Store   Subscribe   About Us   Contact Us   RSS   Advertising

REGISTER | LOG IN

Technologies   News   Markets   Learning Resources   Community   Companies   Part Search

Ask the Experts	Interviews
Blogs	Our Editors
Bob Pease	STEM Starter Tournament
Contributing Technical Experts	Pop Quizzes
Engineering Hall of Fame	Engineering Bracket Challenge

Engineering Essentials: IoT Standards and Frameworks

Building the Internet of Things requires cooperation that frameworks can streamline.

William Wong | *Electronic Design*   May 22, 2015

SHARE   Tweet   G+1   Recommend 11   COMMENTS **2**

You're not building the next smart sports band, so should you care about Internet of Things (IoT) frameworks?

IoT is the latest buzzword that has replaced everything from the cloud to machine-to-machine (M2M) communication. Essentially, it is the same thing that was going on, though the scope tends to be larger. The devices involved may number in the millions, with hundreds or thousands on-premises.

Programming for an IoT environment is no longer simply a TCP/IP socket connection. Even a Transport Layer Security (TLS) connection is insufficient for providing system security. Likewise, more programmers are having to deal with transient connections or connections with varying bandwidths. Wireless connectivity brings even more complexity to the environment, and data may hop through any number of gateways that may or may not perform data massaging along the way.

RELATED

» IoT Is Here Whether You Like It or Not
» Can The Internet Of Things Be Secure?
» The Internet Of Things Is A Standards Thing

Standards and frameworks help simplify programming chores. Sometimes the choices are dictated by a chosen platform. There are a number of vendors that provide hardware and the software support, and developers will be hard pressed to work outside of these confines. There are other open solutions that do not necessarily address an end-to-end approach, but rather, leave providing significant portions of the environment to the programmer. They also may not address issues such as system and power management, remote updates, and so on.

Here we examine some of the alternatives available. The collection is by no means exhaustive, as this is one programming area with a very large number of options. No one platform is suitable for all applications, although many purport to be so.

Typically IoT frameworks are built on top of or otherwise utilize standard communication protocols like TCP/IP. They may be more specific in support such as using IPv6 over Low power Wireless Personal Area Networks (6LoWPAN). They may also use higher level protocols like Message Queue Telemetry Transport (MQTT), Advanced Message Queuing Protocol (AMQP), or Extensible Messaging and Presence Protocol (XMPP). It is also possible to use something like Object Management Group's (OMG) Data Distribution Service (DDS).

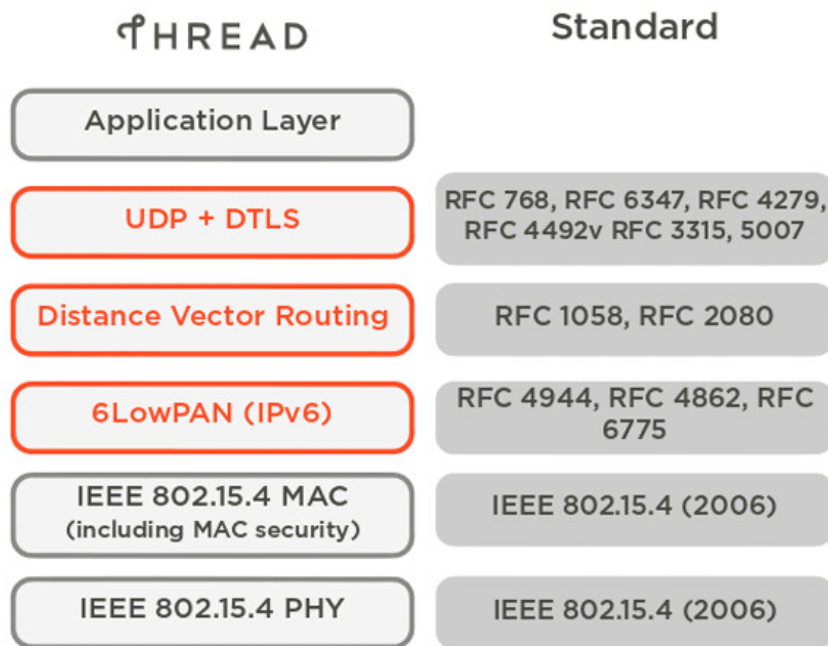Often the challenge lies in determining the scope of the application as well as the

framework. Some solutions are designed for millions of nodes, while others are more suitable for dozens or hundreds of nodes.

 Download this article in .PDF format
This file type includes high resolution graphics and schematics when applicable.

Open Frameworks

1. The Thread Group's Thread framework is built on existing standards like 6LoWPAN and 802.15.4 to address wireless IoT applications.

One organization that has garnered significant vendor support for its wireless framework is the Thread Group. It provides an application framework built on existing IETF and IEEE standards like IEEE 802.15.4 *(Fig. 1)*. It is designed to handle over 250 devices in a Personal Area Network (PAN) or Home Area Network (HAN). These often will be low-power devices that typically sleep, so timing of communication will be critical to efficient operation. Parent routers can hold data for sleeping devices.
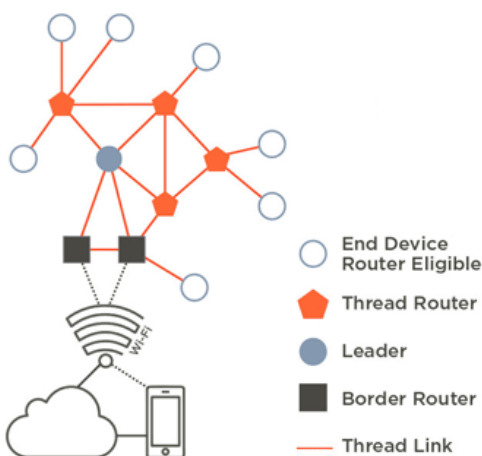
2. The Thread architecture is similar to most wireless, mesh network solutions. Routers provide connectivity to the cloud for end devices. Routers may also provide end node services such as providing local sensor data.

The Thread architecture *(Fig. 2)* is similar to most wireless, mesh network solutions. The system is designed so is it possible to build a system without a single point of failure. This requires at least two border routers, which may not always be possible or economical. Border routers provide a gateway between the Thread network and another network (e.g., a local wireless or wired network).

Within the network, each device has an IPv6 address plus a shorter HAN alias. Devices within the local network can use the HAN addresses while IP addressing is used for services on the other side of the boarder routers. Devices can router eligible or end devices. The latter only communicates with a router. Devices that can be a router are enabled as necessary.

A router maintains state with all other routers in the network using a trickle mechanism and the MLE (mesh link establishment) protocol. The Leader router manages the network. Another router will take over for the Leader if the Leader fails or becomes incommunicado. New nodes can be authorized by a user with an app running on a GUI

device like a smartphone .

The Thread Group is also working with other organizations like the ZigBee Alliance. The Thread stack is used in frameworks like ARM's mbed. The Eclipse Open IoT Working Group has two IoT frameworks: the Kura OSGi-based gateway framework and the Mihini framework for the Lua programming language.

Kura uses the same Java-based OSGi framework from the OSGi Alliance that the Eclipse IDE is based upon. OGSi runs Java bundles. Kura has bundles that provide services like remote management and system configuration, in addition to cloud and data services. Kura components are designed as configurable OSGi Declarative Services exposing the service API with the ability to raise events.
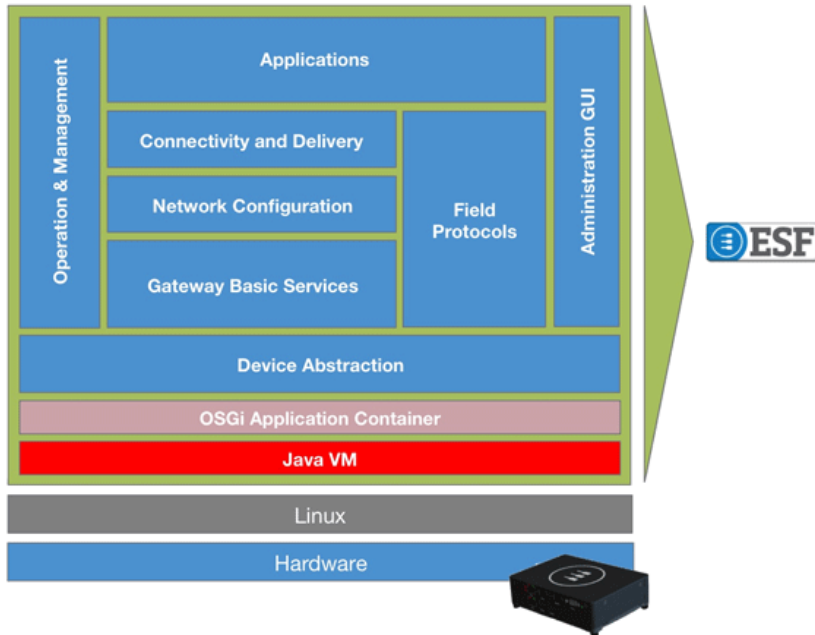
3. Eurotech's Everyware Software Framework (ESF) is the basis for the Eclipse Open IoT Working Group Kura.

Kura is based on work from Eurotech. The firm's Everyware Software Framework (ESF) provides the same common services as Kura *(Fig. 3)*. Eurotech builds on this with features such as reverse NAT support and the Everyware VPN as well as Everyware Cloud Web Console management tool. ESF uses MQTT as its transport mechanism. It can handle device updates and remote diagnostics.
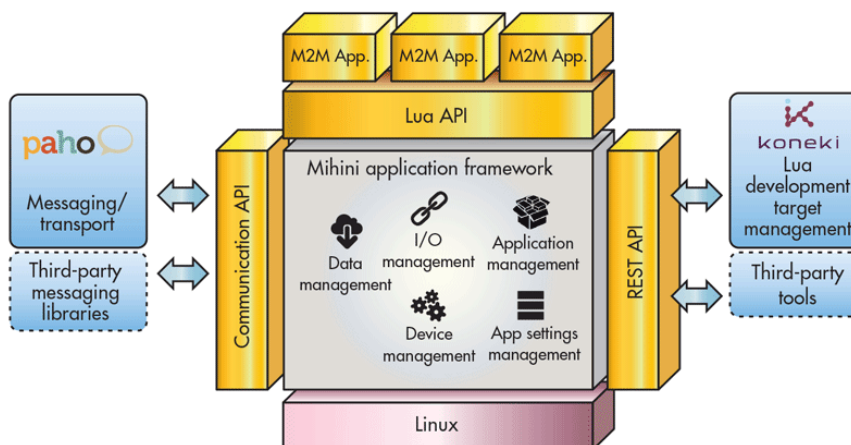
4. Mahini works with the Poho open source MQTT implementation.

The Mihini framework uses the Lua programming language. It is based on work done by Sierra Wireless. Developers can take advantage of Eclipse's Koneki tool suite to develop Mihini-based applications. Mihini also works with the Eclipse Poho project services that provide MQTT support *(Fig. 4)*. Poho include Lua support in addition to MQTT drivers for C/C++, C#, Java, Javascript, Python and Go. Mahini and Kura can run on platforms like Texas Instrument's BeagleBone Black (see "Arduino, Raspberry Pi or BeagleBone?").

IoTivity is another open-source project that targets IoT applications. It is sponsored by the Open Interconnect Consortium (OIC) and hosted by the Linux Foundation. The IoTivity framework incorporates four major building blocks: discovery, data transmission, data management, and device management. It runs on Linux platforms including Ubuntu, Tizen, and Android, as well as Windows 8, iOS, and Arduino.
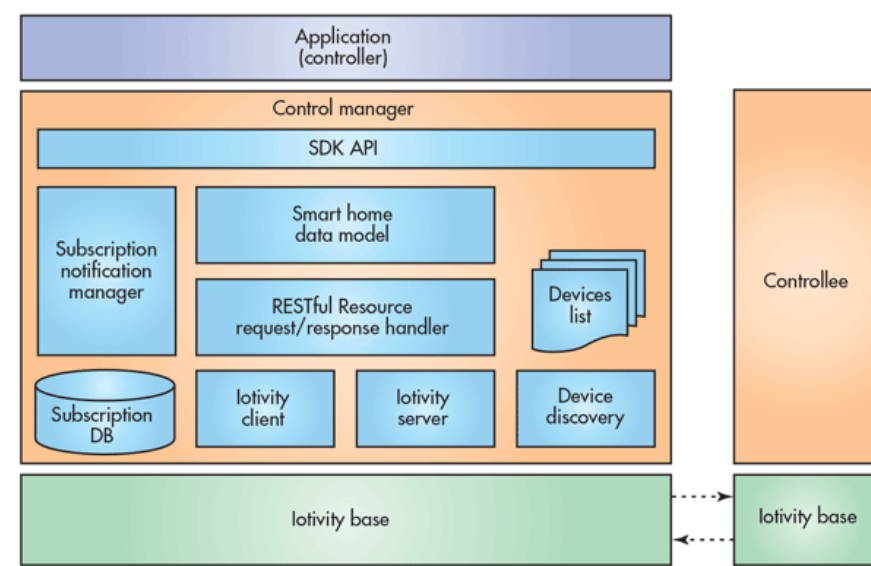
5. The IoTivity Control Manager uses REST to communicate and manage devices within the network.

IoTivity's architecture uses profiles on the application side. The Protocol Plugin Manager handles the other side of the framework where one or more transport drivers exist. Other managers include a Control Manager (CM), Thing Manager (TM) and Soft Sensor Manager (SSM). The CM manages teh connectivity within the network using a REST interface (Fig. 5). The TM allows developers to manager a group of devices or "things" within the network. The SSM handles physical, as well as logical/virtual sensors.

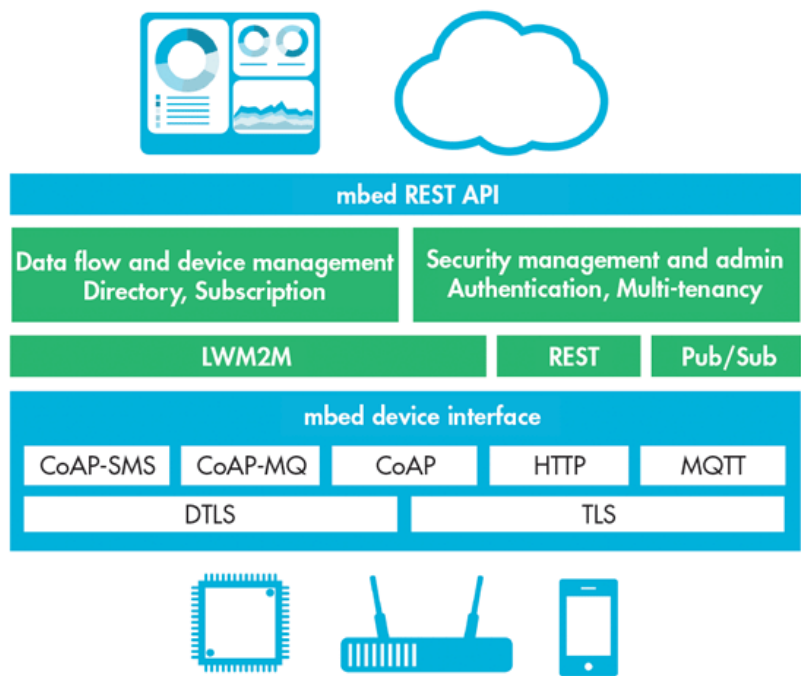ARM's mbed is a bit more focused. It starts as a development environment for ARM Cortex-M microcontrollers. It includes development tools as well as the mbed OS. The environment can be used to develop standalone applications as well as networked applications.

6. The mbed Device Server provides a REST API to manage devices within an mbed network.

The mbed Device Server is an optional component that also uses a REST API (Fig. 6). It provides links between devices as well as connections to cloud-based services. It supports C, Java SE, and Java ME clients. The architecture supports open source, data

communication, and device management protocols like CoAP/HTTP, MQTT, TLS/TCP, DTLS/UDP, and OMALWM2M.

The environment can support any number of cloud services. One that has been integrated with Freescale K64F Freedom board is IBM's cloud services (see "mbed IoT Starter Kit Links to IBM's Cloud"). The ARM mbed Ethernet Starter Kit can be linked to an IBM account to start IoT development immediately with the Freedom board.

Vendor Frameworks

There are a number of vendors that provide environments that address everything from the hardware on up. This can greatly simplify system implementation allowing developers to concentrate on application-specific details. These type of frameworks tend to target specific application areas such as industrial sensor networks.

One example in this is Linear Technology's SmartMesh technologies. SmartMesh IP supports 6LoWPAN and 802.15.4e. Its terminology for nodes is "motes". SmartMesh WirelessHART uses the WirelessHART (IEC 62591) standard. Their Eterna SoC is based on an ultra-low power Arm Cortex-M3.

A key feature of SmartMesh is the robustness of the mesh network. Its self configuring and the network will adjust if a mote becomes inaccessible. The protocol stacks implement a Time Synchronized Mesh Protocol (TSMP). The stack uses a Time Slotted Channel Hopping (TSCH) media access layer (MAC) that divides time into "slots". Timeslots are mapped to channels with a pre-assigned hopping sequence.

Motes transmit, listen, and sleep on a precise schedule. Packet exchanges are synchronized so there are no packet collisions. Packets are scheduled for energy efficiency with no extra, transmit side preamble or receive side guard interval. The approach allows multiple transmissions to occur simultaneously on different channels, thereby increasing overall network bandwidth. SmartMesh is designed to implement very large industrial networks.

Oracle and Microsoft are major software vendors that are large enough to have their own approach to IoT that garners a significant following—both cloud-based solutions and services, such as Microsoft Azure and Oracle Cloud Computing services. These database services are often the cloud target for other IoT frameworks.

Microsoft Windows 10 IoT Core was introduced at Microsoft's Build 2015. It targets IoT end nodes. The core is a compact version of Windows 10 that runs on platforms like Raspberry Pi. It primarily provides communication and operating system services rather than a full Windows 10 GUI platform. An even more compact implementation called Windows Remote Arduino allows applications running on an Arduino microcontroller to communicate with a Windows 10 platform, including the Windows 10 IoT Core.

Oracle's IoT solution takes advantage of its Oracle database and Java platforms to provide the infrastructure of IoT solutions. Partners like PTC fill in the gaps using these platforms. PTC's Axeda provides an end-to-end approach using Axeda Connect connectivity middleware: Axeda Build, which includes data management and scripting support, and Axeda Manage, with web-based applications for managing IoT networks.

All these frameworks have a number of common goals, including reducing time to market, offloading core development, and providing a coordinated communication and security environment. Although many use standard protocols and database solutions, they rarely have the ability to communicate across frameworks. This can lead to environments with mixed solutions in place that may only exchange information and control in the cloud. The advantage of using standards at the communication level is that they can typically coexist even with multiple wireless installations in place.

Unfortunately we have had to gloss over the details of the frameworks to provide an overview of some available solutions. The challenge for developers is identifying the aspects of the framework they will have to deal with as well as the capabilities that the frameworks provide.

In any event, anyone building IoT solutions is unlikely to build their own framework from scratch, given the complexity and components involved that range from microcontroller-based applications communicating with database servers in the cloud.

Download this article in .PDF format
This file type includes high resolution graphics and schematics when applicable.

Discuss this Article   2

Mr Invisible   on Jun 4, 2015

Aaack! Don't say "on premise!" I don't know how that incorrect word substitution got started. But when speaking of concepts along the lines of "here in the building," use the phrase "on-premises." A "premise" is a proposition supporting an argument.

That aside, this is a great article. It provides a good overview of the complex world of IoT-related standards and frameworks. The complexity and enormity of the task engineers face here is head-spinning. It makes me long for the days of Orville and Wilbur.

Log In or **Register** to post comments

Michael.Browne *(staff)*   on Jun 4, 2015

You're correct about the use of "on premise," which we fixed in the article above. Thanks for the good eye!

Log In or **Register** to post comments

Please Log In or **Register** to post comments.

Related Articles

IoT Is Here Whether You Like It or Not   5

Can The Internet Of Things Be Secure?   4

The Internet Of Things Is A Standards Thing   1

Wearing Your Technology

Arduino, Raspberry Pi or BeagleBone?   9

Search Parts Fast

Search by number or manufacturer

	SEARCH

powered by:

sourceesb

ElectronicDesign.com

Technologies   News   Markets   Learning Resources   Community   Companies   Part Search

Site Features   **Penton Corporate**   Search ElectronicDesign.com

Media Center   About Us

Newsletters   Privacy Policy

RSS   Terms of Service

Site Archive   Contact Us

Sitemap

Submit Articles

View Mobile Site

Follow Us

Electronic Design Related Sites

Machine Design   **SourceESB**   **Microwaves & RF**   **Power Electronics**   **Hydraulics & Pneumatics**   **Defense Electronics**   **Global Purchasing**

Electronic Design Europe