Making sense of interoperability: Protocols and Standardization initiatives in IOT

Ronak Sutaria
Mindtree Research Labs
Mumbai, India.
ronak_sutaria@mindtree.com

Raghunath Govindachari
Mindtree Research Labs
Bengaluru, India.
raghunath_govindachari@mindtree.com

ABSTRACT

Recent developments in connectivity technologies have spurred the adoption of internet connected “smart” devices for remote sensing, actuating and intelligent monitoring using advanced analytics and real-time data processing. As the pace and the scale of such solutions increase rapidly, there will soon be a problem of getting these disparate solutions to work seamlessly together to realize a large scale Internet of Things (IOT). Fortunately, several industry bodies and standards forums are working on extending or adopting the internet protocols to the constrained devices. However, in the near term, disparate islands of solutions are likely to outpace deployment of interoperable standards-based solutions. In this paper, we review and discuss the recent developments in protocols and standardization initiatives for IOT from the perspective of interoperability. In particular, we look at application layer protocols and issues of interoperability of solutions.

Categories and Subject Descriptors

C.2.6 [**Internetworking**]: Standards and Protocols.

General Terms

Standardization

Keywords

IOT Protocols, interoperability, CoAP, protocol stack.

1. INTRODUCTION

A large number of device-to-cloud technologies and solutions are being developed by researchers and innovative startups around the world (A few examples are[1]). While many such projects and solutions do provide integration and control of the devices from the internet, they may not always use standardized protocols and hence could lead to interoperability challenges when used on a global scale. There is an ongoing trend in integrating the Machine-to-Machine (M2M) and Wireless Sensor Networks (WSN) solution with other established Internet services using existing Internet Protocols. This has resulted in bringing the standard IP-centric protocols into the realm of smart devices and smart objects. Solutions such as the iDigi Connect[2], attempt to bring to embedded devices, standard connectivity through TCP/IP stacks and web services through which data acquired can be aggregated over the internet for visualization and analysis.

Two of the biggest challenges that IOT implementations are faced with are the presence of Low-powered devices (which need to function for months or years without getting any power recharge) and the frequent data exchanges over a Lossy network. These unique characteristics and challenges makes the as-is use of the existing internet protocols to be less than ideal and sub-optimal.

A number of different standardization bodies and groups are actively working on creating more inter-operable protocol stacks and open standards for the Internet of Things. As we move from the HTTP, TCP, IP stack to the IOT specific protocol stack we are suddenly confronted with an acronym soup of protocols- from the wireless protocols like ZigBee, RFID, Bluetooth and BACnet to next generation protocol standards such as 802.15.4e, 6LoWPAN, RPL, CoAP etc. which attempt to unify the wireless sensor networks and the established internet.

Internet of Things (IOT) is an ambitious paradigm which significantly increases the scale of connected devices. In order to manage the complexity of such a scale, interworking solutions that can reuse pre-existing technologies seamlessly with newer more efficient technologies is a necessity. Ongoing standardization efforts towards harmonizing internet protocols for wireless sensor networks-based internet of things have raised hopes of global interoperable solutions at the transport layer and below. With that being addressed, the next requirement is to address interoperability at the application layer.

This paper provides a brief primer describing the protocol stacks and standardization initiatives that are being actively researched and are under various stages of implementation. Section 2 provides some of the dilemmas that are emerging when diverse and rapidly prototyped solutions involving smart objects are being architected. Section 3 provides a survey of the research being done in building a standards-driven IOT protocol stack. Section 4 provides an overview of some of the interoperability related issues that need to be addressed. Section 5 ends with a conclusion and outlines the direction of our future work.

---

[1] http://smartthings.com/; http://www.mytaglist.com/; http://www.worldsensing.com/products

[2] http://www.digi.com/products/wireless-wired-embedded-solutions/solutions-on-module/digi-connect/

2. EMERGING DILEMMAS

Newer solutions involving Internet of Things are faced with an emerging dilemma. Smart objects, Low-powered devices and WSNs are trying to grow beyond narrower domains/fields of deployments, wanting to be connected to the cloud/internet, but suffer from constraints and scalability limiters. Then there is the world-wide Internet, very successful and spoiled for choices on de-facto standard technologies (like HTTP, SMTP, SSH, etc.) but never had to bother about severe power/data loss constraints. There is a natural consequent attempt from both sides to assimilate and harmonize. This is where the challenges and hence, the opportunities are. Innovation for this can happen in several directions:

a) The resource constraints may get eased through advances in device computation/networking technologies such as low-powered microcontrollers, longer lasting batteries and alternate power sources, cheaper storage and bandwidth, etc. These developments might enable the introduction of proven internet technologies in the newer IOT domains. But these are usually high-investment high-impact innovations and tend to involve corporations with very large research budgets (For e.g. Intel Atom Processors for IOT[3], etc.)

b) Standardization bodies and alliances are working on defining newer protocols which are optimized for specific use cases/underlying deployment scenarios. Adoption of these protocols and standards involves collaboration among government standards institutes like ETSI, organizations like IETF/IAB and alliances like ZigBee, Dash 7, etc. Large corporations can make choices for implementing protocols and standards which may be independent of standardization efforts (For e.g. Facebook using IBM MQTT[4] protocol for integration with mobile devices.)

c) There will be a (perhaps prolonged) period of mixed deployments bridging islands of highly domain specialized/optimized WSNs with the rest of the internet through gateways, middleware, etc. This is where a large number of incremental innovations might be most visible in the coming few years. But this is also going to require the availability and understanding of a few well defined open standards-backed protocols.

3. IOT PROTOCOL STACK

3.1 A Standards-driven approach

Embedded networked sensor researchers had agreed upon the idea that “the Internet Protocol could and should be applied to even the smallest of devices”. IPv6 which has succeeded IPv4 has a near infinite address space, allowing for 2^{128} or approximately 3.4×10^{38} unique addresses. A new protocol 6LoWPAN was defined to enable IPv6 packets to be carried on top of low-powered and lossy personal area networks (LLNs). A draft architecture for a gateway or middleware which provides interoperability between 6LoWPAN and external IPv6 networks has been defined [1]. The underlying physical and MAC layer for the 6LoWPAN protocol is the IEEE 802.15.4 standard. The 802.15.4-2006 (successor of the 2003 version) is the physical or layer 1 protocol for low-power and low rate (data transfer at 250 kbps) LLNs. The MAC layer amendment to the existing 802.15.4-2006 has been created called IEEE 802.15.4e to a) better support the industrial markets and b) permit compatibility with the modifications proposed within the Chinese WPAN. The key technical element of the new proposed 802.15.4e is channel hopping, which significantly increases robustness against noisy and lossy networks and persistent multi-path fading.

The application layer protocol for introducing the Web-services paradigm in the “Internet of Things” is being worked upon by the IETF Constrained RESTful Environments (CoRE) workgroup [2]. The CoRE group has defined a REST based web transfer protocol called Constrained Application Protocol (CoAP).This protocol includes several HTTP functionalities but has been re-designed to take into account the low processing power and energy consumption constraints of IOT devices. CoAP, similar to HTTP, identifies resources using a Universal Resource Identifier (URI) and allows affecting the resource using similar methods such as GET, PUT, POST and DELETE. But CoAP is not just a blind compression of HTTP.

W. Colitti, et al., at Vrije Universiteit [3], have compared the performance of HTTP with CoAP and have experimented with an IP based Smart Objects Protocol Stack (see Figure 1).

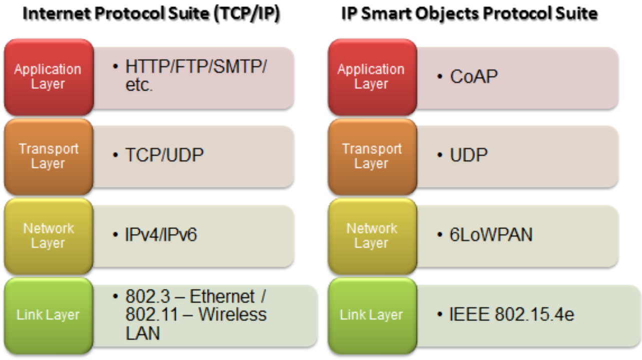

Figure 1 TCP/IP Stack and IP Smart Objects Protocol Stack

The transaction layer of CoAP can contain messages of four types: Confirmable (requires an ACK), Non-confirmable (does not need an ACK), Acknowledgement (ACK for a confirmable message) and Reset (indicates a confirmable message is received but context is missing to be processed). The Request/Response layer is where the REST based communication occurs. The CoAP request has the following format:

```
GET      coap://[<iot_device_ipv6_address>]:
[<port-number>]/[<resource-URI>]
```

The experiment involved a CoAP enabled web-server and a HTTP web-server (on the Contiki embedded OS) has found that requesting the same resource on the device uses 154 bytes per CoAP transaction and 1451 bytes for a HTTP transaction. The power consumption for the CoAP and HTTP transaction is 0.774 mW and 1.333 mW, respectively. Hence, in the particular

---

[3] http://www.intel.com/p/en_US/embedded/innovation/manageability/marisetty-article-manageability

[4] http://mqtt.org/2011/08/mqtt-used-by-facebook-messenger

experiment CoAP transaction was found to consume 42% less power than the HTTP transaction.

An IOT application frequently involves a device or a smart object transmitting information regarding its “state, context or sensory measurements” to other clients or devices. CoAP supports a functionality called “observations” wherein a client (the observer) can register itself to the resource (the subject) by means of a modified GET request. The server establishes an observation relationship between the client and the resource. This functionality avoids the frequent server-polling or keep-alive sessions that clients need to do in the case of a HTTP based connection. CoAP has been built over the UDP since the minimally required reliability is achieved through the transaction layer of the protocol.

Researchers from the OpenWSN team[5] from Berkeley are creating an open-source implementation of protocol stacks based on Internet of Things standards, using a variety of hardware and software platforms. A detailed guide for implementing the 802.15.4e standard for different micro-controllers has been prepared and made freely available on their site.

3.2 Other IOT Alliances and Protocols

Besides the IEFT standards based initiatives, there are several other protocols which are also under various stages of discussions, development and implementation. Some of the protocols which are relevant from an IOT solution implementation perspective are:

a) Message Queue Telemetry Transport (MQTT) – Developed by researchers at IBM, it is designed as a lightweight publish/subscribe messaging transport connectivity protocol. The protocol has been integrated with the IBM WebSphere application server

b) ZigBee Alliance – ZigBee (XBee) is a set of application profiles for creating low-rate wireless mesh networks which has been built upon the 802.15.4-2003 standard. While ZigBee is not directly comparable to IEFT standards like 6LoWPAN, it has been extensively implemented in small-scale ad-hoc networking smart-home and smart-objects related applications.

c) DASH 7 Alliance – Dash7 works at the 433 MHz frequency range and is used, amongst other uses, for tag-to-tag communications. It typically has a range up to 1 KM and the range reduces as the data-transfer bit rate is increased.

d) BACnet – This is a communications protocol for Building, Automation and Control networks. BACnet was standardized in 1995 and became an ISO standard in 2003. It is essentially used in HVAC systems (heating, ventilation and air-conditioning), lighting control, access control, etc.

4. Application Level Interoperability Scenarios - MQTT & CoAP

The MQTT protocol specification [4] describes the protocol to be ideally suited for resource constrained environments, which have characteristics such as “a) the network is expensive, has low bandwidth or is unreliable and b) runs on embedded devices with limited processor or memory resources.” MQTT is an open publish/subscribe protocol designed for asynchronous transfer of telemetry messages. It uses a broker-based asymmetric architecture where the complexity is moved to the higher resource capable broker nodes. While MQTT is based on TCP/IP stack, MQTT-S is an extension for non-TCP/IP stacks, keeping low end sensor devices in mind.

MQTT-S attempts to provide an abstraction for asynchronous communication using content-centric communication paradigm rather than network address centric. Such a decoupling is expected to allow scaling and dynamic application topologies [5]. However, of the three principal types of publish/subscribe systems, topic-based systems are the easiest whereas type-based and content-based systems are more difficult to achieve. Type-based communication demands that type information be additionally exchanged as meta-data among the publishers and subscribers which adds substantial overheads for the constrained devices.

The CoAP protocol, as described in Section 3.1, is being standardized by IETF for Low-powered and Lossy Networked environment. Unlike the MQTT, CoAP while being a single protocol, is conceptually separated into two sub-layers viz., messaging layer and a request-response sub layer. The former provides asynchronous messaging services over datagram transport. The latter provides handling and tracking of requests and responses exchanged between a client and servicer side application endpoints. Thus, CoAP provides direct support for web services. Furthermore, CoAP specification provides stateless mapping between HTTP and CoAP at the proxies.

A situation may arise where an application running on a low-powered device uses the MQTT protocol and may need to communicate and interact knowledgeably with a remote device which may only understand CoAP based messages. This is a plausible real-world scenario which can, as an example, occur when the mobile Facebook app (via the Messenger app) would like to communicate with not just a group of people but also internet-enabled devices. The “Internet of Things” is specifically about people and devices interacting seamlessly. In the case of the Mobile Facebook app, while the messenger app is designed for groups of people to be able to chat easily, if one of those messages involved retrieving data from a remote sensor or sending a command to an actuator, the message would need to be sent in a protocol which is understood by the remote device. A large number of sensing and actuating devices are being built using Open source hardware like Arduino. A number of different CoAP server side libraries have already been ported to the Arduino platform. Hence, an interoperability issue arises when an MQTT enabled client wishes to send a message to a CoAP enabled server or vice-versa.

A seemingly obvious solution, maybe from the experience of engineering in the “Internet” world, would be to simply enable the client and the server to be able to handle both the MQTT and CoAP protocols and depending on the remote device capabilities, to send the message in a suitable protocol. But in an environment where the processing and memory availability is highly constrained, implementing multiple protocols on these low-powered devices would not be an ideal situation. Implementing

---

[5] http://openwsn.berkeley.edu/

multi-protocol libraries and interoperability at the device level should be avoided. Hence, this leads to the need for IOT gateways or middlewares which can provide protocol level interoperability, by translating the data, context or the meaning of the message from one protocol to another protocol. In essence, the IOT gateway should be able to handle a situation where a message comes to a mobile device over the MQTT protocol and the same message, keeping the data (in case of sensing related information) or the command (in case of actuating related instructions) should be translated correctly to the CoAP protocol, without changing the meaning or context of the message.

5. CONCLUSION

While the scenarios as described in the above section may not be an immediate requirement in terms of interoperability, it is representational of the kind of needs that will arise at a rapid pace as IOT solutions get built based on varied and custom protocols.

Interoperability goes beyond providing protocol proxy type of services. Middlewares which provide interoperability need to be aware of the context and meaning contained both in the header (fixed and variable) as well as in the payload of the message. These middlewares will play a crucial role in keeping the applications on resource constrained devices optimized and at the same time providing seamless experience for the solution developers as well as the end user.

6. REFERENCES

[1] K. Kim, S. Yoo, H. Kim, S. D. Park and J. Lee, "Interoperability of 6LoWPAN," *draft-daniel-6lowpan-interoperability-01, IETF,* vol. 7, 2005.

[2] Z. Shelby, K. Hartke, C. Bormann and B. Frank, "Constrained application protocol (coap)," *draft-ietf-corecoap-07,* 2011.

[3] W. Colitti, K. Steenhaut and N. De Caro, "Integrating Wireless Sensor Networks with the Web," *Extending the Internet to Low powerand Lossy Networks (IP+ SN 2011),* 2011.

[4] "MQTT v 3.1 Protocol Specification," IBM, [Online]. Available: http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/MQTT_V3.1_Protocol_Specific.pdf. [Accessed 23 09 2012].

[5] U. Hunkeler, H. Truong and A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks," in *3rd International Conference on Communication Systems Software and Middleware and Workshops. COMSWARE 2008.*, IEEE, 2008, pp. 791--798.