iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** IX **Month of publication:** September 2017

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887
Volume 5 Issue IX, September 2017- Available at www.ijraset.com

A Comparative Study and Analysis of Operating System for the Internet of Things

Mayurbhai p. Zezariya
Computer Science Department, C.U. Shah University, Wadhwan, Gujarat, India

Abstract: ***We describe The Internet of Things (IoT) as a network of physical objects or "things" embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data in real time with the outside world. It therefore assumes an operating system (OS) which is considered as an unavoidable point for good communication between all devices "objects". For this purpose, this paper presents a comparative study between the popular known operating systems for internet of things. In a first step we will define in detail the advantages and disadvantages of each one, then another part of Interpretation is developed, in order to analyze the specific requirements that an OS should satisfy to be used and determine the most appropriate .This work will solve the problem of choice of operating system suitable for the Internet of things in order to incorporate it within our research team.***
Keywords: Internet of things, network, physical object, sensors, operating system.

I. INTRODUCTION

The Internet of Things (IoT) is the vision of interconnecting objects, users and entities "objects". Much, if not most, of the billions of intelligent devices on the Internet will be embedded systems equipped with an Operating Systems (OS) which is a system programs that manage computer resources whether tangible resources (like memory, storage, network, input/output etc.) or intangible resources (like running other computer programs as processes, providing logical ports for different network connections etc.), So it is the most important program that runs on a computer.

Computer operating systems perform basic tasks, such as recognizing input from the keyboard, sending Output to the display screen, keeping track of files and directories on the disk, and controlling peripheral devices such as printers. For large systems, the operating system has even greater responsibilities and Powers. It is like a traffic cop it makes sure that different programs and users running at the same time do not interfere with each other. The main objective of this work is to solve it in a definitive way. This work contains a detailed study of the existing operating systems for the Internet of things, and this through an evaluation of each one with its advantages and disadvantages, and the final part will be devoted to defining and examine the appropriate Operating system to be used in the Internet of Things.

II. OPERATING SYSTEMS

The set of operating systems designed for the internet of things are presented under the architecture shown in Figure below (Figure1) : A hardware layer and drivers, a second layer that contains Kernel / scheduler and network stack, The choice of Kernel is to have a complete control over everything that occurs in the system and the choice of the scheduling strategy is tightly bound to real-time and different task priorities support .Finally the last layer is for applications or user interface for supporting degree of user interaction.

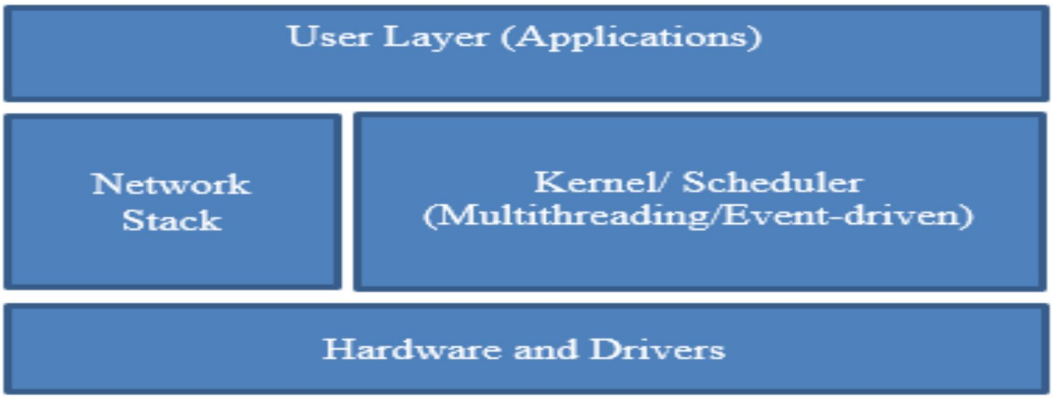


Figure 1: architecture of operating systems

©IJRASET (UGC Approved Journal): All Rights are Reserved

A. TinyOS

TinyOS is an open source, BSDlicensed operating system designed for low power wireless devices, such as those used in sensor networks, ubiquitous computing, personal area networks, smart buildings, and smart meters.

1) *Architecture:* TinyOS follow a monolithic architecture based on a combination of components, reducing the size of the code needed to set it up. This is in line with the constraints of Memories that are observed by sensor networks; however, the TinyOS component library is particularly complete since it includes network protocols, sensor drivers and acquisition tools of data. All of these components can be used as they are, they can also be adapted to a precise application.
2) *Programming Model:* The previous versions of TinyOS do not provide multithreading support; building applications were based on the event driven programming model. TinyOS version 2.1 supports multithreading. And these TinyOS threads are called TOS Threads.
3) *Advantages:* TinyOS is a system mainly developed and Supported by the American University of Berkeley, which offers it in download Under the BSD license and monitors it. Thus, all the sources are Available for many physical targets. Based on event driven operation, TinyOS offers to the user a precise management of the sensors Consumption and makes it possible to adapt better to the Random nature of wireless communication between physical interfaces. The preemptive nature of an operating system specifies if this one allows the interruption of a task in progress. TinyOS does not manage this Preemption between tasks but gives priority to hardware interruptions.
4) *Disadvantages:* TinyOS does not support real-time application operations. It provides a process planning algorithm based on priorities, but once a process is scheduled to run to completion. This may result in a missed deadline for a high priority process that enters the Preparation queue once a low priority process has been scheduled.

B. Contiki

Contiki is an open source operating systems for sensor nodes. It was developed at the Swedish Institute of Computer Science by Dunkels et al., it is a lightweight open source OS written in C for WSN sensor nodes Contiki connects tiny low-cost, low power microcontrollers to the Internet. Contiki is a highly portable OS and it is build around an event driven kernel. Contiki provides preemptive multitasking that can be used at the individual process level. A typical Contiki configuration consumes 2 kilobytes of RAM and 40 kilobytes of ROM. A full Contiki installation includes features like : multitasking kernel, preemptive multithreading, protothreads, TCP/IP networking, IPv6, a Graphical User Interface, a web browser, a personal web server, a simple telnet client, a screensaver, and virtual network computing.

1) *Advantages:* Contiki provides powerful low power Internet communication. Contiki supports fully standard IPv6 and IPv4, along with the recent low power wireless standards: 6lowpan, RPL, CoAP. With ContikiMAC and sleepy routers, even wireless routers can be battery operated. Contiki runs on a range of low power wireless devices, many of which can be easily purchased online. Contiki is open source software: it can be freely used both in commercial and noncommercial systems and the full source code is available.
2) *Disadvantages:* Contiki is an event driven OS, therefore it does not employ any sophisticated scheduling algorithm. Events are fired to the target application as they arrive. Contiki does not support deployment of real time applications, so there is no implementation of any real time process scheduling algorithm in this OS.

C. Nano-RK

NanoRK is a fully preemptive reservation based real time operating system (RTOS) from Carnegie Mellon University with multichip networking support for use in wireless sensor networks. NanoRK currently runs on the Firefly Sensor Networking Platform as well as the MicaZ motes. It includes a lightweight embedded resource kernel (RK) with rich functionality and timing support using less than 2KB of RAM and 18KB of ROM.

1) *Advantages:* One of the goals for NanoRK was to facilitate application developers by allowing them to work in a familiar multitasking paradigm. This results in a short learning curve, rapid application development, and improved productivity.
2) *Disadvantages:* NanoRK only provides support for static memory management; it does not support dynamic memory management. In NanoRK, both the OS and applications reside in a single address space and to the best of authors' knowledge NanoRK does not provide any support to safeguard collocated OS and process address spaces.

©IJRASET (UGC Approved Journal): All Rights are Reserved

D. Free RTOS

"Free RTOS is a market leading RTOS from Real Time Engineers Ltd. that supports 35 architectures and received more than 113000 download during 2014. It is professionally developed, strictly quality controlled, robust, supported, and free to embed in commercial products without any requirement to expose your proprietary source code. Free RTOS has become the de facto standard RTOS for microcontrollers by removing common objections to using free software, and in so doing, providing a truly compelling free software model."

1) Advantages

Free RTOS is downloaded every 260 seconds (on average).
Free RTOS offers lower project risks and a lower total cost of ownership than commercial
Alternatives because:
It is fully supported and documented.
Free RTOS has a tickles mode to directly support low power applications.
Free RTOS was downloaded >113000 times in 2014.
Free RTOS is designed to be simple and easy to use: Only 3 source files that are common to all RTOS ports, and one microcontroller specific source file are required, and its API is designed to be simple and intuitive.

2) *Disadvantages:* There are only limited tasks run at the same time and the concentration of these systems are on few applications to avoid errors and other task have to wait. Sometime there is no time limit of how much the waiting tasks have to wait. Multitasking is done few of times and this is the main disadvantage of Free RTOS because this system runs few tasks and stay focused on them. So it is not best for systems which use lot of multithreading Because of poor thread priority. Free RTOS used lot of system resources which is not as good and is also expensive.

E. RIOT

RIOT or the friendly Operating System for the Internet of Things is a lightweight operating system for networked systems with memory constraints, focused on devices with low power consumption for Internet of things. It is free software, released under the GNU General Public License (LGPL), It was originally developed by the Free University of Berlin, the National Institute for Research in Computer Science and Automation (INRIA) and the University of Applied Sciences of Hamburg (HAW Hamburg). The core of RIOT is largely inherited from FireKernel1, which was originally developed for sensor networks.

1) *Advantages:* RIOT an operating system designed for the particular requirements of Internet of Things (IoT) scenarios. These requirements comprise a low memory footprint, high energy efficiency, real time capabilities, a modular and configurable communication stack, and support for a wide range of low power devices.

2) *Network Stack:* Supports the entire network stack of IoT (802.15.4 Zigbee, 6LoWPAN, ICMP6, Ipv6, RPL, CoAP, etc), both static and dynamic memory allocation, POSIX compliant (partial), All output can be seen in the terminal if hardware is not available.

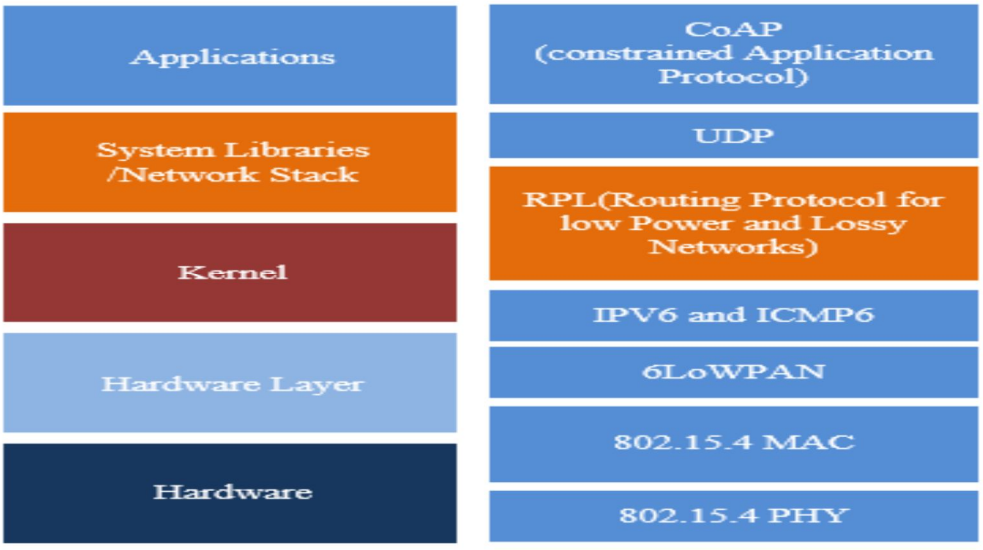


Figure 2: Architecture and Driver support for RIOT

©IJRASET (UGC Approved Journal): All Rights are Reserved

OS	License	Programming Model	Real-time support	Technologies supported
TinyOS	Open Source BSD	Event Drive,thread	No	Broadcast based Routing
Contiki	Open Source BSD	Protothreads and events	Partial	6lowpan, RPL
NanoRK	Open Source	Multithreading	Yes	RTLink,U-Connect
LiteOS	Open Source	Threads and Events	Partial	JTAG
Free RTOS	Open Source	Multithreading	Yes	TCP,UDP, Ethernet, TLS
RIOT	Open Source LGPL v2.1	Multithreading	Yes	6LoWPAN, RPL,CoAP, UDP,TCP, CBOR, CCN-lite, OpenWSN, UBJSON

Table 1: Comparison of OS's

III. COMPARISON OF IOT OS'S

As we can see in the following table (table1), the RIOT system supports the most communication technologies over other systems, supports multithreading and real time media with a GNU GPL license, which allows the user several rights on a computer program.

A. The freedom to run the software for any purpose.
B. The freedom to study the functioning of a program and to adapt it to its needs, which requires access to source codes.
C. Freedom to redistribute copies.
D. The obligation to provide the community with modified versions.

Since Our goal is to discover and compare in a detailed way the characteristics of each operating system in order to detect the most suitable for the Internet of things. For this we have based on the characteristics detailed in section2: OSs to conclude that the RIOT operating system is the most appropriate especially in our search axis since it is free to be run in any domain, adaptable To the needs of the user with simple access to source codes, it aims exactly to fulfill the requirements to should satisfy an operating system for internet of things.

IV. REQUIREMENTS OF AN OS FOR IOT

In summary the main requirements for an operating system are

A. Memory Requirements: To ensure proper memory management for an operating system the minimum memory requirement of the software must be very low. This concerns RAM as well as persistent program storage.
B. Limited resources: The hardware platforms offer very limited resources so the operating System should use them anciently.
C. Concurrency: The operating system should be able to handle different tasks at the same time.
D. Flexibility: Since the requirements for deferent applications vary wildly, the operating system should be able to be flexible to handle those.
E. Low Power: Energy conservation should be one of the main goals for the operating system.
F. Error Free: Error free that mean it has no chances of error in performing tasks.
G. Platform support: The software part of the IOT should have the ability to support the different hardware platforms, but also the ability to exploit their capabilities.
H. Reliability: The operating system must function reliably because it is deployed in critical applications where the access is linked to a high costs.

V. CONCLUSION AND FUTURE WORK

In this work we have defined the notion of the Internet of things and its different fields of applications.
We have also compared the current (IoT) operating systems in a detailed way to see their advantages and disadvantages as well as their characteristics and architectures. Through this comparison we have deduced that the RIOT is the most appropriate system for

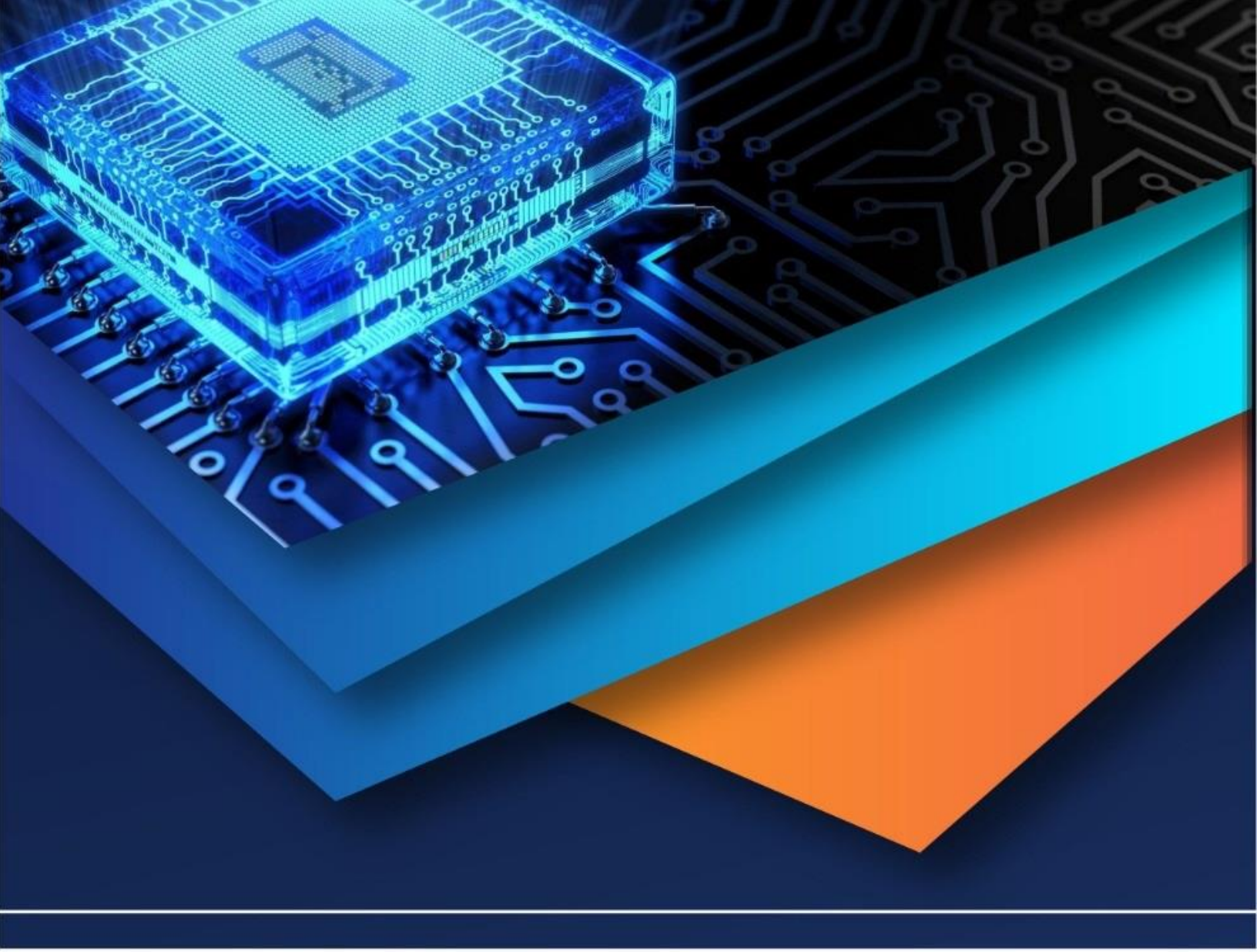
©IJRASET (UGC Approved Journal): All Rights are Reserved

the Internet of things since most of its Strengths correspond exactly to the Requirements that an operating system must satisfy to be used in the internet of things .So this article helps to better choose which operating systems to adapt for the internet of the things .This paper analyzed these parts in a single conclusion: RIOT is an operating system designed for the particular requirements of Internet of Things (IoT) Scenarios.

REFERENCES

[1] www.wikipedia.com
[2] http://www.google.com/
[3] Julian Le Such, Thursday 5 January 2012. Free RTOS sure ATmega328.
[4] Free Software Foundation https://www.gnu.org/licenses/lgpl3.0.en.html.
[5] RIOT (2015). RIOT The friendly Operating System for the Internet of Things. RIOTOS. Org. Retrieved from www.riotos.org/ on 2016, Jan.

©IJRASET (UGC Approved Journal): All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)