Thread and Zigbee for home and building automation

Roberto Sandre
Systems Engineer
Texas Instruments

Introduction

The applications and services enabled by home automation technologies have inspired authors, screenwriters and cartoonists for decades.

But finally, advances in wireless communication, combined with the increased penetration of distributed cloud computing and the launch of innovative user interfaces like voice, have popularized the adoption of smart devices in the home, controlled by digital assistants like the Amazon Echo.

An interesting corollary to that adoption is the rebirth of home automation in a different application context. Rather than a “Jetsons”-like solution with home control at the user’s fingertips, smart home devices do things autonomously, providing convenience and cost savings.

While more glamorous in the residential domain, the advantages that automation brings are not just for homeowners. Automation is even more important in commercial and industrial building domain, where energy consumption and device maintenance have a larger cost impact. For instance, the amount of electricity used by heating, ventilation and air conditioning (HVAC) and lighting systems in office spaces can be managed efficiently with fully autonomous devices that balance environmental conditions, space occupancy and instantaneous energy costs to make both economic and green decisions.

Although a significant percentage of communication in this space is still wired, we see a trend of migrating it to wireless. The choice of which wireless technology to use for the deployment ultimately depends on several factors: the power class of the device (battery-powered), its form factor, the type of traffic profile it needs to support (streaming traffic at high throughput or infrequently sending and receiving actuating commands) and integration with existing ecosystems.

Today, many communication technologies enabling device-to-device, device-to-cloud and device-to-mobile wireless infrastructures are at the heart of home and building automation, including Wi-Fi®, *Bluetooth*® and Bluetooth low energy, Sub-1 GHz, Zigbee and Thread.

Given the ubiquity of access points and smartphones, Bluetooth low energy and Wi-Fi are popular for home and building automation devices like Internet protocol (IP) cameras and door locks, as well as appliances and wearables. Sub-1 GHz-based products, which leverage extended range and penetration capabilities, have also been widely adopted in automated security and safety applications.

However, factors like network size scalability, power consumption, fault tolerance and a lack of a full embedded device-to-device interoperable communication model have lead others to adopt 802.15.4-based technologies like Zigbee and Thread.

Zigbee and Thread offer a native standard mesh-based networking solution that is inherently low power, providing several years on a coin-cell battery for typical sensor and actuator application profiles such as door/window sensors, smoke detectors, light switches and key fobs. The recent introduction of hub-devices with an integrated 802.15.4 radio, like the Amazon Echo Plus and the Nest Protect, have opened up the possibility for rapid proliferation of edge nodes powered by Zigbee and Thread.

This paper will focus on the benefits that Zigbee and Thread provide for home and building automation applications by delivering a solution where range, power consumption and interoperability work at scale.

Table 1 compares the different wireless technologies for home and building automation.

Low power in a large and diverse network

Zigbee and Thread technologies are standard-based protocols that primarily operate in the 2.4-GHz band and provide a built-in mesh networking, security and application infrastructure for embedded, low-power and low-cost devices.

	Zigbee	Thread	Bluetooth low energy	Wi-Fi	Sub-1 GHz
Band	2.4-GHz industrial-scientific-medical (ISM)	2.4-GHz ISM	2.4-GHz ISM	2.4-GHz/5-GHz ISM	Sub-1 GHz with regional bands
Throughput	250 kbps	250 kbps	Can go up to 2 Mbps	Can go up to hundreds of Mbps	Typically a few kbps
One-hop range	Up to a few hundreds of meters; routers can extend range through multihop	Up to a few hundreds of meters; routers can extend range through multihop	Can go up to a few hundreds of meters with long-range mode in Bluetooth 5	Tens of meters; extendable using multiple access points	Up to a few kilometers
Battery type and life	Can operate on a coin cell for a few years	Can operate on a coin cell for a few years	Can operate on a coin cell for a few years	AAA/AA for years	Can operate on a coin cell for many years
Topology	Mesh	Mesh	Point-to-point, mesh	Star; some mesh enhancements emerging for range extension	Mostly star implementations, with some initiatives for mesh
Traffic profile	Best for device-to-device communication in many-to-one, one-to-many, many-to-many	Best for device-to-device communication in many-to-one, one-to-many, many-to-many	Best for device-to-smartphone and smartphone-to-device	Best for device-to-cloud and cloud-to-device, support for one-to-one, one-to-many and many-to-many	Suitable for device-to-device communication in many-to-one and one-to-many
Protocol layering	Network and application	Network	Network and application	Link layer, but all IP standards can run on top	Mostly proprietary implementations with different layering
Certification program and interoperability	End product certification	Stack certification	Stack certification	Data-link layer and some upper-layer stack certification	N/A
Security	Network-wide encryption and authentication through install code	Password-based authentication with Datagram Transport Layer Security (DTLS)	Asymmetric encryption for key generation and exchange, connection pairwise keys	Password and certificate-based authentication, supports all IP-based security standards	Depends on proprietary implementation
IP connectivity and support	Requires gateway to perform IP address translation	Native IPv6 addressing; requires router to convert from 802.15.4 to an additional IP interface	Requires gateway to perform IP address translation	Native	Requires gateway to perform IP address translation

Table 1: *Wireless technologies comparison for home and building automation*

The Zigbee Alliance and Thread Group maintain, promote and manage the Zigbee and Thread standards, respectively, and they both leverage a common underlying data-link communication layer designed and maintained by the Institute for Electrical and Electronics Engineers (IEEE), namely 802.15.4. **Figure 1** shows Zigbee and Thread protocol layering.

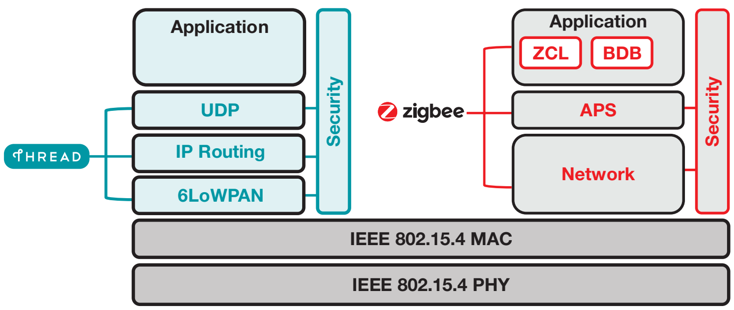

***Figure 1**. Zigbee and Thread protocol layering.*

The 802.15.4 standard specifies the Media Access Control (MAC) and physical (PHY) layer of the Open Systems Interconnection (OSI) communication model, implementing a personal area network (PAN) that guarantees a reliable hop-to-hop link for the transfer of upper-layer data frames at very low-power operations.

Since higher, less timing-sensitive layers are implemented in software, 802.15.4-based standards like Zigbee and Thread can be implemented as different software variants that run on the same silicon [as is the case for the newly launched Texas Instruments (TI) SimpleLink™ Multi-Standard CC2652R wireless microcontroller (MCU)]. With a single unique hardware design, the corresponding firmware can be loaded at the factory or upgraded in the field, providing a simplified and future-proof solution. Among the 802.15.4 specification, both Zigbee and Thread chose the asynchronous mode of operation for the efficient exchange of small packets in a low-power wireless network. Devices do not often generate data but can wake up and reliably send packets with an extremely short latency of tens of milliseconds.

Regardless of the data's destination in the network (whether one or multiple hops away), battery-powered devices wake up from sleep, send the data to their one-hop relay node and then quickly go back to a standby state. Between instances when the device is active and sending or receiving data, the radio can be off and operating in the realm of microamperes. For instance, the CC2652R device can sleep while retaining full random access memory (RAM) contents and consume only 0.9 µA.

This efficiency brings a significant advantage for devices that typically generate data triggered by sporadic alarm events (such as door and window sensors) or user actions (such as switches/key fobs, alarm panels or shade systems). With Zigbee's and Thread's asynchronous operations, devices can sleep most of the time, without waking up to maintain a synchronous connection for housekeeping reasons. Their whole current consumption application profile can be brought down to single-digit microamperes, even with packets emitted every 10 seconds.

With peak current levels around the single-digit microamperes, Zigbee and Thread enable devices in the home and building automation space to operate for years off of coin-cell batteries. The data communication is efficient because both Zigbee and Thread stacks run on top of the 802.15.4-link layer, which comprehends packet size and keeps over-the-air communication to a minimum. Thread leverages the IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) compression layer and link-layer forwarding. Zigbee was designed from the ground up, with binary optimization in the networking protocol for the underlying 802.15.4 frames. Zigbee uses next-hop relay to minimize routing information in packets.

The headers and networking management operations involved to maintain and establish routes are short and reliably enable a 20-byte application frame (for a lighting control command or an alarm event) in a single 802.15.4 packet instance of 50–80 bytes, with a turnaround time of a few tens of milliseconds per hop. In most systems, with four to five hops as the mesh branch's biggest length, this speed still provides less than 100 ms of latency for actuating device-to-device communication (where 100 ms matches the typical human response time for a button re-press).

Low-power operation and the ability of a network to scale are both key requirements in residential systems with tens of nodes interoperating, such as lights, environmental sensors and thermostats. All of these factors are an even bigger concern in commercial and industrial building automation systems, where the number of devices may reach hundreds or even thousands of nodes.

802.15.4 has a native channel arbitration mechanism [Carrier Sense Multiple Access-Collision Avoidance (CSMA-CA)], which enables sharing the air medium by sensing for interference and backing off exponentially if an interferer is detected on the same channel. The MAC layer in the 802.15.4 standard also provides for acknowledgment and retransmissions, maintaining reliable and efficient hop-to-hop communication.

An efficient implementation of the mesh fabric by the routing nodes in the network minimizes over-the-air traffic and broadcasts. The receiver in these nodes is always on (they are usually mains-powered, like a light bulb/fixture or a thermostat) and stores next hops to the final destination proactively (Thread) or actively (Zigbee), building a small and lean routing table. Packets are not relayed by flooding the network through broadcasts, which ultimately can hinder scalability.

Still, the reliability of packet communication is ensured by a one-hop acknowledgment on each segment. Only small intermittent broadcast messages are exchanged between the network's routing nodes, thus minimizing overall housekeeping traffic to maintain the mesh. Routing nodes in the network also have the important role of buffering the data for the downlink communication of their sleepy "children," which can be configured to extract packets efficiently depending on the downlink requirements (which in many cases are latency insensitive).

Both Zigbee and Thread technologies have been demonstrated successfully in large commercial deployments that reach hundreds of nodes within the same network. TI has deployed a **400-node Zigbee-based network**, and the technology can enable even larger networks depending on node density, amount of traffic generated and application profile.

Interoperability at scale

Thread and Zigbee technologies are driven by industry alliances with members such as silicon vendors; service providers (test labs, design houses, integrators); and companies building products such as lighting and lighting infrastructures, shades, fans, blinds, roof automation systems, door locks, thermostat and HVAC systems, and security and safety devices. Zigbee and Thread products are also present in major ecosystems offered by telecom providers, including home automation services.

With such a breadth of ecosystem and end-product categories comes the problem of being able to guarantee that such a diverse set of products is interoperable. Interoperability has incredible value for the end consumer in do-it-yourself (DIY) scenarios, but also for ecosystem owners and service providers who want to curate the user experience and rely on communication technologies to work seamlessly.

Comparing the interoperability paradigm in a large and diverse network to a single device-to-device scheme with a specific application in mind, the complexity of device interoperability grows exponentially. For example, the communication between a smartphone and a wearable device relies on two fundamental assumptions: that the smartphone app has a pre-defined knowledge of what services are linked to the specific embedded end product it is pairing to (and more so with that specific manufacturer); and that the smartphone platform can be easily upgraded with new services or new applications.

Embedded devices are naturally constrained in terms of resources. For instance, if a switch had to download a new embedded app every time the user purchased a new light, the system would not be able to scale enough because of these embedded constraints. The enhanced value of device automation relies on these embedded devices to autonomously make instantaneous or predictive decisions, while collaborating with the cloud infrastructure to bring convenience of use. This means that the underlying application framework powering those devices must be flexible and adaptive in order to scale to support any device type and brand.

Once again, the challenge here is interoperability at scale: different categories of products, different product manufacturers and different sets of applications. Zigbee and Thread solved this problem by tackling interoperability at different levels.

Both standards define a common set of core procedures for networking operations, including the way networks are created; the way devices are joined and participate in the mesh fabric routing packets; procedures for network maintenance; security and device bootstrapping in the network through a user interface. Products can be securely commissioned and have a common way to be initiated and function through the core networking operations of their respective protocols, even if designed by different manufacturers.

Zigbee Pro and Thread technologies are available today through widespread stack component sets from different stack vendors and silicon providers. Zigbee has more than 20 Zigbee Pro-compliant certified platforms implementing Zigbee Pro from 11 different technology providers. Thread has more than 10 certified components from seven different platform vendors. In addition to that, Thread also has an open-source implementation available through the OpenThread project, led by Nest with participation from major silicon vendors. This open-source platform constitutes the basis for a mesh networking technology that is interoperable across different hardware implementations.

Interoperability is guaranteed at the networking level through core stack certification. Both the Zigbee Alliance and Thread Group run core stack certification programs built on top of a rigorous test plan with coverage of hundreds of test cases. These industry alliances also provide a set of tools and test harnesses to their members to validate implementations in-house. Certification tests are performed at authorized test labs with facilities around the globe. They act as third-party authorities who validate implementation by testing different vendor implementations.

TI, both a promoter member of Zigbee and an active contributor to Thread, has Zigbee and Thread-certified stack solutions and has been actively participating in the OpenThread community since its start in May 2016.

Embedded devices need to have a common language to discover each other, find services, transfer application data (temperature measurements, alarms, grouping, dimming commands) and create application linkages so that a specific switch knows which lights it can talk to.

Zigbee has successfully solved this problem with the concept of the Zigbee Cluster Library (ZCL). This is the application layer for the whole Zigbee protocol, running on top of the Zigbee Pro mesh network.

Beyond the core foundational layers, which provide the underling pipe for the mesh operations in the network, the ZCL builds an application framework that ensures that the application space is uniform across device types that claim to implement a specific function. The ZCL is similar to the concept of objects in object-oriented programming, with commands and attributes analogous to function members and variables of a class[1]. Similarly, physical products like lights can be a combination of imported objects (a light that can be on/off and/or dimmable).

Exchanging and agreeing on a set of clusters (or objects) in a client/server relationship model is what matches functions between products. As objects are defined in the ZCL standard and are therefore the common denominator across products, no matter what the light functionality is (a light that can be simply turned on/off, or dimmed and have color control capability), the corresponding device intended to operate with it can choose to simply pair with one, multiple or all functionalities implemented by the lighting device (e.g., it can be a simple on/off switch or a color light controller).

Discovering and matching services is controlled through a specific management service cluster called a “Zigbee device profile,” which serves as common object that encompasses application and networking housekeeping operations.

The standardization of the ZCL is the result of more than 10 years of effort. It is a mature library set of commands and states enabling ~50 different device types and has been optimized to run on the underlying mesh fabric delivered by Zigbee Pro.

The Zigbee product certification program run by the Zigbee Alliance validates the functionality of products implementing a specific set from the ZCL, according to the functionality of the device. Manufacturers can design products by picking the desired cluster set (much like importing a class in object-oriented programming) and validate the conformance to their product standard by validating those clusters at an authorized test lab. Like the Zigbee Compliant Platform certification program, the Zigbee Alliance provides a Zigbee test tool and access to a test-bed for their members to prepare for product certification tests at labs. With the **SimpleLink CC26x2 software development kit (SDK)**, TI offers a complete solution including the ZCL, which has been validated against the Zigbee test tool, to jump-start the development of interoperable products.

How does Thread solve the embedded application fragmentation problem? Thread offers an IP-based low-power core networking standard for mesh operation, and it does not specifically define an application layer. On the other hand, being an IP-based standard, it benefits from existing decades of industry efforts in the IP domain for the standardization of embedded devices in application layering, like the Constrained Application Protocol (CoAP).

Thanks to this IP convergence, Thread networking technology can easily scale and support different consortia, which aim to unify the application objects [Open Connectivity Foundation (OCF) IoTivity, Open Mobile Alliance (OMA) Lightweight Machine-to-Machine (LWM2M), Energy Efficiency Bus (EEBus)], or industry-leading initiatives like OpenWeave from Nest that rely on open-sourcing of the application layer and objects.

Among all available application layers, Dotdot from the Zigbee Alliance offers one of the most promising opportunities for rapid convergence. Dotdot reuses the library object defined from the ZCL, and the defined set of data and functionality models ported over from a binary implementation on top of Zigbee Pro to run on top of an IP-based network like Thread using CoAP as the application layer (see **Figure 2**). Just as HTTP applications on high-end devices run on top of IP-based networks, CoAP is the application link for embedded devices to transport cluster-based objects in a RESTful model.

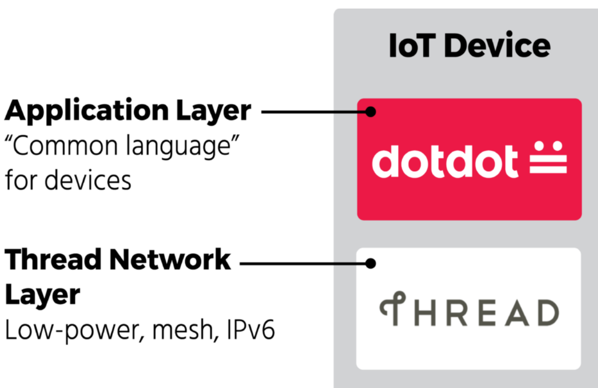

Figure 2: *Thread and Dotdot application layer.*
Image source: Zigbee Alliance (www.zigbee.org)

Being mindful of resource-constrained devices, power consumption targets and the requirements of network scalability, Dotdot uses an efficient combination of CoAP (which can run on top of a connectionless IP network like Thread) and Concise Binary Object Representation (CBOR) to keep data communication, from an application standpoint, small and effective.

Whether the core mesh networking layer is Zigbee Pro based or Thread based; whether the ZCL model from the Zigbee Alliance is in its native binary form or its translation over IP through Dotdot for Thread, a solution for the automation of devices and end-to-end application interoperability is available on 802.15.4-based silicon, like the newly launched CC2652R wireless MCU.

Zigbee vs. Thread

In addition to the differences between Zigbee and Thread in how they authenticate products and establish routes, there are other slight variations between the two standards related to their mesh foundation. Zigbee core networking supports a centralized and distributed (touchlink) coordination. In the centralized approach, Zigbee uses a statically allocated coordinator in the network, while Thread implements this functionality in the network leader, which is autonomously elected and can change dynamically. The latter implies a more autonomous balancing of network resources and improved self-healing capability, in contrast with the more authoritative and centralized approach in Zigbee core networks.

Zigbee is a transport-oriented networking protocol, where logical links between endpoints are established before any application data transfer and retransmissions and acknowledgments are provided natively. Thread uses the User Datagram Protocol (UDP), which is a connectionless transport protocol, and as such must rely on application layers like CoAP to cope with un-sequenced packets and retransmissions.

However, the main difference between the two standards comes with native support for IP. All Thread nodes have one or multiple IPv6 addresses, while Zigbee nodes have a binary 16-bit address that must be translated into the external IP world.

This comes with implications for application design and system deployment. First, when communicating with cloud-based applications, Zigbee devices must go through a gateway that essentially converts IP addresses to Zigbee addresses. Thread devices still need to go through an 802.15.4-based bridge, which routes packets between that interface and another IP one (like Ethernet, cellular or Wi-Fi), which in Thread terminology is called a border router.

While even in case of Thread, the border router case must take care of some addressing and function translation from cloud to local network (potentially the IPv6-to-IPv4 conversion, as well as being the proxy for the authentication of Thread devices), the design of such border router can leverage existing libraries and technologies [like network address translation (NAT)64, or multicast main domain system (mDNS)].

Also, as Thread networks are inherently IP-based, they bring the foundation for a unified application and security model between cloud applications and devices in the Thread network. When IP integration becomes a primary requirement (as is the case for communication with building management systems and automation in office spaces managed by a systems administrator), Thread-based networks offer an advantage.

Natively supporting IP traffic has the advantages I mentioned earlier, but at the same time application packets conveyed over IP may be bigger with respect to tailor-made binary-based frames used in Zigbee. This implies more efficient and power-optimized operations, with reduced latency for nodes in a Zigbee network.

The Zigbee Pro networking layer, alongside the ZCL, has been successfully adopted and produced in more than 100 million products since its inception, with several revisions of the standards for both the core mesh networking functionality and application layers. When time to market, latency, integration with existing device types and power-constrained operations (like energy-harvesting switches controlling and dimming light bulbs) become the primary factor for technology selection, as in the case of lighting for residential home automation, Zigbee is a very compelling solution.

Table 2 lists the differences between the two core mesh networking standards and the implications for technology adopters.

Although there is some overlap in the target application space and the potential for

Functionality	Zigbee	Thread
Authentication at joining	Centralized via the trust center with optional out-of-band device-based install code, or distributed with proximity pairing	Smartphone-based, with device-specific quick response (QR) code scanning
Security	Advanced Encryption Standard (AES)-128 network-level with key transported from joinee to joining device Optional application-level key	AES-128 MAC level derived from an elliptic curve cryptography (ECC)-based password juggling scheme and DTLS session establishment
Device bootstrapping and commissioning	Button-press easy mode or proximity-based (touchlink)	Smartphone-based, with device-specific QR code scanning
Network and mesh management	Typically done in the Zigbee coordinator in centralized network, or distributed in the touchlink case	Dynamic leadership
Self-healing	Native router and mesh self-healing	Routers and leader self-election and self-healing
Cloud integration	Zigbee gateway	Thread border router
Power performance for application packets	Optimum	Very good
Latency performance for application packets	Best	Very good
IP native integration	No	Yes
Standard longevity	First revision in 2005	First revision in 2015
Industry forum breadth and size	~400 companies	~270 companies

Table 2: *Zigbee and Thread comparison.*

cannibalization of the two technologies for the same class of devices, Zigbee and Thread share many core fundamentals and are based on the same 802.15.4 radio. Product manufacturers can start today with a single hardware design, with the flexibility of seamlessly using one or another through a software upgrade.

Conclusion

This paper analyzed the benefits that Zigbee and Thread technologies bring to the residential and building automation space and their relevant applications, focusing on their ability to bring low-power performance and device-to-device interoperability at scale.

Although there are differences, both Zigbee and Thread can be implemented as different software programs on the same hardware silicon, as with the TI SimpleLink Multi-Standard CC2652R wireless MCU, which features Zigbee and Thread stacks within the CC26x2 SimpleLink SDK offering. TI created the CC2652R wireless MCU for 2.4-GHz operation, which can be used to run Bluetooth 5-, Zigbee- or Thread-based applications. Additionally, the SimpleLink Multi-Band CC1352R wireless MCU supports multiple standards and technologies in the 2.4-GHz space (like Bluetooth 5, Zigbee and Thread) and Sub-1 GHz, bringing down the overall cost with ultimate flexibility for designers to load the software of their choice, depending on their requirements.

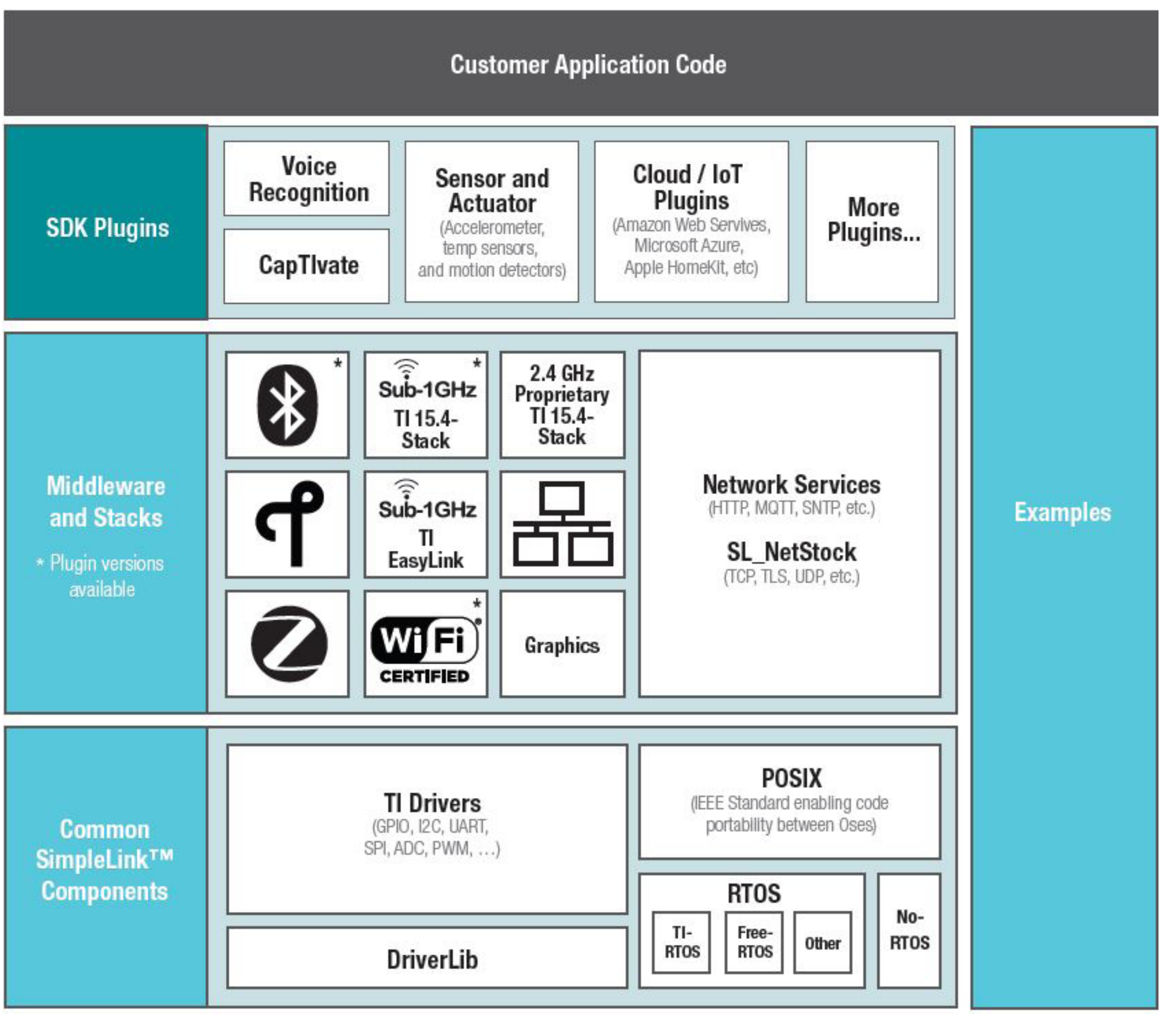

Figure 3. SimpleLink SDK diagram.

The SimpleLink CC26x2 and CC13x2 SDKs feature a common firmware upgrade model between all existing protocol software (Zigbee, Thread, Bluetooth 5 and Sub-1 GHz) that can switch seamlessly from one to another between protocols at the time of manufacture or when the product is in the field. Because software upgrades are natively supported, you can start today and upgrade with protocol combinations in dynamic operation mode (concurrent) that will be offered throughout 2018.

TI also has gone one step further by providing a common software development framework that can be used for wired and wireless embedded devices belonging to the SimpleLink family, thus enabling modularity and application portability. Code developed for an application can be reused when migrating from one communication technology to another, or when adding on top of it.

TI offers a one-stop shop for connected embedded products and a unified software development environment with the SimpleLink MCU platform, which ultimately brings down engineering costs, reduces risks in execution and accelerates time to market—all crucial elements for product innovation and differentiation in the fast-paced Internet of Things (IoT) market for home and building automation. See **Figure 3** on the previous page for a diagram of the SimpleLink MCU and SDK structure.

References

1. Gislason, Drew. "Zigbee Wireless Networking." Newnes (Boston, Massachusetts): 2008.
2. Zigbee Alliance. "The Zigbee Alliance and Thread Group Address IoT Industry Fragmentation with the Availability of the Dotdot Specification over Thread's IP Network." http://www.zigbee.org. 2017, December 12.

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES