Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 534378, 9 pages
http://dx.doi.org/10.1155/2015/534378

Review Article

Software Toolkits: Practical Aspects of the Internet of Things—A Survey

Feng Wang,[1] Liang Hu,[1] Jin Zhou,[2] Yang Wu,[1] Jiejun Hu,[1] and Kuo Zhao[1]

[1]*College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[2]*College of Information Science and Technology, Bohai University, Jinzhou 121000, China*

Correspondence should be addressed to Kuo Zhao; zhaokuo@jlu.edu.cn

Received 20 May 2015; Revised 21 July 2015; Accepted 24 August 2015

Academic Editor: Michael J. O'Grady

The Internet of Things (IoT) is neither science fiction nor industry hype; rather it is based on solid technological advances and visions of network ubiquity that are zealously being realized. The paper serves to provide guidance regarding the practical aspects of the IoT. Such guidance is largely missing in the current literature in which the focus has been more on research problems and less on issues describing how to set up an IoT system and what software toolkits are required. This paper aims to bridge this gap by providing guidance in terms of the software toolkits that can be utilized by IoT practitioners and research students to set up a practical IoT system. The IoT is the combination of multiple techniques; a onefold technology cannot become the IoT. This is conducted in a multilayer approach that covers node and operating systems, middleware, databases, IoT energy harvesting method, and so forth. We hope that this survey serves to be useful to researchers and practitioners in practical aspects of IoT.

1. Introduction

In 2005, the International Telecommunication Union in Tunisia formally identified the concept of the IoT at the World Summit on Information Society and released ITU internet reports. The report gave an in-depth introduction to the IoT and its effect on businesses and individuals around the world. It contained information on key emerging technologies, market opportunities, and policy implications. In the report, the IoT is described as follows: connections will multiply and create an entirely new dynamic network of networks—that is, the IoT.

Since its introduction, the IoT received considerable attention from around the world. IBM has been working with companies, cities, and communities around the world to build a smarter planet for over five years. The following three primary characteristics of the IoT were generalized by IBM in its smarter planet plan: instrumented, intelligent, and interconnected [1]. IoT is neither science fiction nor industry hype; rather it is based on solid technological advances and visions of network ubiquity that are zealously being realized.

The motivation of this paper is to provide guidance regarding the practical aspects of the IoT. This is a gap in the current literature. We therefore aim to bridge this gap by describing software toolkits that can be utilized by IoT practitioners and research students to help realize a practical IoT system. We employ a multilayer approach that covers node operating systems, middleware, database and IoT energy harvesting method, and so forth. In addition to this introductory section, the rest of our paper is organized as follows. From Section 2 to Section 5, we summarize the node and operating systems, IoT middleware, data storage of the IoT, and IoT energy harvesting. Finally we conclude our paper in Section 6.

2. Node and Operating System

Numerous heterogeneous devices comprise the IoT, including resource-constrained wireless sensors [2, 3], household appliances, automation equipment, mobile phones, and tablets. Below, we review several popular node and operating systems running on some of these heterogeneous devices.

TinyOS [4] is an open source, BSD-licensed OS designed for low-power and resource-constrained wireless devices. Unlike many other node operating systems, TinyOS is coded in nesC, a C dialect optimized for the memory limitations

of sensor networks (SNs). TinyOS is component based and event-driven, providing a series of reusable components. Programs are built from software components that are connected to one another using interfaces. TinyOS is completely nonblocking and has one stack to accomplish split-phase operations for maintaining high concurrency. TinyOS is widely used in WSNs and can be ported to various kinds of sensor nodes. TinyOS employs nesC as its programming language. nesC, an event-driven programming language based on components, is designed for building applications that run on TinyOS. nesC is built as an extension to C. Extensions include the separation of construction and composition, specification of component behavior in terms of a set of interfaces, bidirectional interfaces, components statically linked to one another via their interfaces, and code generated by whole-program compilers. Alizai et al. [5] proposed the TinyWifi platform, built on nesC code, as a base for protocol testing and evaluation in multiple heterogeneous networks and discussed the requirements and challenges in such a solution.

Contiki [6] is a lightweight, open source node OS for networked, memory-constrained systems with a particular focus on low-power wireless IoT devices. Contiki was created by Adam Dunkels in 2002 and was further developed by a worldwide team of developers from Atmel, Cisco, Enea, ETH Zurich, and Oxford University. Contiki provides multitasking and a built-in TCP/IP stack, including IPv6 networking, as well as 6LoWPAN, RPL, and CoAP, but it only requires approximately 10 kilobytes of RAM and 30 kilobytes of ROM. ContikiMAC enables nodes to run in a low-power mode and still be able to receive and relay radio messages. Even wireless routers can be battery operated. Created specifically for the IoT, the 6LoWPAN of Contiki makes it possible to connect resource-constrained devices to the internet because there are plenty of available IP addresses.

Nut/OS [7] is a small open source real-time OS licensed under the BSD license. It is the OS for boards under the project Ethernut, an open source hardware and software embedded-Ethernet-system. All reference designs of the hardware share the same set of interfaces such as Ethernet connectors, RS-232 ports, and power connectors, and nearly all I/O pins of the microcontroller are available on an expansion port. The primary differences are the use of a microcontroller and available memory size. The Nut/OS supports cooperative multithreading and is most applicable to situations in which there are several concurrent tasks that must be simultaneously executed. Ethernut implements its own network stack called Nut/Net, which implements a number of protocols and socket APIs for use in Nut/OS. Nut/OS is in fact an embedded-Ethernet-system that controls and monitors embedded devices without a PC. Therefore, this OS may be used as part of an intelligent home system or many other such applications of the IoT.

IBM Zurich Research Laboratory developed Mote Runner, a high-performance, resource-efficient middleware platform with hardware-agnostic and language-independent features. It provides a viable solution to some of the cost-related and technological problems of large-scale applications in the IoT. As shown in Figure 1, using Mote Runner, applications that are dynamically distributed, loaded, updated, and deleted

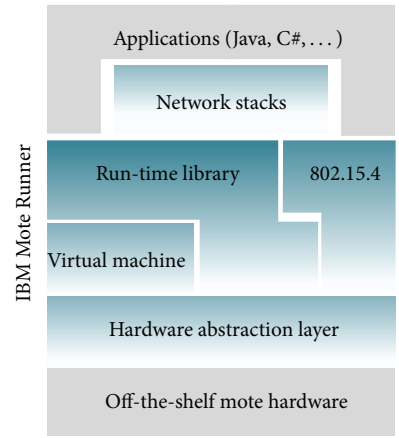

Figure 1: IBM Mote Runner architecture.

can be easily created and managed with a high-level object-oriented programming language by using an event-driven programming model. In this manner, a large amount of postdeployment and maintenance costs are reduced. In addition, with state-of-the-art back-end infrastructures, Mote Runner makes it possible for IoT applications to be seamlessly integrated.

The Raspberry Pi [8] is an open source hardware system with strong manipulability. There are several OSs that can be ported to Raspberry Pi, including GNU/Linux flavors (e.g., Debian, Fedora, Arch Linux ARM, and Android), RISC OS, FreeBSD, and Plan 9. The Raspbian, which is a Debian-based OS optimized for the Raspberry Pi, is the OS recommended by the Raspberry Pi Foundation. Although the Raspberry Pi is a credit-card-sized single-board computer, the computing and memory resources are more remarkable than the typical sensor nodes or boards.

Arduino [9] is another open source hardware system similar to the Raspberry Pi. Just as with the Raspberry Pi, we can also port various OSs to the Arduino, including the Free Real Time Operating System (FreeRTOS), Duin OS, Pyxis OS, ArduinoMacOS, TaOS, Contiki, and FemtoOS. Arduino's microcontroller is based on the 8-bit Atmel AVR or 32-bit Atmel ARM, while the software consists of a standard programming language compiler and a boot loader that executes on the microcontroller. OS programs are written either in C or in C++.

Android [10] is a Linux-based OS designed primarily for touchscreen mobile devices; however, Android can be ported to many other devices such as the Raspberry Pi. Android uses Java for creating interfaces and functions. Although Android has a Linux kernel, it does not have a native X Window System by default or the full set of standard GNU libraries, which makes it difficult to port existing Linux applications or libraries to Android. It runs on embedded devices that generally have sufficient computing and storage resources beyond that of the Raspberry Pi. Therefore, Android provides a much higher level and richer set of services in the IoT, but the consumption is inevitably much more than that of traditional sensor nodes.

Table 1 summarizes and compares the above OSs.

Table 1: IoT node OSs comparison.

	TinyOS	Contiki	Nut/OS	Mote Runner	Raspbian	FreeRTOS	Android
Real time	√	√	√	√	√	√	√
Hardware platform	Low-power wireless devices	Low-power wireless devices, console platforms, computers	Low-power wired devices	Low-power wireless devices	Raspberry Pi	Arduino	Smart devices: mobile phones, computers, home appliances
Resource requirements	Low	Low	Low	Low	Middle	Middle	High
Energy consumption	Low	Low	Low	Low	High	High	High
IPv6 support	√ (2.1.2 supports it)	√	✗ (planned in future release)	√	√	√	√
Network communication protocol	IEEE 802.15.4 [37]	IEEE 802.15.4	IEEE 802.3, RS-232	IEEE 802.15.4	IEEE 802.3, RS-232, IEEE 802.11 (Raspberry Pi supporting not perfect)	IEEE 802.3, RS-232, IEEE 802.11, GPRS	IEEE 802.15.4, IEEE 802.11, Bluetooth, GPRS, 3G

3. IoT Middleware

Middleware is a type of computer software that connects different software components and applications. It consists of a set of enabling services that allow multiple processes running on one or more machines to interact with one another across a network. For many new applications, middleware components are becoming more important than the underlying OS and networking services on which the applications formerly depended [11].

Various types of middleware exist, including message-oriented middleware, object middleware, Remote Procedure Call middleware, database middleware, transaction middleware, and portals. Atzori et al. [12] summarized the relation of IoT components as being things-oriented, semantic-oriented, and internet-oriented visions. As summarized in Figure 2, according to these characteristics, middleware in the IoT should be able to address things and internet-related issues, as well as issues related to the semantics gap, including interoperability across heterogeneous devices, context awareness, device discovery, management of resource-constrained embedded devices, scalability, management of large volumes of data, and privacy.

In [13], Wang et al. reviewed middleware for WSNs and provided a detailed analysis of the approaches and techniques offered by middleware to meet the requirements of WSNs. Wang et al. also discussed generic components of the middleware and a reference model of WSN-based middleware. In [14], middleware was surveyed from an adaptability perspective that presented a taxonomy for adaptive middleware and its application domains and provided details for each middleware category.

Context-aware middleware was also studied. The survey in [15] was based on the architectural aspects and provided a taxonomy of the features of a generic context-aware middleware. The survey reported in [16] evaluated several context-aware architectures based on relevant criteria from the ubiquitous and pervasive computing perspectives. Finally, the survey presented in [17] was a survey of middleware systems for the IoT.

Hydra [18] is a middleware that bridges a scientific workflow management system and high-performance computing to enable workflow parallelization with provenance gathering. Coutinho et al. [19] used it to support a data parallelism solution for a bioinformatics experiment. Anggorjati et al. [20] introduced ASPIRE, which focused on the design, development, and adoption of an innovative, programmable, royalty-free, lightweight, and private RFID middleware. Naumenko et al. [21] described their long-term vision for security and privacy management in complex multiagent systems, such as UBIWARE. UBISOAP is a service-oriented middleware for WSNs [22] with a two-layer architecture based on SOAP.

Terziyan et al. [23] proposed the requirements and architecture of a complex traffic management system and illustrated how such systems may benefit by using semantic and agent technologies with the UBIROAD middleware. Belmonte et al. [24] presented a real implementation of a coalition and incentive theoretical mechanism for content distribution that aimed to tackle the problems in user behavior by using SMEPP, a middleware that aims to ease the development of secure distributed applications.

Cannata et al. [25] proposed that the framework for addressing intelligent systems should adopt SOCRADES, a European research project exploiting the SOA paradigm at the device and application levels. Maló et al. [26] presented the SIMPLE self-organized and intelligent middleware platform, which focuses on data-dissemination using multilevel

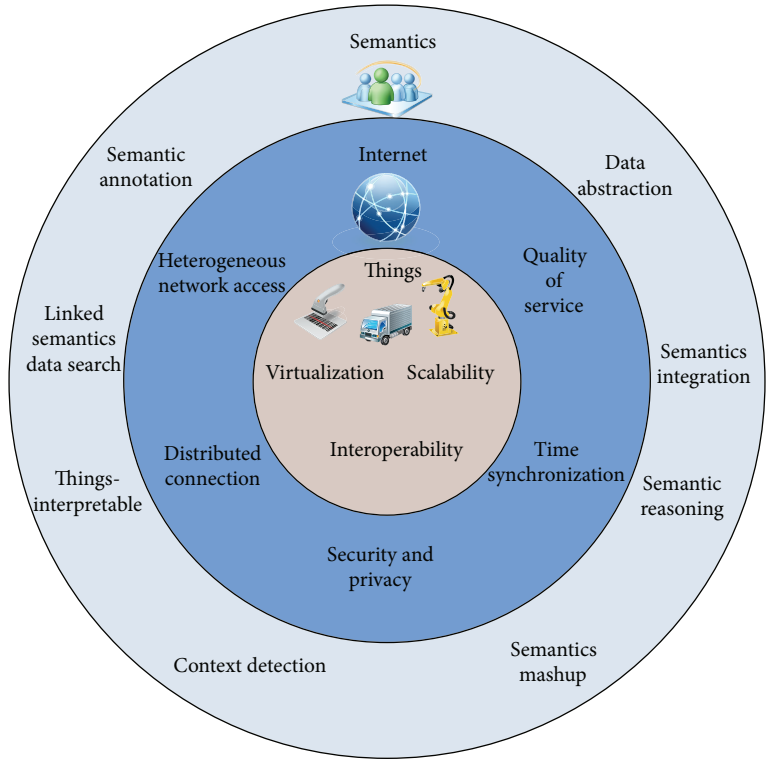

Figure 2: Specific functional areas of IoT middleware.

subscriptions processing and a tiered networking approach to cope with many disparate, widespread, and heterogeneous sensing networks.

Global sensor networks (GSNs) are designed to facilitate the deployment and programming of SNs and serve as middleware to connect the information from the IoT to the internet. GSN is a Java environment that collects live data from a set of wrappers, processing these data streams based on XML specification files, then releasing the processed data to the web. A GSN wrapper is a piece of Java code that performs data acquisition for various devices and different types of data sources, including local sensor nodes, remote sensor nodes, TinyOS nodes, CSV files, HTTP GET requests, serial ports, and USB cameras. Compared with existing middleware [27], GSN implements the feature of automatic process of generating wrappers. Given these characteristics, GSN has better connectivity and configurability to deploy embedded sensor devices. In 2014, an extended version of GSN was presented by [28]. XGSN is capable of enriching virtual sensor descriptions with semantically annotated content using standard vocabularies.

The Message Queuing Telemetry Transport (MQTT) protocol is a lightweight, real-time messaging protocol developed by IBM. Although MQTT is lightweight, it is powerful and robust enough to support smart devices in the IoT, including Arduino platform devices, smartphones, PDAs, and tablets, efficiently delivering messages between them. This makes it possible to link the IoT to the internet. The MQTT protocol has three types of service primitives: (a) at most one delivery; (b) at least one delivery; and (c) exactly one delivery. Different service primitives correspond to different QoS levels and flows. Lee et al. [29] analyzed the MQTT message transmission process through three levels of QoS with various payload sizes, and they also analyzed end-to-end delays and message loss. Using MQTT, IoT services can be fluently and safely transmitted between the internet and devices of the IoT.

Fuseki, a server providing access to the Simple Protocol and Resource Description Framework Query Language (SPARQL) 1.1 standard, uses it via HTTP and provides RESTful HTTP updating, querying, and updating. SPARQL is a querying language and data acquisition protocol developed for RDF by the W3C. It links two new web technologies, namely, Web 2.0 and the semantic web, together and might be the future querying language of mainstream network databases and data acquisition standards. After tagging heterogeneous data to form semantic data, Fuseki can be used to release such data to the web. Therefore, the IoT is connected to the traditional internet-based semantic data.

Operator [30], developed by Michael Kaply from IBM and arguably one of the important projects of the Mozilla labs, connects microformats and semantic data from many web pages, bringing us a new web service model. Microformats are a set of simple open data formats built upon existing and widely adopted standards designed for humans first and machines second. Operator makes it possible for us to integrate different data for different web services. An important issue is to study how the IoT should be linked to the internet for which the Operator middleware provides a possible solution.

ThingSpeak [31] is a decentralized, open source platform and API for the IoT. ThingSpeak provides a server that may be used to facilitate data storage and retrieval from IoT devices. One of the advantages of ThingSpeak is openness to different hardware profiles and provides visualization tools and enables the data in a more personified fashion.

FamiWare [32] is a middleware family for ambient intelligence that allows the customization and instantiation of a specific middleware platform taking into consideration the peculiarities of every device and application domain, with the goal of reducing the middleware size.

SIXTH [33] is a Java-based middleware for the Sensor Web. It allows sensor-driven applications and enables the dynamic tasking of sensor nodes to suit shifting application demands in near real time. AndroSIXTH [34] aims to improve network middleware SIXTH with discovery services and extend its abilities to mobile networks.

MOSDEN [35] captures and shares sensor data across multiple apps, smartphones, and users. MOSDEN supports the emerging trend of separating sensors from application-specific processing, storing, and sharing. MOSDEN promotes reuse and repurposing of sensor data hence reducing the efforts in developing novel opportunistic sensing applications. MOSDEN has been implemented on Android-based smartphones and tablets.

ManySense [36] is an Android-based middleware for heterogeneous consumer-oriented body sensor networks which allow sensors and context inferencing algorithms to be coupled with the middleware. And its accessibility allows third-party applications to access raw sensor data and inferred context data uniformly.

Table 2 shows a comparison of the existing middleware from the perspective of the IoT.

4. Data Storage via RDF

In the IoT, as discussed above, the semantic vision is that the semantic meanings of objects are stored separately and effective tools exist for managing this information. As a key to the IoT, these data should be reconstructed using the RDF [38] model and published as Linked Data [39] on the web.

Information about resources is represented using the RDF, which provides a data model that is very simple but strictly tailored toward the architecture of the web. In RDF, a description of a resource is represented as a number of triples. The three parts of each triple are its subject, predicate, and object. RDF data can be accessed by HTTP URIs.

The main benefits of using the RDF data model in a Linked Data context are as follows:

(a) users can look up every URI in an RDF graph over the web to retrieve additional information;

(b) information from different sources merges naturally;

(c) the data model enables the users to set RDF links between data from different sources;

(d) the data model allows users to represent information expressed using different schemata in a single model;

(e) combined with schema languages such as RDF-S or OWL, the data model allows users to use as much or as little structure as they need; that is, users can represent tightly structured and semistructured data.

There are data management and storage systems that can store raw sensing data and its corresponding semantics data, managing the relations between the two via a mapping.

The D2RQ Platform [40, 41] is a system for accessing relational databases as virtual read-only RDF graphs. It offers RDF-based access to the content of relational databases without having to replicate it in an RDF store. Database content is mapped to RDF by a declarative mapping that specifies how resources are identified and property values are generated from database content. Benefits of D2RQ are as follows:

(a) it queries a non-RDF database using SPARQL;

(b) it accesses the content of the database as Linked Data over the web;

(c) it creates custom dumps of the database in RDF formats for loading into an RDF store;

(d) it accesses information in a non-RDF database using the Apache Jena API;

(e) it has rich interlink RDF and XHTML to enable browsers and crawlers to navigate database content.

As shown in Figure 3, the D2RQ Platform is composed of the following components:

(a) the D2RQ Mapping Language, a declarative mapping language for describing relations between an ontology and a relational data model;

(b) the D2RQ Engine, a plug-in for the Jena Semantic Web toolkit, which uses the mappings to rewrite Jena API calls to SQL queries against the database and then passes query results to the higher layers of the framework;

(c) the D2R Server, a type of HTTP server that provides a Linked Data view, an HTML view for debugging, and a SPARQL Protocol endpoint over the database.

Supported databases currently include Oracle, MySQL, PostgreSQL, SQLServer, HSQLDB, and Interbase/Firebird.

There are other data storage systems worth reviewing. In [42], David et al. provide an overview of available triple stores, their technical implementations, support for the SPARQL W3C recommendations, and available APIs.

5. IoT Energy Harvesting

With the development of the IoT, an increasing number of resource-constrained wireless sensors are being applied in industrial, commercial, medical, consumer, and military fields. Minimizing power consumption is the key to extending the application life of wireless sensors and reducing costs. Energy harvesting is a technique for utilizing energy that is harvested from the environment. Energy collected from the environment is stored after conversion and then assigned to various parts of the wireless sensor, as shown in Figure 4.

Table 2: IoT middleware comparison.

	Platform portability	Semantics mapping	Mashup	Quality of service	Virtualization	Interoperation	Device management interface	Security and privacy
GSN	√	✗	√	√	√	√	√	√
MQTT	√	✗	✗	√	✗	√	√	√
Fuseki	√	√	√	✗	√	√	✗	✗
Operator	√	√	√	✗	√	√	✗	✗
HYDRA	√	√	√	✗	✗	√	√	√
ASPIRE	✗	✗	✗	✗	✗	✗	√	✗
UBIWARE	√	√	√	√	√	✗	√	✗
UBISOAP	√	✗	✗	✗	✗	√	√	✗
UBIROAD	√	√	√	√	√	√	√	√
SMEPP	√	√	√	✗	✗	√	√	√
SOCRADES	√	✗	✗	✗	✗	√	√	√
ThingSpeak	√	✗	√	✗	√	√	√	√
FamiWare	√	✗	√	✗	✗	√	√	✗
AndroSIXTH	√	✗	√	√	✗	√	✗	✗
MOSDEN	✗	✗	√	✗	✗	✗	✗	✗
ManySense	√	✗	√	✗	✗	✗	✗	✗

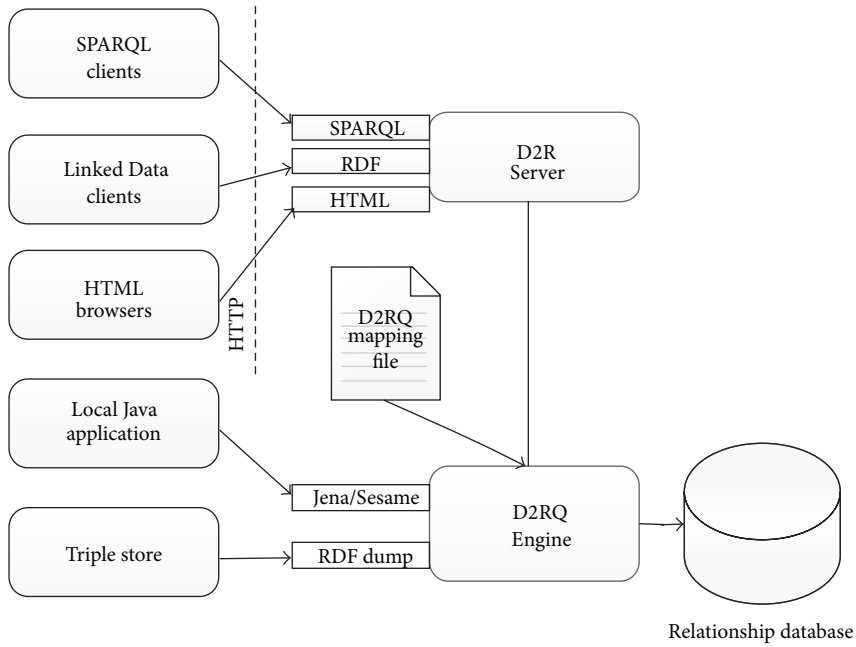

Figure 3: D2RQ Platform.

The efficient use of energy in WSNs involves energy supply and energy consumption. These issues are core to successfully designing and deploying WSNs. Most research has focused on harvesting and using ambient energy. In [43], Gorlatova et al. proposed a kinetic energy collection method involving experiments with briefly moving objects that provided insights into the suitability of different scenarios for harvesting energy. To characterize the energy availability associated with specific human activities (e.g., relaxing, walking, and cycling), Gorlatova et al. analyzed a motion data set with more than 40 participants. Their observations provided insights into the design of networking algorithms and motion energy harvesters, which could then be embedded into mobile devices. If this technology is embedded into wireless sensors, moving wireless sensor nodes would achieve self-sufficiency in terms of energy.

In [44], Zhang et al. introduced an effective method for converting ambient energy to electrical energy. Their research enables wireless sensor nodes to power themselves by converting ambient energy into electrical energy. Among the

Table 3: Comparison of energy harvesting methods.

	Solar	Wind energy	Sound energy	Vibrational energy	Thermoelectric energy
Energy production	Optoelectronic materials	Piezoelectric and mechanical technology	Artificial lithium niobate, thermoacoustic engine	Mechanical vibration	Temperature difference generator
Amount of energy	Photocell 0.5 V	8–16 km/h wind velocity 50 mW	160 Db 100 kW	28 Hz 100 mg 9.3 mW	5°C 1.5 V
Factors	Illumination environment, area, intensity Temperature	Wind speed, piezoelectric equipment and materials	Sound decibel	Volume of the collector Vibration frequency Vibration force	The temperature difference Output efficiency
Application	Small remote sensors	Power station	Power station, huge turbines	Small sensors	Small power less than 5 W

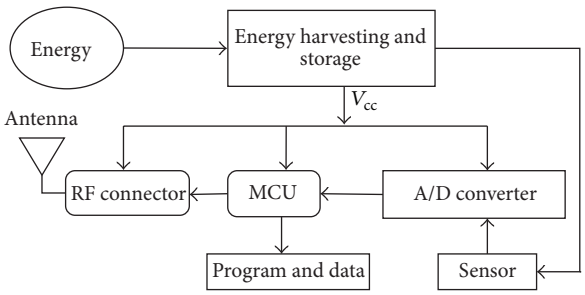

Figure 4: Wireless sensor node energy supply.

various energy types, ambient vibration energy is particularly attractive owing to its abundance. In [45], Sudevalayam and Kulkarni compared the physical needs and energy requirements for processing and communication tasks conducted by sensor nodes. After demonstrating hard limits, Sudevalayam and Kulkarni compared relevant energy harvesting methods and determined the most likely candidates applicable to the IoT.

Table 3 shows various ambient energy harvesting methods, influencing productions, factors, and applications [46–50]. Methods for using natural energy are straightforward and cost is typically low, but there are many unsolved issues, including low energy efficiency and dependence on the environment. Other unnatural energy sources are easily controlled, but the process of generating energy is typically very complex and expensive. The limited battery power carried by wireless sensors is not suitable in the long run. Therefore, we must identify long-standing, low-power, easy-to-use energy harvesting methods for wireless sensors.

6. Conclusions

The IoT is in the early development stages; the scalable and durable network will be deposited effectively in the future. The IoT is the combination of multiple techniques; a single technology cannot become the IoT. In this survey paper, we provided some guidance regarding the practical aspects of the IoT and give guidance of the practicality aspect of IoT which is largely missing in the current literature where the focus has been more on the research problems and less on issues such as how to set up an IoT system step by step and what software toolkits are needed.

We hope that this survey has served to be useful to researchers and practitioners in the field, helping them by providing some guidance in terms of the software toolkits that can be utilized to construct a practical IoT system. Predictably, the IoT will be information infrastructure in people's future lives and more efforts to tackle those challenging issues shall be made from both industry and academia to promote the progress of the IoT.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is funded by the European Framework Program (FP7) under Grant no. FP7-PEOPLE-2011-IRSES, National Natural Science Foundation of China under Grant nos. 61073009 and 61103197, National High Tech R&D Program 863 of China under Grant no. 2011AA010101, National Sci-Tech Support Plan of China under Grant no. 2014BAH02F03, National Sci-Tech Major Projects of China under Grant nos. SinoProbe-09-01-03 and 2012ZX01039-004-04-3, Key Sci-Tech Program of Jilin Province of China under Grant nos. 2011ZDGG007 and 20150204035GX, and Fundamental Research Funds for Central Universities of China under Grant nos. JCKY-QKJC46 and 2412015KJ005.

References

[1] S. J. Palmisano, *A Smarter Planet: The Next Leadership Agenda*, vol. 6, IBM, 2008.

[2] S. Kaur, "On-chip networks!," *IETE Technical Review*, vol. 30, no. 3, pp. 168–172, 2013.

[3] P. Yi, Y. Wu, F. T. Zou, and N. Liu, "A survey on security in wireless mesh networks," *IETE Technical Review*, vol. 27, no. 1, pp. 6–14, 2010.

[4] P. Levis, S. Madden, J. Polastre et al., "TinyOS: an operating system for sensor networks," in *Ambient Intelligence*, pp. 115–148, Springer, Berlin, Germany, 2005.

[5] M. H. Alizai, H. Wirtz, B. Kirchen, and K. Wehrle, "Portable wireless-networking protocol evaluation," *Journal of Network and Computer Applications*, vol. 36, no. 4, pp. 1230–1242, 2013.

[6] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki—a lightweight and flexible operating system for tiny networked sensors," in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks (LCN '04)*, pp. 455–462, IEEE, November 2004.

[7] F. Dressler, M. Strübe, R. Kapitza, and W. Schröder-Preikschat, "Dynamic software management on BTnode sensors," in *Proceedings of the 4th IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS '08): IEEE/ACM International Workshop on Sensor Network Engineering (IWSNE '08)*, pp. 9–14, Santorini Island, Greece, June 2008.

[8] E. Upton and G. Halfacree, *Raspberry Pi User Guide*, John Wiley & Sons, 2014.

[9] J. Oxer and H. Blemings, *Practical Arduino: Cool Projects for Open Source Hardware*, Apress, 2009.

[10] E. Burnette, *Hello, Android: Introducing Google's Mobile Development Platform*, Pragmatic Bookshelf, 2009.

[11] P. A. Bernstein, "Middleware: a model for distributed system services," *Communications of the ACM*, vol. 39, no. 2, pp. 86–98, 1996.

[12] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[13] M.-M. Wang, J.-N. Cao, J. Li, and S. K. Dasi, "Middleware for wireless sensor networks: a survey," *Journal of Computer Science and Technology*, vol. 23, no. 3, pp. 305–326, 2008.

[14] S. M. Sadjadi and P. K. McKinley, "A survey of adaptive middleware," Report MSU-CSE-03-35, Michigan State University, 2003.

[15] K. E. Kjær, "A survey of context-aware middleware," in *Proceedings of the 25th Conference on IASTED International Multi-Conference: Software Engineering*, pp. 148–155, February 2007.

[16] M. Miraoui, C. Tadj, and C. B. Amar, "Architectural survey of context-aware systems in pervasive computing environment," *Ubiquitous Computing and Communication Journal*, vol. 3, pp. 1–9, 2008.

[17] S. Bandyopadhyay, M. Sengupta, S. Maiti, and S. Dutta, "A survey of middleware for internet of things," in *Recent Trends in Wireless and Mobile Networks*, vol. 162 of *Communications in Computer and Information Science*, pp. 288–296, Springer, Berlin, Germany, 2011.

[18] F. Coutinho, E. Ogasawara, D. de Oliveira et al., "Many task computing for orthologous genes identification in protozoan genomes using Hydra," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 17, pp. 2326–2337, 2011.

[19] F. Coutinho, E. Ogasawara, D. de Oliveira et al., "Data parallelism in bioinformatics workflows using Hydra," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC '10)*, pp. 507–515, June 2010.

[20] B. Anggorjati, K. Çetin, A. Mihovska, and N. R. Prasad, "RFID added value sensing capabilities: European advances in integrated RFID-WSN middleware," in *Proceedings of the 7th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '10)*, pp. 1–3, June 2010.

[21] A. Naumenko, A. Katasonov, and V. Terziyan, "A security framework for smart ubiquitous industrial resources," in *Enterprise Interoperability II*, pp. 183–194, Springer, Berlin, Germany, 2007.

[22] M. Ananthi and M. R. Sumalatha, "Integrating WSN with web services for patient's record management using RFID," in *Proceedings of the 3rd IEEE International Advance Computing Conference (IACC '13)*, pp. 605–609, February 2013.

[23] V. Terziyan, O. Kaykova, and D. Zhovtobryukh, "UbiRoad: semantic middleware for context-aware smart road environments," in *Proceedings of the 5th International Conference on Internet and Web Applications and Services (ICIW '10)*, pp. 295–302, IEEE, Barcelona, Spain, May 2010.

[24] M.-V. Belmonte, M. Díaz, and A. Reyna, "Coalitions and incentives for content distribution over a secure peer-to-peer middleware," in *Proceedings of the 3rd International Conference on Advances in P2P Systems (AP2PS '11)*, pp. 71–78, November 2011.

[25] A. Cannata, M. Gerosa, and M. Taisch, "SOCRADES: a framework for developing intelligent systems in manufacturing," in *Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM '08)*, pp. 1904–1908, December 2008.

[26] P. Maló, B. Almeida, R. Melo, K. Kalaboukas, and P. Cousin, "Self-organised middleware architecture for the internet-of-things," in *Proceedings of the IEEE International Conference on and IEEE Cyber, Physical and Social Computing Green Computing and Communications (GreenCom '13) and Internet of Things (iThings/CPSCom '13)*, pp. 445–451, Beijing, China, August 2013.

[27] C. Perera, A. Zaslavsky, P. Christen, A. Salehi, and D. Georgakopoulos, "Connecting mobile things to global sensor network middleware using system-generated wrappers," in *Proceedings of the 11th ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pp. 23–30, May 2012.

[28] J.-P. Calbimonte, S. Sarni, J. Eberle, and K. Aberer, "XGSN: an open-source semantic sensing middleware for the web of things," in *Proceedings of the 7th International Workshop on Semantic Sensor Networks*, Riva del Garda, Italy, 2014.

[29] S. Lee, H. Kim, D.-K. Hong, and H. Ju, "Correlation analysis of MQTT loss and delay according to QoS level," in *Proceedings of the 27th International Conference on Information Networking (ICOIN '13)*, pp. 714–717, January 2013.

[30] S. Nasrolahi, M. Nikdast, and M. M. Boroujerdi, "The semantic web: a new approach for future world wide web," *World Academy of Science, Engineering & Technology*, no. 34, p. 1149, 2009.

[31] M. Adda and R. Saad, "A data sharing strategy and a DSL for service discovery, selection and consumption for the IoT," *Procedia Computer Science*, vol. 37, pp. 92–100, 2014.

[32] N. Gámez and L. Fuentes, "FamiWare: a family of event-based middleware for ambient intelligence," *Personal and Ubiquitous Computing*, vol. 15, no. 4, pp. 329–339, 2011.

[33] D. Carr, M. O'Grady, G. P. O'Hare, and R. Collier, "SIXTH: a middleware for supporting ubiquitous sensing in personal health monitoring," in *Wireless Mobile Communication and Healthcare*, B. Godara and K. Nikita, Eds., vol. 61 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 421–428, Springer, Berlin, Germany, 2013.

[34] L. Görgü, B. Kroon, A. Campbell, and G. P. O'Hare, "Enabling a mobile, dynamic and heterogeneous discovery service in a sensor web by using AndroSIXTH," in *Ambient Intelligence*, J. Augusto, R. Wichert, R. Collier, D. Keyson, A. Salah, and A.-H.

Tan, Eds., vol. 8309 of *Lecture Notes in Computer Science*, pp. 287–292, Springer, 2013.

[35] P. Jayaraman, C. Perera, D. Georgakopoulos, and A. Zaslavsky, "MOSDEN: a scalable mobile collaborative platform for opportunistic sensing applications," *EAI Endorsed Transactions on Collaborative Computing*, vol. 1, pp. 1–16, 2014.

[36] J. Westlin and T. H. Laine, "ManySense: an extensible and accessible middleware for consumer-oriented heterogeneous body sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2014, Article ID 321534, 15 pages, 2014.

[37] A. F. Molisch, K. Balakrishnan, D. Cassioli et al., "IEEE 802.15. 4a channel model—final report, 2006," *IEEE P802*, vol. 15, no. 4, 662 pages, 2011.

[38] G. Klyne, J. J. Carroll, and B. McBride, "Resource description framework (RDF): concepts and abstract syntax," W3C Recommendation 10, W3C, 2004.

[39] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far," *International Journal on Semantic Web and Information Systems*, vol. 5, no. 3, pp. 1–22, 2009.

[40] C. Bizer and A. Seaborne, "D2RQ-treating non-RDF databases as virtual RDF graphs," in *Proceedings of the 3rd International Semantic Web Conference (ISWC '04)*, 2004.

[41] C. Bizer and R. Cyganiak, "D2r server-publishing relational databases on the semantic web," in *Proceedings of the 5th International Semantic Web Conference*, p. 26, 2006.

[42] C. David, C. Olivier, and B. Guillaume, "A survey of RDF storage approaches," *ARIMA Journal*, vol. 15, pp. 11–35, 2012.

[43] M. Gorlatova, J. Sarik, G. Grebla, M. Cong, I. Kymissis, and G. Zussman, "Movers and shakers: kinetic energy harvesting for the internet of things," in *The 2014 ACM International Conference on Measurement and Modeling of Computer Systems*, pp. 407–419, ACM, 2014.

[44] Z. Zhang, D. Xianzhi, and W. Yong, "An improved magneto-electric vibration energy harvester for wireless sensors," in *Proceedings of the IEEE 14th International Conference on Communication Technology (ICCT '12)*, pp. 589–593, November 2012.

[45] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: survey and implications," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, pp. 443–461, 2011.

[46] R. J. Vullers, R. Schaijk, H. J. Visser, J. Penders, and C. V. Hoof, "Energy harvesting for autonomous wireless sensor networks," *IEEE Solid-State Circuits Magazine*, vol. 2, no. 2, pp. 29–38, 2010.

[47] J. Liu and W. Tong, "Dynamic share energy provisioning service for one-hop multiple RFID tags identification system," in *Proceedings of the IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS '11)*, pp. 342–347, IEEE, Beijing, China, September 2011.

[48] M. Abdallah, M. Chetto, and A. Queudet, "Scheduling with quality of service requirements in real-time energy harvesting sensors," in *Proceedings of the IEEE International Conference on Green Computing and Communications (GreenCom '12)*, pp. 644–646, November 2012.

[49] R. Yao, W. Wang, M. Farrokh-Baroughi, H. Wang, and Y. Qian, "Quality-driven energy-neutralized power and relay selection for smart grid wireless multimedia sensor based IoTs," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3637–3644, 2013.

[50] S. Devasenapathy, R. Venkatesha Prasad, V. S. Rao, and I. Niemegeers, "Impact of antenna directionality and energy harvesting rate on neighbor discovery in EH-IoTs," in *Proceedings of the IEEE 10th Consumer Communications and Networking Conference (CCNC '13)*, pp. 302–307, IEEE, January 2013.