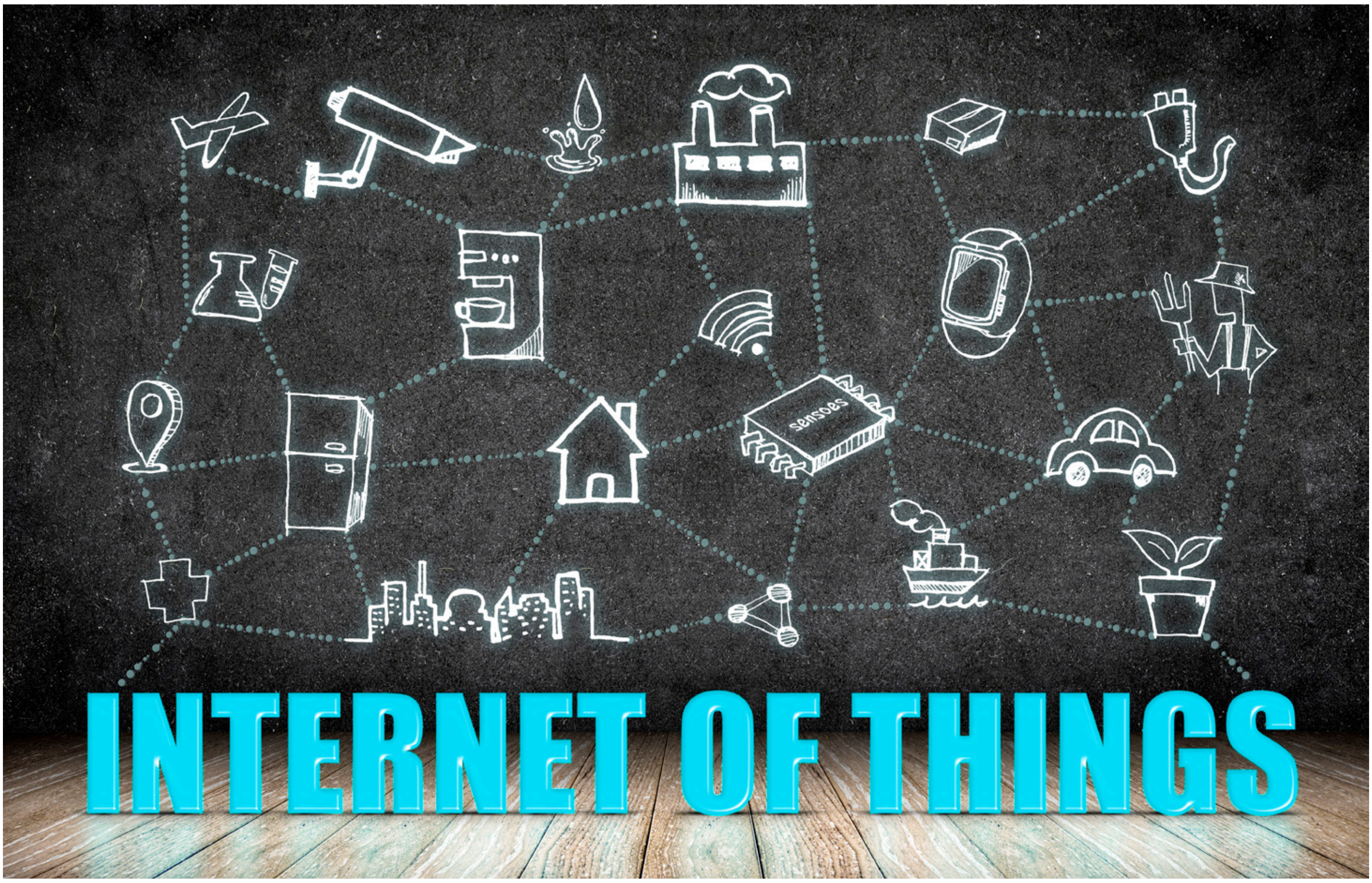
The Origin of IoT OSes

Research and thoughts on the Internet of Things Operating Systems

By **Allan He**, Embedded Systems Specialist



Most IoT OSes originate from embedded OSes, whose kernels most likely come from RTOSes. Of course, there are IoT OSes that are original, like Android Things, Microsoft Windows 10 IoT Core, and others. Here we present a „A Brief History of RTOS“!

The history of RTOS dates back four decades ago to 1970s in North America. At the peak, there were as many as a few hundred different RTOSes. Even now there are dozens of RTOSes, with a few in China.

Figure 1 lists a dozen RTOSes that are still active today (meaning there are companies that service and support them), ranked by their original release time. Many of them have changed hands before, for example, Wind River was acquired by Intel in 2009 [1]; however, Wind River has kept all its products. There are also cases where a product is not available anymore, like VRTX and pSoS (acquired by Wind River). Zephyr and Nuttx are among the newer open-source embedded OSes; they both fall into the category of RTOS.

It is worth noting that there are a few good RTOSes in China as well; these include Mr. Puxiang Xiong’s RT-Thread and SylixOS that is more recent and is active in military projects.

The evolution of embedded OSes

Embedded OSes (also called device OSes) have gone through several stages

RTOS	Company / Author	Website	Status
VRTX	Ready System/Microtec		Acquired by Mentor Graphics
pSoS	ISI		Acquired by Wind River
OS-9	Microware	http://www.microware.com/	Acquired by Metrowork
SMX	Micro Digital	http://www.smxrtos.com/	
VxWorks	Wind River	https://www.windriver.com/products/vxworks/	Acquired by Intel39
LynxOS	Lynx Software	http://www.lynx.com/products/real-time-operating-systems/lynxos-rtos/	
QNX	QNX	https://blackberry.qnx.com/en	Acquired by Blackberry
CMX	CMX System	http://www.cmx.com/rtos.htm	
ThreadX	Express Logic	https://rtos.com/solutions/threadx/real-time-operating-system/	Microsoft acquired Express Logic in April 2019.
µC/OS	Micrium	https://www.micrium.com/	Acquired by Silicon Labs
Integrity	Green Hills Software	https://ghs.com/products/rtos/integrity.html	
OSE	Enea	https://www.enea.com/products/operating-systems/enea-ose/	
Zephyr	The Linux Foundation	https://www.zephyrproject.org/	
Nuttx	Gregory Nutt (2007, BSD licensed)	http://www.nuttx.org/	Used in drones

Figure 1: A list of active RTOSes.

Time	Revolutionary Development	Signature Product(s)
1980s	Tools and RTOSes	RTOS kernels
1990s	IDE and OS platform	RTOSes
2000s	Open-source platform	Linux
2010s	Application solution	Android
2020s	Security and connectivity	IoT OSes ppeger

Figure 2: The evolution of embedded OSes.

of evolutions since the 1980s; the industry sees a stage of development roughly each decade. **Figure 2** shows the stages of embedded OSes. In every stage, there will be a few landmark OSes. The 1980s saw RTOS kernels like µc/OS and FreeRTOS. In the 1990s, VxWorks, which integrates an RTOS kernel with file system, networking, shell, and IDE, was very popular partly due to its ease of use.

In the 2000s, general-purpose OSes like Linux entered the embedded system area. Many of them are open-source, but there are also proprietary OSes like Windows (e.g. Windows 10 IoT Core). After 2010, Android became the representative of mobile OSes. The earliest design of Android came from Andy Rubin, which originated from the existing idea of using Linux on mobile phones. There have been many Linux phones before Android; they ran on OSes made by MontaVista and other companies. It wasn't easy for these companies to make Linux phones, and the same held true for Android in its early days. However, Android managed to become the top mobile phone OS with more than 80% market share.

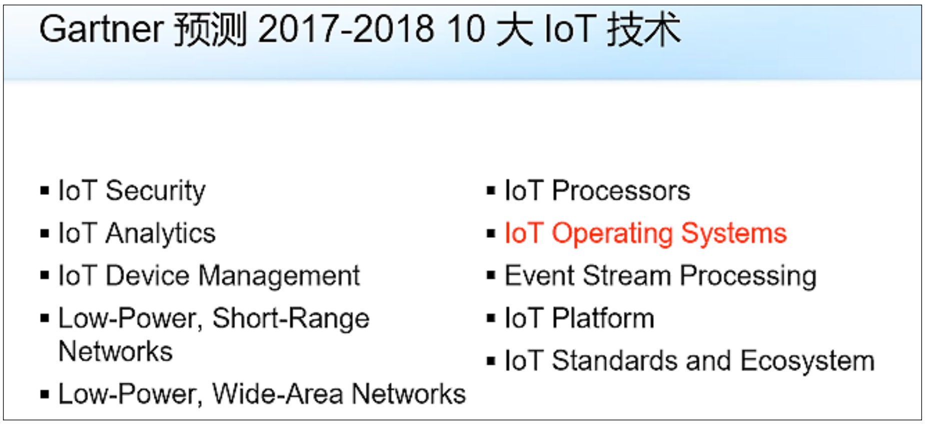
The newest members in the evolution of embedded OS are IoT OSes. By 2020, we might be able to see a few popular and widely-adopted IoT OSes.

Why do we need IoT OSes?

Both traditional embedded OSes and general-purpose OSes cannot satisfy the requirements of IoT due to its complexity. End-to-end complete solutions will be much better suited.

In 2016, WindRiver pointed out in a keynote session that there are 8 major requirements from IoT devices: a modulated and upgradeable architecture, flexibility on device software of different grades, device security and safety, virtualization, performance and reliability, connectivity, a feature-rich UI, and that it should pass industrial certifications. These requirements mean a new type of OS, or a significantly modified current OS, is needed by IoT.

Most of the current market offerings can partially satisfy these 8 requirements. While traditional embedded OSes cannot provide modulated upgradeable architectures, many OS products in China have not passed internationally recognized certifications and thus cannot be used in many industrial setups. As for UI, only a small number of OSes offer a good one, like RT-Thread.



Gartner 预测 2017-2018 10 大 IoT 技术

- IoT Security
- IoT Analytics
- IoT Device Management
- Low-Power, Short-Range Networks
- Low-Power, Wide-Area Networks
- IoT Processors
- IoT Operating Systems
- Event Stream Processing
- IoT Platform
- IoT Standards and Ecosystem

Figure 3: Top IoT Technologies as analyzed by Gartner. [2]

The concept of IoT OS first emerged in 2014. But it was not until 2016/2017 when the concept started to receive wide attention. According to the industry analyst firm Gartner, the top 10 IoT technologies for 2017-2018 include IoT OSes. This indicates that the industry does hold a consensus that IoT OSes need to be worked on.

The origin and development of IoT OS

The concept of IoT OS started from two sensor network open-source OSes, TinyOS and Contiki; today's IoT shares quite some similarities with a sensor network. The Contiki project is still very active today. Contiki's author is Dr. Dam Dunkels who used to work in the computer science department of KTH Royal Institute of Technology in Sweden; he also founded Thingsquare and is the author and maintainer of the LWIP/uIP project. Universities in Europe still offer classes that use Contiki. TinyOS is a project of the University of California Berkeley; it is not being maintained anymore.

In 2010, RIOT, an OS that is designed for IoT, emerged in Europe; unfortunately, it has not impacted the market much.

In 2014, the area of IoT OS became much more lively. *Wired* journal and IEEE's *Spectrum* both reported on IoT OSes in this year. The question is, why was 2014 the year? The answer lies partly in the fact that there were two major companies released their IoT offerings: Microsoft's Windows 10 IoT Core and ARM's mbed OS. In October 2014, a handful of smaller companies also announced their products, like Micrium's Spectrum (based on µC/OS) and MxChip's MiCO OS. There are many other products that simply add some functional modules and cloud connectivity (e.g. AWS and Azure) to existing RTOSes and present themselves as new IoT software solutions.

In 2015 Huawei released LiteOS but with a limited initial splash. Brillo OS, which Google announced in its 2015 I/O event, was the major announcement of the year. Brillo OS can be traced back to Nest that Google acquired shortly before the announcement. Google has since renamed Brillo OS to Android Things. With these developments, the industry started to think IoT OS as the next "thing" and many companies followed suit: Alibaba announced YunOS [3] (later

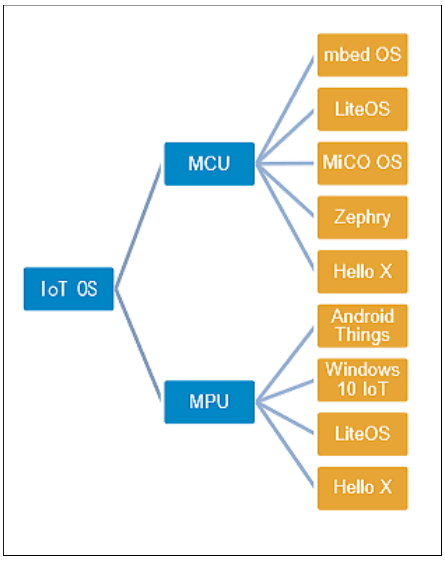


Figure 4: Categorization of IoT OSes.

renamed to AliOS) for IoT at its cloud conference in October 2017, the Linux Foundation released Zephyr in 2016, and Haier showcased its UIhome OS-based refrigerator (which has a big screen on it) with a big board saying "Your Home OS" at CES 2017 in the USA.

Different types of IoT OSes

IoT OSes cover a wide area from network gateways, servers to endpoints. Among the building blocks of IoT, embedded OS claims several parts, open-source Linux covers connectivity nodes, and the rest goes to Android and iOS (smartphones). According to the 2018 IoT Developers Survey [4], Linux claimed more than 71.8% of the market share; there were a few noteworthy smaller OSes as well. We also have products that do not use an OS at all. As most IoT devices will be running on an MCU that cannot run a full-blown Linux OS, there can be three times as many devices that run RTOSes compared to Linux. In summary, the world of IoT has many different types of OSes.

What is an IoT OS?

It is hard to find a formal definition of IoT OS. In Elsevier press's call for paper for an IoT special issue of *Next Generation Computing System* journal, it laid out key characteristics of an IoT OS on protocol design, verification, modulation, power consumption, scheduling (based on power consumption), hardware support, architecture, networking, protocol stack, dependability, interoperability, API, real-time characteristic, etc.

While Microsoft advertises its product as "The operating system built for the Internet of Things" [5], Google describes the mission of Android Things as "Build connected devices for a wide variety of consumer, retail and industrial applications" [6]. ARM, on the other hand, thinks an IoT OS should be open-source and designed for embedded systems ("things") — to them, "things" typically have ARM Cortex-M inside.

An IoT OS generally should provide a real-time-enabled, low-power and secure platform that offers sensing capacity, connectivity, and cloud management ability. While the platform is built on technologies
related to real-time computing, power management, and security, it needs an end-to-end solution to connect the cloud to a device.

The *status quo* of IoT OS

While big corporations are already actively executing their plans, smaller companies are mostly trying out their strategies. Technologies and products are certainly easier to work out than a mature business model.
There are two major types of IoT OSes. At one side we have OSes that are designed for IoT and do not exist before, like mbed OS, MiCO OS, and Android Things. **Figure 4** shows the categorization of IoT OSes; they can be divided according to their support of MCU and MPU.
At the other side, we have Oses that are based on embedded OSes; Linux and Android are the two biggest players in this category. After modification and strengthening, FreeRTOS can also be used in IoT applications; Amazon has recently released

its own modified version of FreeRTOS. Similarly, µC/OS and ThreadX can also be used; Renesas ARM MCU platform Synergy is based on ThreadX. Some variants of VxWorks, Nucleus and RT-Thread 3.0 are suitable as IoT OSes as well.

There are more than a dozen IoT OSes; unfortunately, they are all at their initial stages. ARM mbed OS has released 1.0, 2.0, 3.0, and 5.0 versions; however, the OS is very much still in an exploration phase with changes happening all the time. If you browse the official Chinese website of mbed, you can still find a note mentioning that mbed OS is under active development.
MxChip's MiCO OS has changed its marketing position. Now it is being advertised as companion software of MiCO's WiFi modules. The reason is that, as the SDK does not support smart hardware other than MxChip's own, it cannot be called an IoT OS. It is a basic fact that an OS should work with multiple hardware platforms.
Microsoft's Windows 10 IoT Core is making its way to the cloud. After attending a few conferences, I personally feel that Microsoft focuses more on the cloud management, i.e. how can IoT devices be managed from the cloud, rather than the role/functionality of the OS itself.

AliOS has been seeking to make a formal entrance into the IoT market; it has formed partnerships with NXP and ST.
After 2015, MCU vendors have not interacted much with IoT OSes. ST has not said much on mbed support, but have recently indicated that it wants to partner with AliOS. Huawei LiteOS has organized hackathons, but it currently has a small user/partner base and only one simple strategy to develop on NB-IoT.
We seldom hear companies in China or abroad advertising their partnerships with IoT OSes (we are referring to native new IoT OSes like mbed, rather than embedded OSes that have already accumulated users, e.g. RT-Thread). Traditional RTOSes and Linux are still mainstream among IoT devices. Admittedly, getting accepted by users is a lengthy process. Once we went through the process, we would have a positive feedback loop around these new OSes that advance them further.

Conclusions

First, any OS, including an IoT OS, needs a long time to popularize and adapt to the market. While users would not make up their minds in a short period of time, they are also unlikely to switch once decided. In other words, the loyalty of the user base is often quite high when it comes to OSes. In fact, it is more possible and easier for a user to switch between different chips.

Second, the characteristics and appearances of IoT OSes are not that clear. There is a trend where many companies want to go large on their products. For smaller competitors, this means they should rank feature works from small to big, and from simple to complicated; attempting to accomplish everything in one go would not be a wise choice. For example, according to RT-Thread's founder Puxiang Xiong, they have not yet to work on security features. As security is a completely different area, it would be smart to accumulate more technical expertise before attempting. Before that happens, external security technologies and products can be used.

Third, resources should be dedicated to security researches on IoT OSes. Security is an essential part of IoT; a product with security solutions can be more attractive to the users. For instance, FreeRTOS has security features built in; they certainly have added more value to the OS.

Fourth, higher education institutions should invest more in IoT OS research and education. In 2017 I introduced IoT OSes in a few IoT education conferences, but with limited impact. We are now in a period where it is ideal to innovate on the existing traditional embedded system OS courses. Some schools have already started to consider RTOS classes. Institutions and companies that work closely with them (higher education projects) should actively seek to introduce IoT OSes to their curriculums; this is necessary to improve the influence of these OSes.

Last but not least, there are two points which companies should consider to advance the usage of IoT OSes:

1. they should carefully draw a boundary for IoT OSes; obviously, IoT OSes cannot be used anywhere.

2. persistence is the key to success. There are many companies that have previously been active in open-source or ecosystem, and then quit because they are unsatisfied about the slow return of investment; in many cases, they simply go back to please their major customers.

As applications of IoT grow and mature, IoT OSes are starting to make an entry into the industry. However, patience is needed to see them become more mature. ⏮

191125-01

Editor's Note: This article is an extract from the book *Embedded Operating Systems*, available shortly from the Elektor Store, www.elektor.com.

Notes

[1] Wind River was bought by TPG from Intel in 2018.

[2] Summarized on Argus Technologies blog, http://www.tekargus.com/blog/top-10-internet-of-things-technologies-2017-and-2018/.

[3] YunOS is not specifically designed for IoT. It was initially released for smartphone, and later it started to target automobile.

[4] Eclipse Foundation, Inc. 2018 IoT Developer Survey. April 2018. Available on https://iot.eclipse.org/resources/iot-developer-survey/iot-developer-survey-2018.pdf.

[5] According to the page summary on https://developer.microsoft.com/en-us/windows/iot.

[6] While Google no longer write this on its website, it is quoted on https://medium.com/@iskerrett/microsoft-and-googles-iot-strategy-tale-of-two-conferences-3d1b9d28ebe5.