A Review on Architecture of Internet of Things (IoT)

Ms. Priyanka D. Patil[1], Ms. Priyanka B. Shinde[2]
[1,2] *Assistant Professor, Department of E & TC Eng. , P.V.P.I.T. Budhgaon, Sangli, India*

***Abstract*- IoT represents a system which consists things in the real world, and sensors attached to or combined to these things, connected to the Internet via wired and wireless network structure. This paper uses Internet of Things as a medium to propose a solution to the problems in various applications. The IoT sensors can use various types of connections such as RFID, Wi-Fi, Bluetooth, and ZigBee, in addition to allowing wide area connectivity using many technologies such as GSM, GPRS, 3G, and LTE. IoT-enabled things will share information about the condition of things and the surrounding environment with people, software systems and other machines. by the technology of the IoT, the world will becomes smart in every aspects, since the IoT will provides a means of smart cities, smart healthcare, smart homes and building, in addition to many important applications such as smart energy, grid, transportation, waste management and monitoring . In this paper we review a concept of many IoT applications and future possibilities for new related technologies in addition to the challenges that facing the implementation of the IoT.**

***Index Terms*- Internet of Things (IoT), MQTT**

I. INTRODUCTION

The Internet of Things (IOT) is an emerging topic which includes the entire world. This technology includes a wide spectrum of networked products, systems, and sensors, which take advantage of advancements in computing power, electronics miniaturization, and network interconnections to offer new capabilities not previously possible. It is going to transform many aspects of the way we live. For consumers, new IoT products like Internet enabled appliances, home automation components, and energy management devices are moving us toward a vision of the "smart home'', offering more security and energy efficiency[1],the main strength of IoT is its high impact on numerous aspects of routine lifestyle and actions of the potential users. The success of IoT from the point of view of private user will be evident in both domestic and working fields. In this situation, e-Health, assisted living, enhanced learning are few of many possibilities where the new paradigm will play a leading role. From the business user's perspective, the most important applications will be in automation and industrial manufacturing, intelligent transportation, logistics, and process management [2]. Hence, IoT can be defined as: "A network of physical objects or 'things' that can interact with each other to share information and take action."

II. HOW IOT WORKS

First, IoT acquires information with respect to basic resources (names, addresses and so on) and related attributes of objects by means of automatic identification and perception technologies such as RFID, wireless sensor and satellite positioning, in other words the sensors, RFID tags and all other uniquely identifiable objects or "things" acquire real-time information (data) with the virtue of a central hub like smart phones.
Second, by virtue of many kinds of communications technologies, it integrates object-related information into the information network and realizes the intelligent indexing and integration of the information related to masses of objects by resorting to fundamental resource services (similar to the resolution, addressing and discovery of the internet).
Finally, utilizing intelligent computing technologies such as cloud computing, fuzzy recognition, data mining and semantic analysis, it analyzes and processes the information related to masses of objects so as to eventually realize intelligent decision and control in the physical world.

III. ARCHITECTURE OF IOT

The reference architecture consists of a set of components. Layers can be realized by means of specific technologies. There are also some cross-

cutting/vertical layers such as access/identity management.

The layers are

- Client/external communications - Web/Portal, Dashboard, APIs
- Event processing and analytics (including data storage)
- Aggregation/bus layer – ESB and message broker
- Relevant transports - MQTT/HTTP/XMPP/CoAP/AMQP, etc.

Devices

The cross-cutting layers are

- Device manager
- Identity and access management [3]

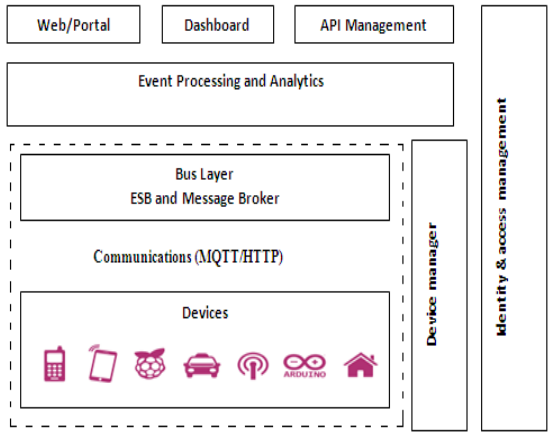

Figure.1. Reference Architecture for IoT

3.1 The Device Layer

The bottom layer of the architecture is the device layer. Devices can be of various types, but in order to be considered as IoT devices, they must have some communications that either indirectly or directly attaches to the Internet. Examples of direct connections are

- Arduino with Arduino Ethernet connection
- Raspberry Pi connected via Ethernet or Wi-Fi
- Intel Galileo connected via Ethernet or Wi-Fi

Examples of indirectly connected device include

- ZigBee devices connected via a ZigBee gateway
- Bluetooth or Bluetooth Low Energy devices connecting via a mobile phone
- Devices communicating via low power radios to a Raspberry Pi[3]

3.2 The Communications Layer

The communication layer supports the connectivity of the devices. There are multiple potential protocols for communication between the devices and the cloud. The most well known three potential protocols are

- HTTP/HTTPS (and RESTful approaches on those)
- MQTT 3.1/3.1.1
- Constrained application protocol (CoAP)

HTTP is well known, and there are many libraries that support it. Because it is a simple text based protocol, many small devices such as 8-bit controllers can only partially support the protocol – for example enough code to POST or GET a resource. The larger 32-bit based devices can utilize full HTTP client libraries that properly implement the whole protocol. There are several protocols optimized for IoT use. The two best known are MQTT6 and CoAP7. MQTT was invented in 1999 to solve issues in embedded systems and SCADA. MQTT is a publish-subscribe messaging system based on a broker model. The protocol has a very small overhead (as little as 2 bytes per message), and was designed to support lossy and intermittently connected networks. MQTT was designed to flow over TCP. In addition there is an associated specification designed for ZigBee-style networks called MQTT-SN (Sensor Networks). In order to support MQTT we need to have an MQTT broker in the architecture as well as device libraries. One important aspect with IoT devices is not just for the device to send data to the cloud/ server, but also the reverse. This is one of the benefits of the MQTT specification: because it is a brokered model, clients connect an outbound connection to the broker, whether or not the device is acting as a publisher or subscriber. This usually avoids firewall problems because this approach works even behind firewalls or via NAT.

3.3 The Aggregation/Bus Layer

An important layer of the architecture is the layer that aggregates and brokers communications. This is an important layer for three reasons:

1. The ability to support an HTTP server and/or an MQTT broker to talk to the devices;
2. The ability to aggregate and combine communications from different devices and to

route communications to a specific device (possibly via a gateway)

3. The ability to bridge and transform between different protocols, e.g. to offer HTTP based APIs that are mediated into an MQTT message going to the device.
4. The aggregation/bus layer provides these capabilities as well as adapting into legacy protocols. The bus layer may also provide some simple correlation and mapping from different correlation models (e.g. mapping a device ID into an owner's ID or vice-versa). Finally the aggregation/bus layer needs to perform two key security roles. It must be able to act as an OAuth2 Resource Server (validating Bearer Tokens and associated resource access scopes). It must also be able to act as a policy enforcement point (PEP) for policy-based access. In this model, the bus makes requests to the identity and access management layer to validate access requests. The identity and access management layer acts as a policy decision point (PDP) in this process. The bus layer then implements the results of these calls to the PDP to either allow or disallow resource access.[3]

3.4 The Event Processing and Analytics Layer

This layer takes the events from the bus and provides the ability to process and act upon these events. A core capability here is the requirement to store the data into a database. This may happen in three forms. The traditional model here would be to write a server side application, e.g. this could be a JAX-RS application backed by a database. However, there are many approaches where we can support more agile approaches. The first of these is to use a big data analytics platform. This is a cloud-scalable platform that supports technologies such as Apache Hadoop to provide highly scalable mapreduce analytics on the data coming from the devices. The second approach is to support complex event processing to initiate near real-time activities and actions based on data from the devices and from the rest of the system.[3]

3.5 Client/External Communications Layer

The reference architecture needs to provide a way for these devices to communicate outside of the device-oriented system. This includes three main approaches. Firstly, we need the ability to create web-based front-ends and portals that interact with devices and with the event-processing layer. Secondly, we need the ability to create dashboards that offer views into analytics and event processing. Finally, we need to be able to interact with systems outside this network using machine-to-machine communications (APIs). These APIs need to be managed and controlled and this happens in an API management system. The recommended approach to building the web front end is to utilize a modular front-end architecture, such as a portal, which allows simple fast composition of useful UIs.

The API management layer provides three main functions:

- The first is that it provides a developer-focused portal (as opposed to the user focused portal previously mentioned), where developers can find, explore, and subscribe to APIs from the system. There is also support for publishers to create, version, and manage the available and published APIs;
- The second is a gateway that manages access to the APIs, performing access control checks (for external requests) as well as throttling usage based on policies. It also performs routing and load-balancing;
- The final aspect is that the gateway publishes data into the analytics layer where it is stored as well as processed to provide insights into how the APIs are used.

3.6 Device Management

Device management (DM) is handled by two components. A server-side system (the device manager) communicates with devices via various protocols and provides both individual and bulk control of devices. It also remotely manages software and applications deployed on the device. It can lock and/or wipe the device if necessary. The device manager works in conjunction with the device management agents. There are multiple different agents for different platforms and device types.

The device manager also needs to maintain the list of device identities and map these into owners. It must also work with the identity and access management layer to manage access controls over devices

There are three levels of device: non-managed, semi-managed and fully managed (NM, SM, and FM).
Fully managed devices are those that run a full DM agent. A full DM agent supports:

- Managing the software on the device
- Enabling/disabling features of the device (e.g. camera, hardware, etc.)
- Management of security controls and identifiers
- Monitoring the availability of the device
- Maintaining a record of the device's location if available
- Locking or wiping the device remotely if the device is compromised, etc.

Non-managed devices can communicate with the rest of the network, but have no agent involved. These may include 8-bit devices where the constraints are too small to support the agent. The device manager may still maintain information on the availability and location of the device if this is available.
Semi-managed devices are those that implement some parts of the DM.

3.7 Identity And Access Management

The final layer is the identity and access management layer. This layer needs to provide the following services:

- OAuth2 token issuing and validation
- Other identity services including SAML2 SSO and OpenID Connect support for identifying inbound requests from the Web layer
- XACML PDP
- Directory of users (e.g. LDAP)
- Policy management for access control (policy control point) [3]

IV. CHARACTERISTICS OF IOT

To make this technology an integral part of our lives, let's have an understanding of the characteristics of the IoT. The "things" in IoT are also called applications that need to have some specific characteristics to make this technology work smoothly and in order to be called a full-fledged IoT system or application. These characteristics are:

1. The things should have a unique identification so that each of them can be distinguished from various objects in the network. If they don't have unique identification then it becomes difficult for developers to work with.
2. Things should be able to detect the presence of other objects, following a rule of autonomy. If they can do so then they can further interact with each other and work accordingly.
3. Things should be able to capture data autonomously.
4. Since there are various communication protocols and technologies that IoT devices will work with, things should be interoperable among various communication technologies.
5. Things should have a service–based operation, such that if any two or more objects are in the vicinity or in contact then they should be able to communicate directly with each other and exchange information and data if necessary.
6. There should be cooperation among autonomous objects (things). If two autonomous objects can interact and cooperate with each other to accomplish any preset or necessary task, it can intensify the value of such an application manifold.
7. Things should be able to operate at low power.
8. Things should be contextual in nature.
9. Things should be programmable by the user.
10. Things should have a fail-safe operation and most importantly secure.

V. CONCLUSION

Designing data-driven IoT solutions is complex due to the scale and heterogeneousness of the devices and connectivity involved. In future IOT is going to become a reality. It will change our life style. But there are many challenges to face related to the deployment, growth, implementation, and use of this technology. The Internet of Things involves a complex and evolving set of technological, social and policy considerations across a diverse set of stakeholders. But it will be an asset for us in future.

REFERENCES

[1] https://www.internetsociety.org/sites/default/files/ISOC-IoTOverview-20151014_0.pdf
[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: Asurvey," *Computer Networks*, vol. 54, pp. 2787-2805, 10/28/ 2010.

[3] https://wso2.com
[4] D. Giusto, A. Iera, G. Morabito and L. Atzori, editors. The Internet of Things, Springer, 2010
[5] John A. Stankovic, "Research Directions for the Internet of Things" IEEE Internet of Things Journal, Vol.1, No.1, pp. 3-9
[6] https://en.wikipedia.org/wiki/Internet_of_Things ,_Jun_25_(2016)