEEE, Mehdi Mohammadi, Student Member, IEEE,Mohammed Aledhari, Student Member, IEEE, and Moussa Ayyash, Senior Member, IEEE
[5] Integrating Wireless Sensor Networks with the Web. Authors are Walter Colitti, Kris Steenhaut, Niccolò De Caro
[7] Use case possibilities with Bluetooth low energy in IoT applications white paper Author Mats Andersson Senior Director Technology, Product Center Short Range Radio, u-blox
[8] Web Messaging for Open and Scalable Distributed Sensing Applications Vlad Trifa, Dominique Guinard, Vlatko Davidovski, Andreas Kamlaris, and Ivan Delchev