IoT application components

A Thinnect Whitepaper
Jürgo-Sören Preden, PhD

While there is general agreement in the market that IoT represents a significant opportunity and that the technologies required for realizing IoT applications need to be selected and developed, there is also a lot of confusion in regard to what technologies are specifically suited for IoT. Some people advocate the need for "standard" technologies, such as Thread, CoAP or MQTT but what most people don't recognize is the fact that any one of these "standard" technologies is just one (small) piece of an IoT application and that many other technology components are also needed to realize a complete application. The objective of this paper is to outline the required components in an IoT application communication stack, and to describe the role of individual technologies in applications so as to bring more clarity.

In Figure 1, below the communication technologies of a typical IoT solution are depicted. The upper layers (depicted in green) typically run on the Cloud, although some of these functions can be also run on Gateways, depending on the architecture of the specific solution. The lower layers (depicted in blue) run on the devices in a local wireless network. The wireless network can be characterized as having: limited bandwidth, limited resources, relatively high latency and potentially high error rates because of a varying wireless noise environment. Below we discuss the individual technology components in more detail, to provide the reader with an overview of the purpose of each individual component and its associated constraints. Where possible we also provide a short overview of the available technologies that can be used to implement the specific functionality.

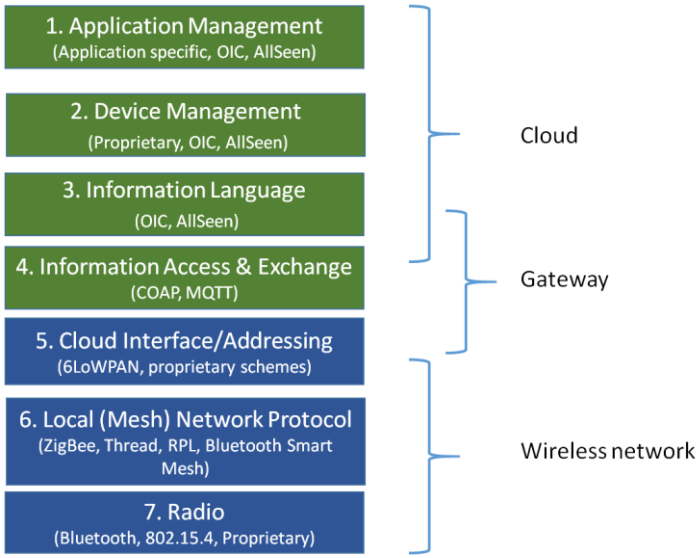

Figure 1 Layers in the IoT communication stack

www.thinnect.com
440 No Wolfe Rd
Sunnyvale, 94085

1. **Application management**

The application management layer is responsible for managing the behavior and functionality of the application and also managing the information collected by the application. The application management layer is typically custom built to meet the needs of a specific application (e.g., lighting control, HVAC management, production management). The OIC and the AllSeen alliance offer components that provide functionality to solve some of these tasks.

Thinnect offering: Thinnect offers cloud-based tools that provide an overview of application behavior, providing information on device activity and abnormalities.

2. **Device management**

The device management layer is needed to ensure that all IoT devices that are part of an application (the sensors and actuators) are performing optimally and delivering the services that are expected from these devices. Device management is a somewhat generic task and therefore can be isolated from the application management and the application specific information. There exist some generic proprietary device management platforms, which claim to be usable for any application (e.g., Arrayant, Exosite, Parse). It must be noted that the listed solutions all claim to also offer IoT application data management capabilities, however, the actual application management functionality must still be created based on the application's requirements. The OIC and the AllSeen alliance also offer components that provide functionality to solve some of the tasks in device management.

Thinnect offering: All devices in a Thinnect network report network diagnostics information to the cloud, Thinnect offers cloud-based tools that provide an overview of the network status, link quality and any abnormalities in device connectivity.

3. **Language for information representation**

While the language for information representation is not a technology component or communication layer per se, it is a required component to ensure that the data generated by individual application components is usable by other components. The OIC and the AllSeen alliance are both working on (non-compatible) standards to offer a language to enable device interoperability.

Thinnect offering: Thinnect uses a proprietary binary protocol at the network level, which enables rule based control and efficient node to node communication. Thinnect also offers conversion tools, which enable converting commands and data requests to and from standard JSON formats to the binary format.

4. **Information access and exchange**

The information access and exchange layer is required to make the data and information generated by sensors available to Cloud based application components.

www.thinnect.com
440 No Wolfe Rd
Sunnyvale, 94085

MQTT (Message Queuing Telemetry Transport) offers queue based access to data, assuming that sensors provide data to the queue. The MQTT components manage the queue, insert data items into the queue and provide data items to clients that have subscribed to data streams relevant to them.

Constrained Application Protocol (CoAP) is an application-level protocol for querying data from resource constrained nodes using a request/response scheme. CoAP was inspired by information exchange patterns used in the Internet, however the same patterns are not (in most cases) applicable to IoT devices as the devices are expected to provide a persistent service, rather than just responding to individual requests.

Both of the technologies described above have their virtues and shortcomings, however, since these are not the focus of this current paper, they will not be discussed further. A service based architecture, where clients subscribe directly to data they need, is much better suited for IoT but to date, no standard has been proposed for such a data exchange scheme for resource constrained devices.

Thinnect offering: Thinnect offers a mesh network adapter at the cloud level, which manages wireless nodes, ensuring that all nodes are performing the tasks assigned to them.

5. Interfacing to cloud, addressing

Interfacing a resource constrained wireless network to the cloud requires an adaption layer as it is not feasible to use internet addressing schemes in a wireless network due to the high overhead this generates. The overhead from using IP addressing would be too high as the length of an IPv6 address is 16 bytes so sender and receiver IPv6 addresses would take up 25% of the packet size in the case of 802.15.4 and 40% in case of Bluetooth LE Advertising Packet. So in a wireless network local (short) addresses should be used, which makes the communication feasible.

6LoWPAN offers schemes for address translation (from IPv6 addresses to local network addresses) and packet fragmentation, enabling communication of IPv6 packets to a resource constrained wireless network. However, 6LoWPAN does not address the inefficiency of the payload data – due to the limited bandwidth available in IoT wireless networks a binary representation should be used for the data and metadata to use the network resources more efficiently. In addition, the interface layer should also address the inherent limitations of IoT wireless network, such as their inability to handle high number of packets, the communication delays involved, high packet loss etc. None of these issues are being addressed by the 6LoWPAN protocol.

Thinnect offering: Thinnect will offer 6LoWPAN compatibility at Gateway level with flow control implemented in the Gateway to protect the wireless network from overload.

6. Local (mesh) networking protocol

The local networking protocol is responsible for moving packets received at the gateway to the individual devices on the network. In addition to existing protocols, such as Zigbee and RPL, new protocols are emerging, such as Thread and Bluetooth Smart Mesh. The existing network protocols have severe

limitations – Zigbee networks need careful planning, are designed for single application solutions and are not self-configuring. With the RPL protocol all nodes are dependent on the Gateway for communication – the network structure is hierarchical and even node-to-node communication must occur through the gateway.

Unfortunately the emerging protocols also have severe limitations: a Thread network supports only 32 routers, thereby limiting the network scalability to the extent where it is not feasible for commercial applications. The upcoming Bluetooth Smart Mesh will be a flooding mesh protocol, which limits the properties of the network severely – the data rate and maximum number of nodes in this network will be very limited.

Some emerging mesh network specific protocols offer a solution for effective communication in wireless networks, however these solutions are just gaining market traction.

Thinnect offering: Thinnect offers a patented clustered mesh network protocol, which offers high resilience, is self-forming and self-healing.

7. Radio

A few radio protocols exist that are suitable for IoT end device communication.

802.15.4 was originally designed for resource constrained embedded devices, it has a packet size of 127 bytes and the medium access scheme used is CSMA/CA (Carrier Sense, Multiple Access / Collision Avoidance), which is effective for a small number of nodes but fails to deliver once the concentration of nodes gets too high.

Bluetooth was originally designed for point to point communication and it is very well suited for this but at the expense of quite substantial power consumption. Bluetooth Low Energy (BTLE or Bluetooth Smart) was added to facilitate communication between low power resource constrained devices. A meshing capability is being added as an afterthought by making use of the advertising mode in BTLE – this being driven by the need to maintain compatibility with existing Bluetooth LE devices. As the size of an advertising packet in BTLE is just 41 bytes the capabilities of Bluetooth Smart Mesh are fundamentally limited due to its resulting bandwidth limitation. The medium access scheme used in BTLE is random, which potentially also results in higher packet loss.

Thinnect offering: Thinnect network stack runs currently on IEEE802.15.4 but the network stack is PHY agnistic, i.e., it can be ported to run also on other radio protocols.

This document summarizes the main aspects relevant for IoT communication technologies and the relevant Thinnect offerings. If you would like to receive further information, please contact:

Jurgo Preden,
CEO
jurgo@thinnect.com