Computer Systems Science & Engineering
DOI:10.32604/csse.2021.015713
Article

Tech Science Press

A Sensor Network Web Platform Based on WoT Technology

Shun-Yuan Wang[1], Yun-Jung Hsu[1], Sung-Jung Hsiao[2] and Wen-Tsai Sung[3,*]

[1]Department of Electrical Engineering, National Taipei University of Technology, Taipei, 10608, Taiwan
[2]Department of Information Technology, Takming University of Science and Technology, Taipei, 11451, Taiwan
[3]Department of Electrical Engineering, National Chin-Yi University of Technology, Taichung, 41170, Taiwan
*Corresponding Author: Wen-Tsai Sung. Email: songchen@ncut.edu.tw
Received: 03 December 2020; Accepted: 17 February 2021

Abstract: This study proposes a Web platform, the Web of Things (WoT), whose Internet of Things (IoT) architecture is used to develop the technology behind a new standard Web platform. When a remote sensor passes data to a microcontroller for processing, the protocol is often not known. This study proposes a WoT platform that enables the use of a browser in a mobile device to control a remote hardware device. An optimized code is written using an artificial intelligence-based algorithm in a microcontroller. Digital data convergence technology is adopted to process the packets of different protocols and place them on the Web platform for access by other mobile devices. The platform has high efficiency and cross-platform advantages, with no limitation on the operating system. Message queueing telemetry transport (MQTT) technology is used to simplify the original HTTP protocol. Assume that the mobile device is a subscriber, i.e., the controller, and a microcontroller that connects the sensing device is the publisher. The publishers and subscribers of MQTT need not know each other if they share a message broker. The intermediate agent role is much like a router. Publishers and subscribers do not need to interact, and publishers do not have to wait for subscriber confirmation to cause interactive permission be locked. Nor must publishers and subscribers be online at the same time, and they are free to choose when to get messages. The proposed WoT method is compared with the traditional IoT method regarding data transfer. The results show that the proposed method can save time in processing large amounts of data, as the traditional IoT method wastes time, especially in data format transfer.

Keywords: Embedded systems; mobile Web servers; big data analysis; WoT; IoT

1 Introduction

Recent developments of the Internet of Things (IoT) have led to the successful implementation of Web technology in embedded devices, resulting in the Web of Things (WoT), which can be considered a special case of the IoT [1–3]. The WoT takes the IoT to another level, helping to integrate the real and digital worlds. The typical IoT structure and the construction of the WoT [4–6] are presented in Figs. 1 and 2, respectively.

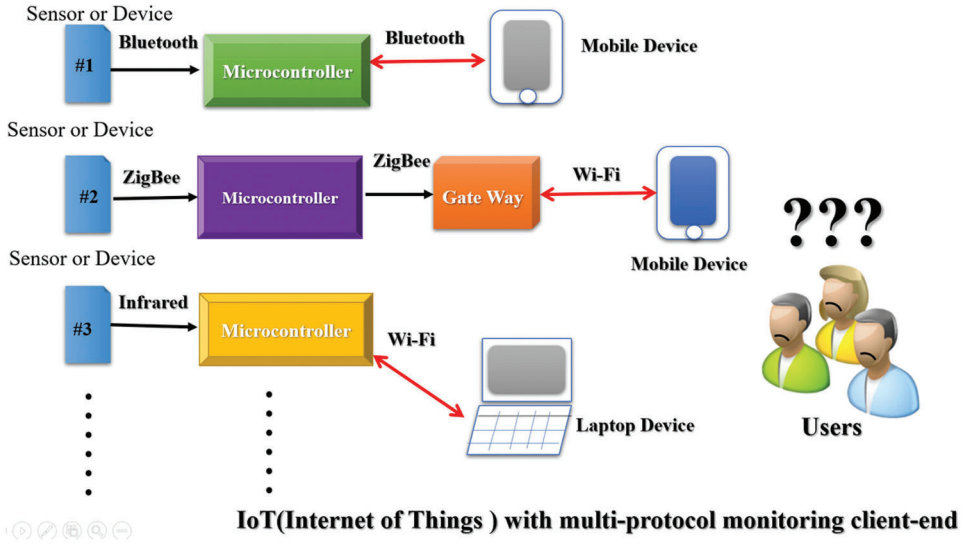

Figure 1: A typical IoT structure with client-end multi-protocol monitoring

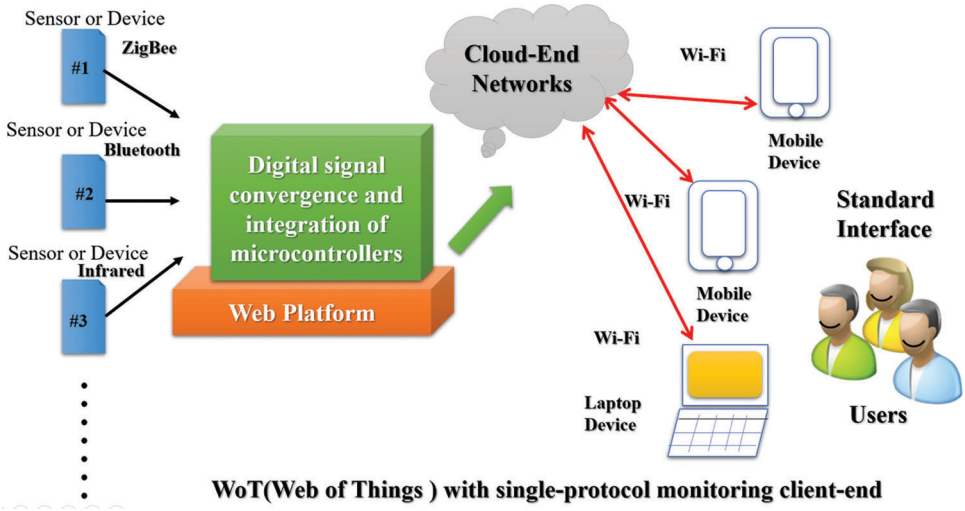

Figure 2: WoT system architecture implemented in a mobile Web service

The authors are aware of no research on related digital message convergence technology, although some international suppliers have used a gateway [7–9]. This study proposes to integrate an original gateway and a computer using Raspberry Pi. This research proposes the integration of digital message convergence technology based on asynchronous message transmission in RS-232 [10–12].

2 Literature and Structure Description

The current IoT application-layer protocol provides several features that are useful for deploying embedded devices, including device and service discovery, message delivery reliability, and secure peer-to-peer pairing. The HTTP (or WebSocket) protocol differs from the IoT protocol in that it does not support these features, which are not designed for embedded devices. In practical implementation, the WoT also needs the basic capabilities of the IoT, and must be able to easily expand other functions to

meet real-world requirements. We present the WoT structure used in this study and discuss how to extend Web protocols and support the functionality required for any type of WoT application while integrating it into the internet [13–16].

Yao proposed the concept of a WoT framework [17]. Kuromiya et al. proposed the technology of the WoT server and developed a mobile communication method [18]. Chen studied the concept of the WoT business environment [19]. Antoniazzi et al. discussed the establishment of the Semantic Web of Things by dynamic ontology [20]. Faheem and Mainetti et al. proposed the WoT for software architecture [21,22].

This study uses the architecture presented in Fig. 3 to examine various layers of the WoT architecture and explain their purpose [23–25]. We present layers 1 to 4, which provide the tools for systems using WoT technology to fully integrate all the features of a device.

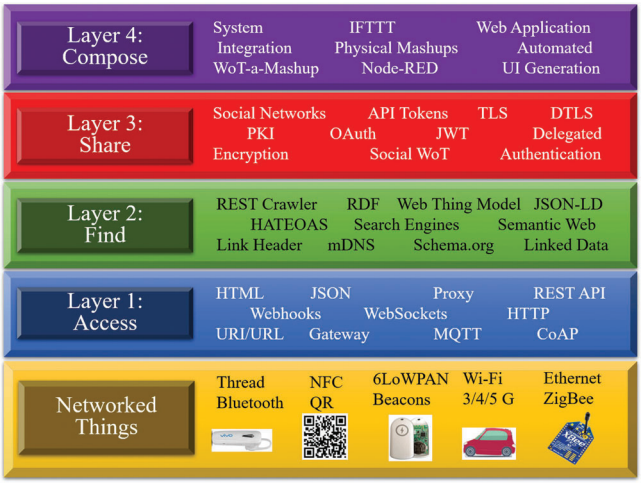

Figure 3: The architecture of the WoT and its layers

2.1 Layer 1: Access

The most basic level, the access layer, provides a Web API to connect intelligent objects to the Web. The access layer transforms intelligent objects to composable Web objects to which other devices and applications can easily connect. Intelligent objects can expose their services through the HTTP RESTful API, which is built on top of the TCP/IP and JSON data formats for the smooth integration of intelligent objects into the Web. The function of the access layer includes how to use the WebSocket to satisfy most of the instant and event-driven IoT use cases.

2.2 Layer 2: Find

To mark an intelligent object as accessible through the Web API does not mean that a client can understand exactly what the intelligent object is and what information or service it provides. This is the main function of the find layer. At this layer, the proposed approach adopts an HTTP-based protocol that consists of a set of resources, data models, payload data syntax, and language extensions. Intelligent Web objects and applications should follow this set of protocols. This layer ensures that a device not only can be used by other HTTP clients but can be searched and used automatically by other WoT applications.

2.3 Layer 3: Share

The WoT allows intelligent objects to deliver data to the Web, enabling more intelligence and big data technologies to be used at the Web level. For instance, data analytics can help manage health or optimize

energy consumption, but only at a sufficient data scale, and when data can be effectively and securely shared across services. The shared layer ensures the efficient and secure sharing of information by intelligent objects on the Web.

2.4 Layer 4: Compose

Intelligent objects connected to the Web (layer 1) can be found by people and machines (layer 2), and their resources safely shared with other devices (layer 3). The compose layer facilitates applications that mix intelligent objects and virtual Web services.

3 Intelligent Web Object API and REST

The representational state transfer (REST) defines the core architecture of the Web. Our WoT-based system provides rules and methods to design a RESTful API for real objects, allowing an HTTP client to easily read data or send control commands to the sensor. However, the REST API implemented via HTTP has certain limitations when an operator needs immediate sensing data and notifications. We discuss how to use Web technologies such as WebSocket to provide an instant notification push functionality to the WoT. Operators can use the RESTful API to model services and materials provided by intelligent objects, making it easy for developers and devices to understand and use them [26–30].

REST is a resource-oriented architecture that regards each component of a system or application, including sensors, sensor sampling frequencies, and variables, as a resource. Resources are not only clearly identifiable but individually addressable. For instance, HTTP is addressed using the standard method of a uniform resource identifier (URI) defined in RFC 39864. Using the same standard naming scheme as other Web resources, operators can fully integrate intelligent objects and their properties into the Web because their functions and data sensors can be connected, shared, and bookmarked.

A URI is a string of characters that clearly indicates abstract or real resources. There are many types of URIs; those mentioned in this work can be recognized on the Web and are located in a network. These are called uniform resource locators (URLs). In this study, the URL of any resource in the WoT must conform to the following syntax:

a. <scheme> “:” <authority><path> [“?” query] [“#” fragment];

b. In the WoT, <scheme> is either HTTP or HTTPS;

c. <authority> indicates the port or access credentials selected by a host;

d. <path> is any hierarchical path to a resource, and it must start with “/”;

e. The last part of the URL represents the selected query parameter or fragment parameter.

Resource identifiers are addressable, portable, and unique in both external and internal networks, and can be resolved by HTTP libraries or tools (e.g., browsers). They can be bookmarked, exchanged in email, and used in instant messaging tools, QR codes, RFID tags, and signal broadcasts.

Resources on the Web are usually organized hierarchically. This is particularly well suited to organize and link resources in the real world because it can identify the resources and relationships of intelligent objects, and distinguish a relationship between an intelligent object and its real position. The Web system platform can identify a collection of resources, which is also a resource. Fig. 4 illustrates the URL parsing of intelligent objects, and presents the URL of an intelligent object and its various parts.

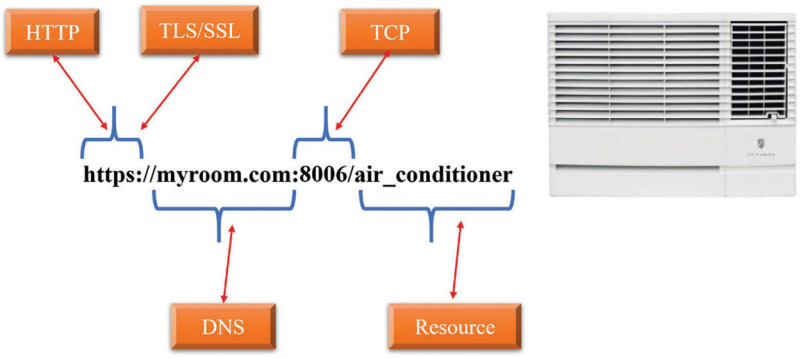

Figure 4: URL parsing of intelligent objects

4 MQTT-Based WoT

The content of the message queueing telemetry transport (MQTT) protocol is streamlined and suitable for IoT devices with limited processor resources and network bandwidth. Many MQTT libraries have been developed in recent years. The use of the Arduino control board (C/C++), JavaScript (Node.js, Espruino control board), Python, and the open-source MQTT server can simplify the development of machine-to-machine (M2M) communication in the MQTT IoT. Facebook instant messaging is also based on the MQTT protocol. The underlying layers of the MQTT and HTTP protocols are TCP/IP, which means that IoT devices can use existing network architectures and devices, but the message format and application processing mechanisms are different. Fig. 5 compares the MQTT and HTTP protocols.

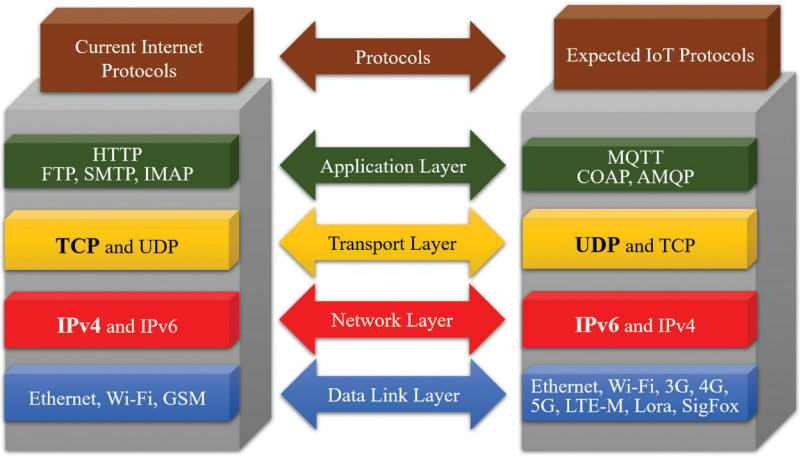

Figure 5: Comparison of MQTT and HTTP communication protocols

According to the version specification, the MQTT represents a client-server publish-or-subscribe messaging transport protocol. The MQTT message sender is equivalent to a magazine publisher, and the MQTT can be regarded as a mechanism for magazine distribution and subscription [31,32]. After publication, the magazine is not sent directly to the consumer; instead, it is handed to a dealer or broker to manage its distribution and subscription. Each source of information has a unique subject name.

The agent represents server software. A publisher represents the party that sends a subject to the server, and the party that receives the subject from the server is a subscriber.

5 Mobile Device Connection to Web Platform

The most straightforward mode by which an operator connects a device to the Web is called a direct integration pattern. This can be used for devices that support HTTP and TCP/IP because a device can directly expose its Web API. This is especially useful when a device can connect directly to the internet via Wi-Fi or Ethernet. Another mode uses the gateway integration pattern; resource-constrained devices cannot connect directly to the internet without a Web protocol but can connect to a more capable gate. The router device exposes the REST API associated with the gateway with the device access. This is particularly useful for devices that support only Bluetooth or ZigBee or have limited resources to accept HTTP requests. In the proposed method, the gateway is changed to a software mode to unify the new standard interface.

The first step in implementing the intelligent Web object API is to design the interface directly on an intelligent object, as shown in Fig. 6. This requires an intelligent object to be accessible via the IP (TCP/IP) and have a built-in HTTP server. The commonly used Web server has no system size difference in many applications. Many small devices can easily run as servers. Also, some implementations of micro-HTTP servers need less than 50 bytes of RAM to run, including TCP/IP stacking, which means that cheap and versatile 8-bit devices can also use the HTTP protocol. Such a Web server can externally provide some REST APIs to access external resources.

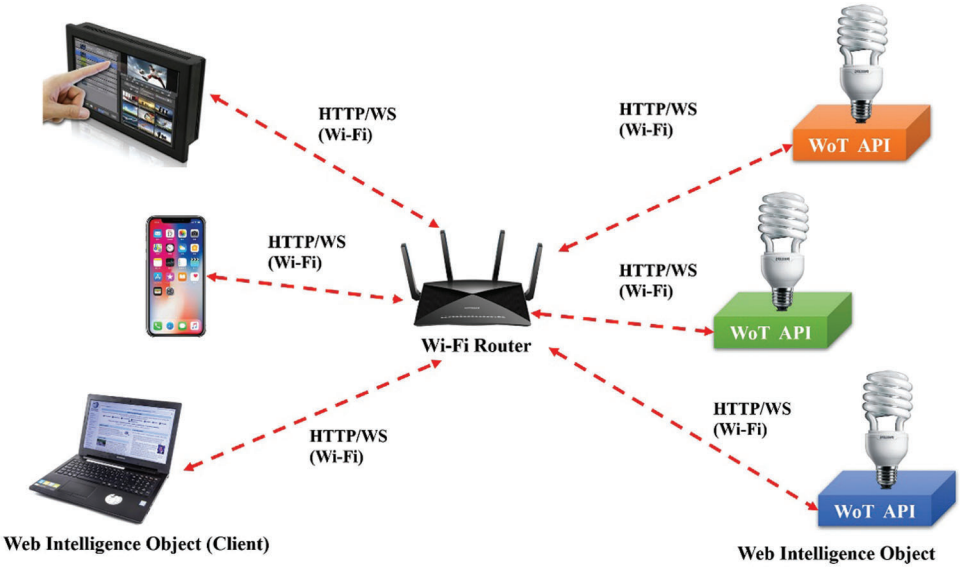

Figure 6: Direct integration of intelligent Web objects and system interface

As shown in Fig. 6, using direct integration mode, intelligent Web objects are connected via Wi-Fi to the lights, and they run the HTTP server inside, so they can directly provide the WoT API. This allows clients of intelligent Web objects, such as mobile applications, to communicate directly with the lights via HTTP. Although the Web protocols can be implemented in most embedded devices, direct integration is more suitable for devices that are not battery powered. The direct integration mode is a good choice when a device must be able to allow direct client access through a mobile communication network, and when a device is powered by a utility.

6 Web Design Framework Using Node.js

Express is not only a Web server but a complete design framework that can meet almost all the requirements of modern Web applications, including the RESTful APIs, HTML, and CSS template engines, database connectors, cookie management, and even social networking site integration. Express has a large developer community, and it has a variety of plugins. Express runs smoothly on Raspberry Pi and most Linux devices. However, Express was not designed to implement Web APIs for IoT devices, so designing Web APIs via Express is challenging. The proposed method can be used to set up an Express of the WoT server. The WoT server folder structure is shown in Fig. 7. In this study, NPM instructions are used to install Express and reproduce this directory structure.

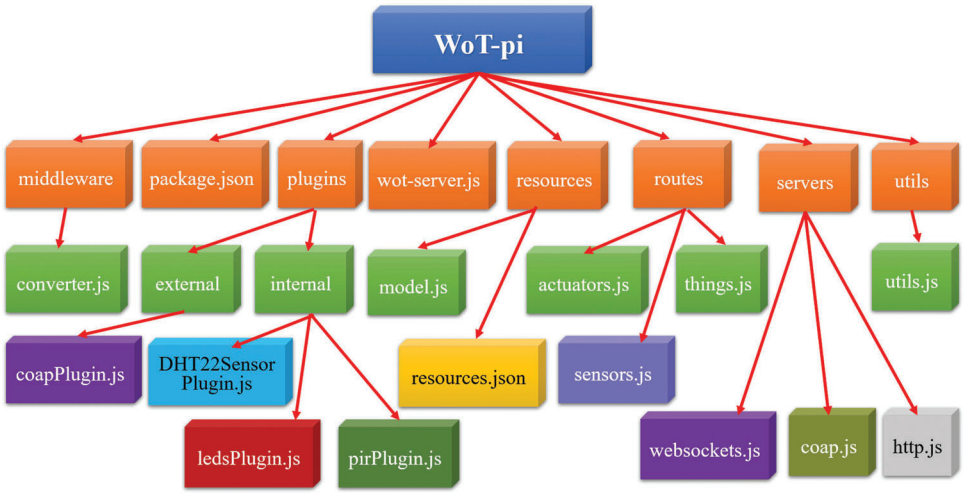

Figure 7: WoT server folder structure

In terms of resource design, the proposed approach plans actual resources to be used on a device, and maps them to the REST resources. As shown in Fig. 8, the hardware includes an LED, passive infrared sensor (PIR), temperature sensor, humidity sensor, and other sensors.

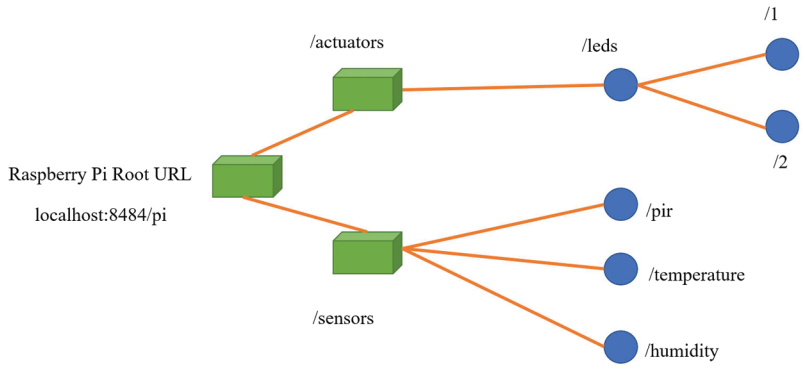

Figure 8: Setting the paths of various devices

A resource tree of sensors and actuators of many Raspberry Pis represents a hierarchical structure. Each resource is represented by a URL path, whose formation is related to the level of the resource tree. For instance, the URL of the passive infrared sensor will be http://localhost:8484/pi/sensors/pir.

In this study, this resource tree is compiled into a JSON file that the system application can use to expose the required URL structure. Then, the resources/model.js file is created, and the JSON model is imported as follows:

(a) var resources = require (‘. /resources.json ’);

(b) module.exports = resources.

The Raspberry Pi exposes the sensors and actuators on the Web to the outside world via the Web API, and accesses them via the REST interface, e.g., http://raspberrypi.local:8484/pi/sensors/pir.

7 System Data Security

Asymmetric encryption generally has a higher cost than symmetric encryption, and the processing capacity and storage space of general sensors are small. Therefore, it is not suitable to perform many asymmetric calculations, such as in the Rivest Shamir Adleman (RSA) algorithm [33]. We use the Rabin asymmetric key encryption algorithm [34] because it is consistent with the low power consumption characteristics of wireless sensor networks. The sensor network architecture is adopted, and Rabin asymmetric key encryption is used for the node sensor to reduce encryption complexity. After a sensor adds a plaintext to the confirmation data, Rabin encryption is used to obtain the ciphertext, which is divided into two data packets that are sent out, and decryption is performed on the server-side.

When the Rabin asymmetric key encryption algorithm is used, the longer the plaintext data the longer it takes to encrypt. The encryption method is given by

$$X = A^2 \bmod B, \tag{1}$$

where X is the ciphertext; A is the plaintext; $B = T \times K$ is the public key; T and K are the private keys, which are composed of prime numbers; $T \bmod 4 = 3$; and $K \bmod 4 = 3$. The squared value of the plaintext is divided by the value of the public key, a remainder is obtained, and this is the ciphertext generated after encryption. In this study, a private key with a size of about 512 bits is used, thus generating a public key with a size of about 1024 bits. This method is more suitable for sensor information encryption.

After the system server receives the complete ciphertext X sent by the sensor, the system can calculate four sets of plaintexts A1–A4 using private keys T and K and public key B. The decryption formula is as follows.

(i)

$$Y_1 = X^{(T+1)/4} \bmod T \tag{3}$$

$$Y_2 = T - X^{(T+1)/4} \bmod T \tag{4}$$

$$Y_3 = X^{(K+1)/4} \bmod K \tag{5}$$

$$Y_4 = K - X^{(K+1)/4} \bmod K \tag{6}$$

(ii)

$$e = K \times \left(K^{-1} \bmod T\right) \tag{7}$$

$$f = T \times \left(T^{-1} \bmod K\right) \tag{8}$$

(iii)

$$A_1 = (e \times Y_1 + f \times Y_3) \bmod B \tag{9}$$

$$A_2 = (e \times Y_1 + f \times Y_4) \bmod B \tag{10}$$

$$A_3 = (e \times Y_2 + f \times Y_3) \bmod B \tag{11}$$

$$A_4 = (e \times Y_2 + f \times Y_4) \bmod B \tag{12}$$

Among $A_1, A_2, A_3,$ and A_4, only one solution is equal to the original A.

In Rabin's asymmetric key encryption algorithm, the ciphertext is resolved by the four sets of plaintexts. When the correct plaintext is found, the confirmation data can be added to the original text. The proposed method uses one letter from the original plaintext as comparison data and appends it to the original plaintext to form a new plaintext for encryption, as shown in Fig. 9.

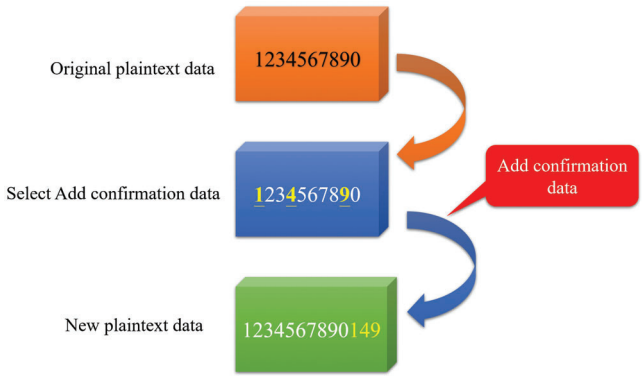

Figure 9: Adding confirmation data

Similarly, when the ciphertext is reduced to the plaintext, the system can verify the correct plaintext by comparing the data, as shown in Fig. 10. If there are A letters in the original plaintext, from which B letters are selected for comparison, the probability of an accurate result of one or more plaintext verifications by the proposed method is $10^{(A-B)}/10^{A}$. Therefore, when the system selects more bits of confirmation data, many different plaintext verifications return to the original plaintext, and the probability of the same occurrence of the original plaintext is very low.

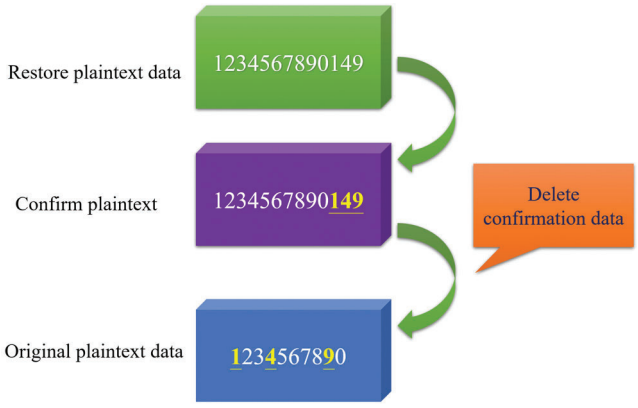

Figure 10: Verification confirmation data

8 Comparison of WoT and Traditional Methods

We compare the WoT and traditional methods in the execution of data records, which originate from information obtained by sensor components. The traditional method processes both the data format and the data itself, which requires significant time. In the proposed WoT method, various data formats can be processed in real-time using software integration, and the results posted on a webpage of monitoring results. WoT technology enables data formats of communication protocols to be redefined in advance, so the proposed method takes less time than traditional methods to process data. Another advantage is better system performance, and the data do not require conversion by various platforms. Sensing data sent from the far end are instantly displayed on the network, and there is no need to convert data formats. Fig. 11 compares the processing time of the WoT technology-based method and traditional methods [35,36].

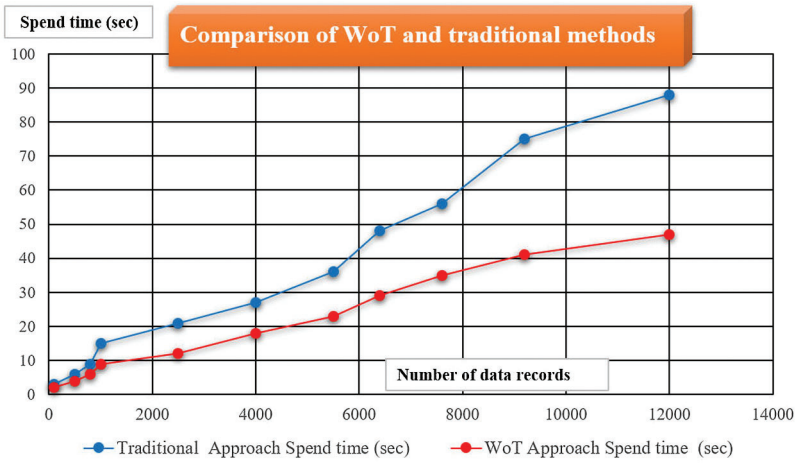

Figure 11: Data processing time of WoT technology-based method and traditional methods

With WoT technology, the Wi-Fi protocol is used directly to connect the Wi-Fi signal ranges of the entire area in a point-to-point manner to form a global Wi-Fi signal range. This enables the connection of local Wi-Fi clouds into a wide-area Wi-Fi cloud. Since the Arduino Yún microcontroller includes a Linux operating system, the Web-server platform is already installed, which makes the network powerful [37]. The proposed network system architecture consists of several local clouds that form a single global cloud. Each local cloud can be regarded as a relay station of the Wi-Fi wireless network. Each Wi-Fi wireless relay station allows operators to login, and a relay station has wireless signals that enhance Wi-Fi throughout the network [38].

The range of Wi-Fi signals that an operator can use is large. Measurement experiments were conducted on an experimental rice cultivation farm. Preliminary experiments were conducted in the laboratory. The wireless sensor network of the entire farming field is presented in Fig. 12.

The sensing data of temperature, humidity, and illuminance were simulated and uploaded to the cloud database via the internet. The LCD in Fig. 13 shows that the WoT system is uploading the temperature, humidity, and illuminance sensing data to the cloud database, which was first established on Google. Fig. 14 shows the real-time upload of the cloud database. Tools provided by Google were used to draw the historical data analysis diagrams of temperature and humidity. In Fig. 15, the blue curve represents temperature, and the red curve represents the humidity. The illuminance can be added and analyzed with the temperature and humidity, as shown in Fig. 16, which presents the integrated historical data, where the orange curve denotes illuminance data.

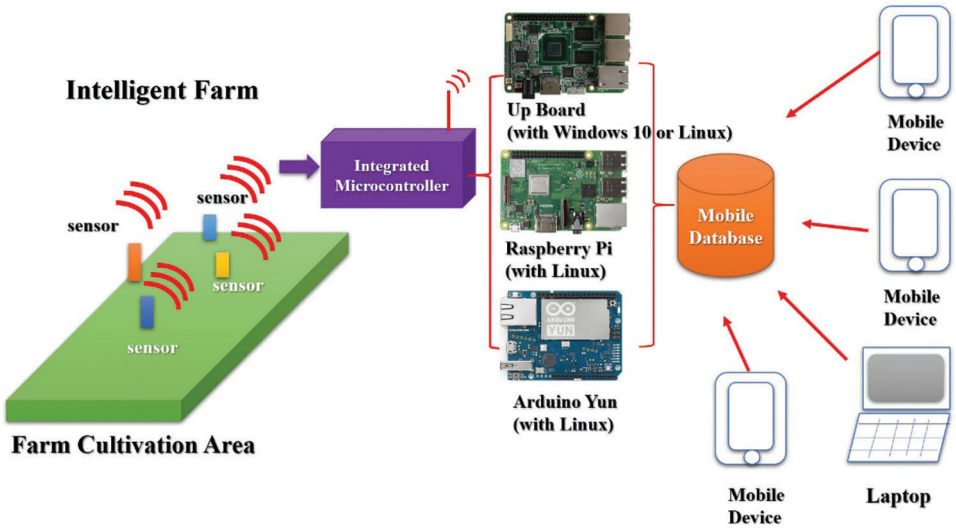

Figure 12: Wireless sensing network architecture of intelligent farm

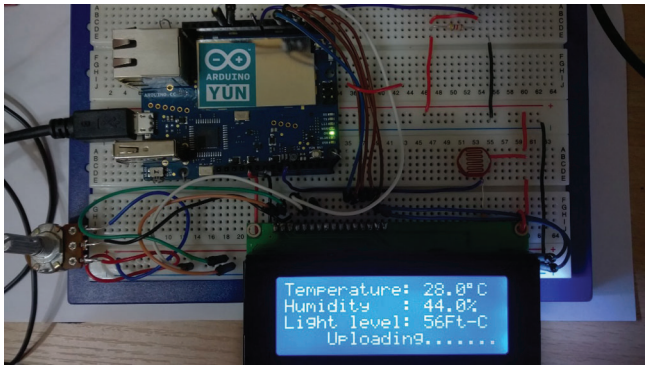

Figure 13: Simulated WoT system uploading temperature, humidity, and illuminance data to cloud database

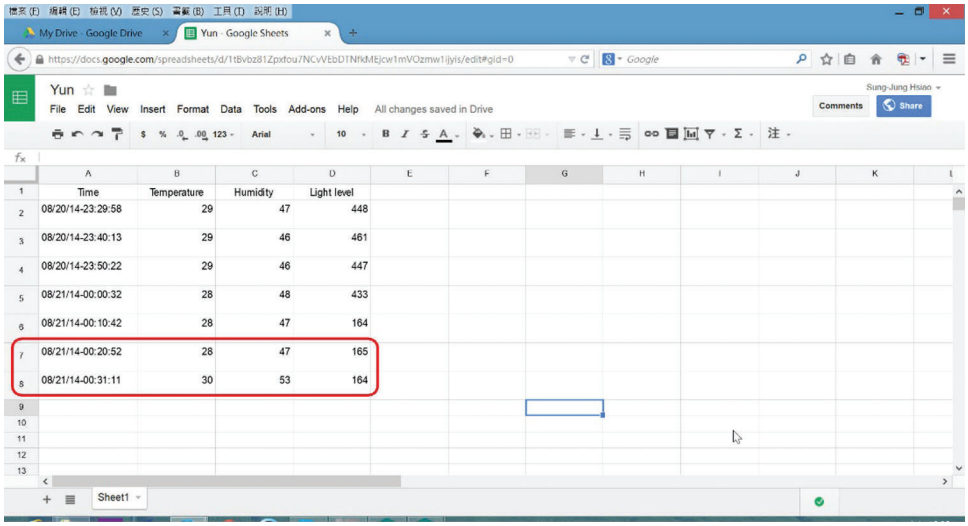

Figure 14: Cloud data table with real-time data upload

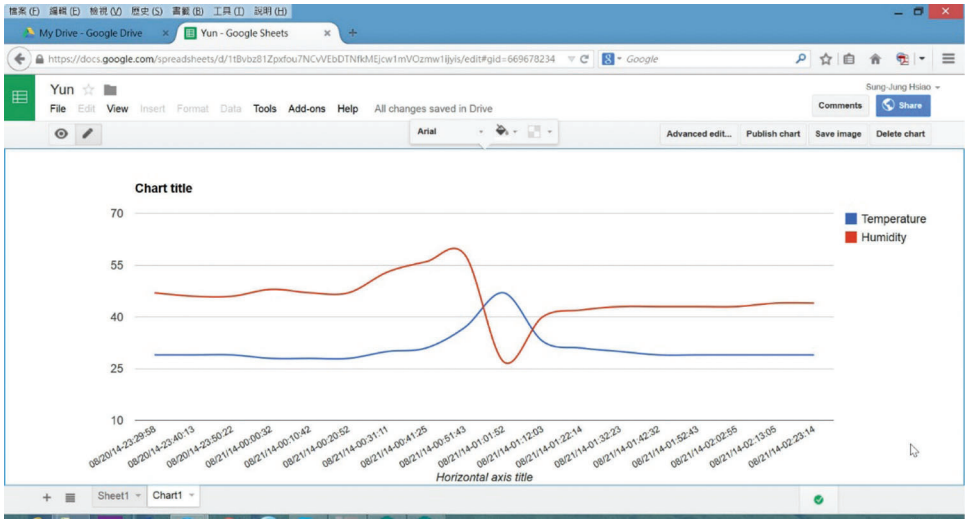

Figure 15: Historical data analysis diagram

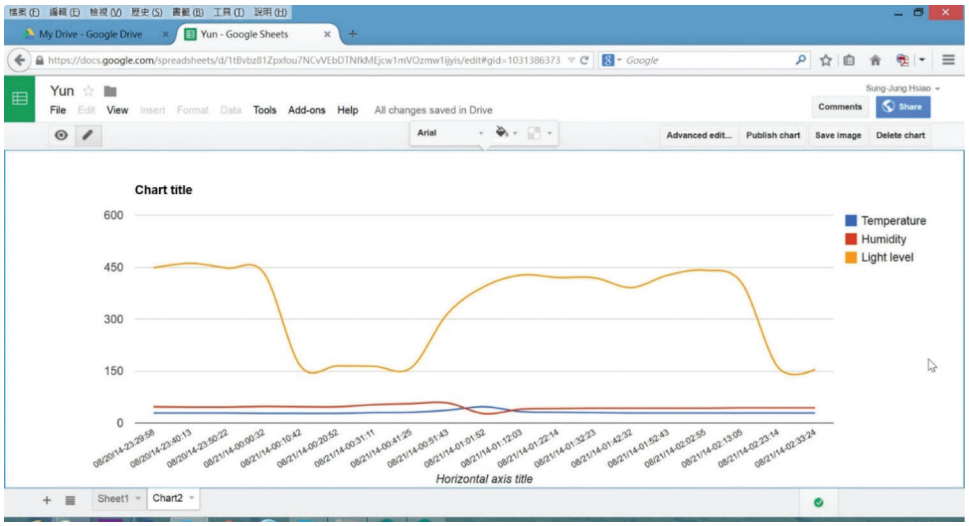

Figure 16: Analysis of historical data

The WoT system simulated in the laboratory could input commands from a browser to a mobile device with any operating system. As shown in Fig. 17, the eighth pin of the digital control hardware was set to zero, which was low. In Fig. 18, the eighth pin of the digital control hardware was set to 1, which was high.

9 Experimental Results

The technology of wireless sensor network integration was used to conduct various farmland environmental sensing analyses of rice growth. The measured data included information on the temperature and humidity in the air, the illuminance of the light, ultraviolet light, average and maximum wind speed, accumulated rainfall per hour, temperature and humidity of the soil, and power curves of the sensor battery. The microcontroller and sensor hardware are shown in Fig. 19. The positions of the sensor and base station in the experiment are presented in Fig. 20. Thirty sensor nodes were placed around the rice fields. The proposed system used the MySQL database system. The farmland sensor device

transmitting the data to every node in real-time is presented in Fig. 21, where the farmland observation data could be immediately accessed by logging into the system on a mobile device. The measured results showed that the daily temperature and humidity changes were relatively small, and the temperature and humidity of the soil changed due to irrigation. Ultraviolet light varied with the sunrise time. The experiment focused on the correct delivery of remote-sensor data.

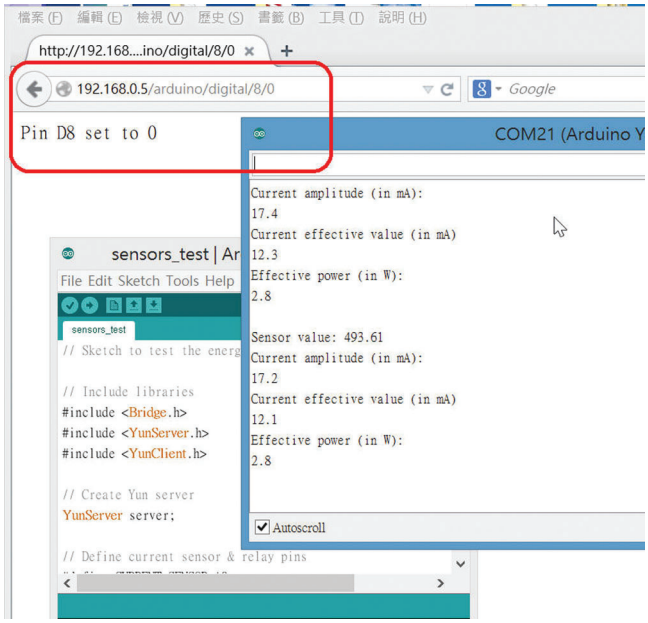

Figure 17: Eighth pin of digital control hardware set to zero

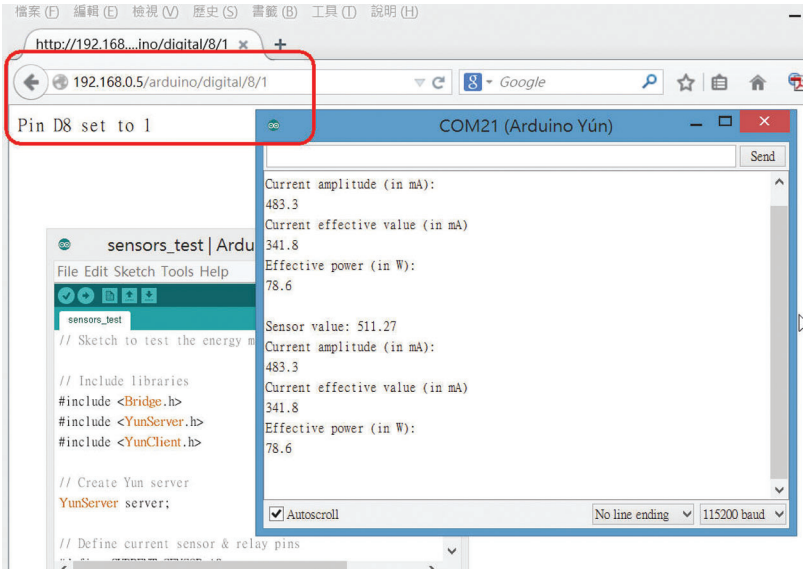

Figure 18: Eighth pin of digital control hardware set to 1

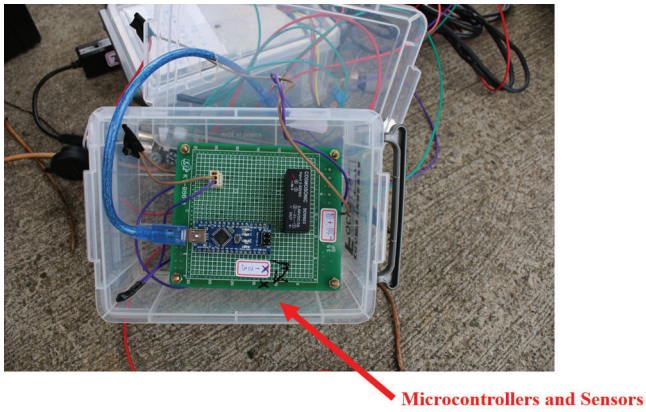

Figure 19: Microcontroller and sensor device in experiment

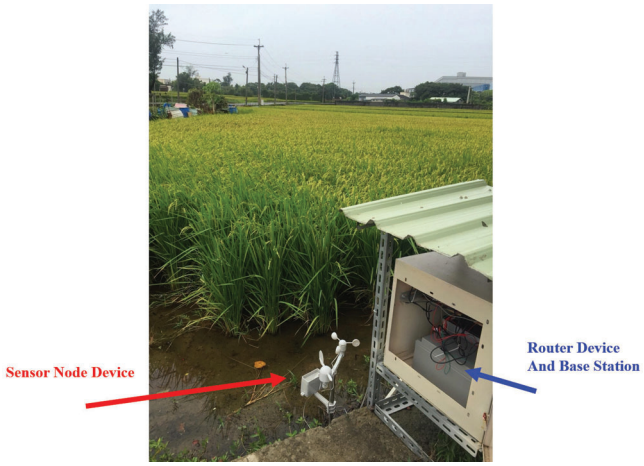

Figure 20: Positions of sensor-node and router devices in experiment

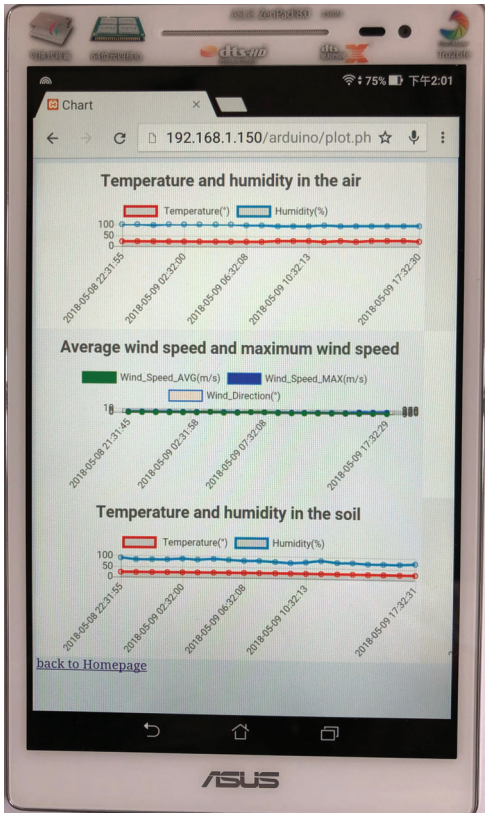

Figure 21: A mobile device displaying a sensing curve in real time

10 Conclusions

In current systems based on IoT technology, objects are connected to a local area network, but due to differences in communication protocols, they represent isolated parts that cannot share data. There are several reasons for this problem:

a. IoT protocols and standards are highly fragmented;

b. At present, many systems use IoT technology applications, and their network attributes are relatively closed applications;

c. IoT applications use various communication protocols, such as ZigBee and LoRa. Many developers hope to use a single communication protocol to unify the applications of the entire network world through IoT applications, but they have not yet achieved this;

d. Communication protocols focus on technology at the expense of application layer protocols.

Integration of WoT technology using the communication standard protocol of the Web can solve the fragmentation problem of the IoT. In the World Wide Web, HTTP is used for interaction between devices, XML and JSON for structured data description, TLS for transmission security protection, and OAuth and JWT for authentication. These are mature technologies whose standards are recognized by the industry, and the global internet also uses them.

We presented a WoT approach that can process various data formats in real-time using software integration and post the results on a Webpage. WoT technology redefines data formats of communication protocols beforehand, and thus consumes less time than traditional methods to process data. The technology also performs better, and the data format requires no conversion between platforms.

The contributions of this study can be summarized as follows:

a. Our research team programmed an application to overcome the problem of packet data returned by different communication protocols. The sensing signals of the wireless sensing network are integrated into the Web platform.

b. The method uses the HTTP network address to monitor and control remote hardware devices.

In the proposed system, various sensors and hardware devices on the remote-end can be assigned separate IPs, and the user can login to the system using a browser for interactive control. The system can be controlled remotely using any internet-enabled device, with no limitation on the operating system.

Acknowledgement: This research was supported by the Department of Electrical Engineering, National Chin-Yi University of Technology, Taiwan. The authors thank the National Taipei University of Technology, Takming University of Science and Technology, Taiwan, for financially supporting this research. We thank LetPub (www.letpub.com) for linguistic assistance during the preparation of the manuscript.

Availability of data and materials: Data sharing is not applicable to this article, as no datasets were generated or analyzed in the current study.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Y. Wang, Y. J. Hsu and S. J. Hsiao, "Implementing a mobile web server with the features of cross-platform and high-efficiency based on WoT with an innovative approach," in *2018 Int. Symposium on Computer, Consumer and Control (IS3C)*, pp. 334–337, 2018.

[2] R. Want, W. Wang and S. Chesnutt, "Accurate indoor location for the IoT," *Computer*, vol. 51, no. 8, pp. 66–70, 2018.

[3] K. Li and J. Nabrzyski, "Virtual machine placement in cloudlet mesh," *Journal of Communications and Networks*, vol. 20, no. 3, pp. 266–278, 2018.

[4] D. Zhu, M. Shafique, M. Lin and S. Pasricha, "Guest editorial: Special issue on low-power dependable computing," *IEEE Trans. on Sustainable Computing*, vol. 3, no. 3, pp. 137–138, 2018.

[5] J. Liu, C. Fang and N. Ansari, "Request dependency graph: A model for web usage mining in large-scale web of things," *IEEE Internet of Things Journal*, vol. 3, no. 4, pp. 598–608, 2016.

[6] A. Saifullah, Y. Xu, C. Lu and Y. Chen, "End-to-end communication delay analysis in industrial wireless networks," *IEEE Trans. on Computers*, vol. 64, no. 5, pp. 1361–1374, 2015.

[7] J. S.-Garcia, S. F.-Castell, J. J. P.-Solano, M. Cobos and J. M. Navarro, "Low-cost alternatives for urban noise nuisance monitoring using wireless sensor networks," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 836–844, 2015.

[8] S. Joardar, A. Chatterjee and A. Rakshit, "A real-time palm dorsa subcutaneous vein pattern recognition system using collaborative representation-based classification," *IEEE Trans. on Instrumentation and Measurement*, vol. 64, no. 4, pp. 959–966, 2015.

[9] M. H. K. Sampaio, A. L. G. Modesto, E. G. A. Sobrinho, J. F. Almeida and O. A. Chase, "SELVABOT-1: Embedded control system using bluetooth technology," *IEEE Latin America Trans.*, vol. 12, no. 8, pp. 1404–1409, 2014.

[10] W. Xue, L. Wang and D. Wang, "A prototype integrated monitoring system for pavement and traffic based on an embedded sensing network," *IEEE Trans. on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1380–1390, 2015.

[11] M. M.-Vazquez and M. L.-Nistal, "A monitoring system to ease self-regulated learning processes," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 10, no. 2, pp. 52–59, 2015.

[12] H. Khamfroush, D. E. Lucani, P. Pahlevani and J. Barros, "On optimal policies for network-coded cooperation: Theory and implementation," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 2, pp. 199–212, 2015.

[13] S. Chakraborty, S. K. Singh and K. Kumar, "Facial biometric system for recognition using extended LGHP algorithm on raspberry pi," *IEEE Sensors Journal*, vol. 20, no. 14, pp. 8117–8127, 2020.

[14] S. Huang, L. Li, H. Cai, B. Xu, G. Li *et al.,* "A configurable WoT application platform based on spatiotemporal semantic scenarios," *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 1, pp. 123–135, 2017.

[15] Q. Zhang, H. Zhong, J. Wu and W. Shi, "How edge computing and initial congestion window affect latency of web-based services: Early experiences with baidu?," in 2018 IEEE/ACM Symposium on Edge Computing (SEC), pp. 393–398, 2018.

[16] N. Sahni, J. Bose and K. Das, "Web APIs for internet of things," in *2018 Int. Conf. on Advances in Computing, Communications, and Informatics (ICACCI)*, pp. 2175–2181, 2018.

[17] Y. Yao, "The WoT framework for the vehicle-to-grid (V2G) implementation," in *2014 IEEE Conf. and Expo Transportation Electrification Asia-Pacific (ITEC Asia-Pacific)*, 2014.

[18] K. Kuromiya, H. Suzuki, K. Naito and A. Watanabe, "Proposal for private address-type WoT server using ntmobile technology," in *2020 IEEE Int. Conf. on Consumer Electronics (ICCE)*, 2020.

[19] L.-L. Chen, "Context-aware technology research in WoT business environment," in *2015 Seventh Int. Conf*. on Measuring Technology and Mechatronics Automation, 2015.

[20] F. Antoniazzi and F. Viola, "Building the semantic web of things through a dynamic ontology," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10560–10579, 2019.

[21] M. R. Faheem, T. Anees and M. Hussain, "The web of things: Findability taxonomy and challenges," *IEEE Access*, vol. 7, pp. 185028–185041, 2019.

[22] L. Mainetti, V. Mighali and L. Patrono, "A software architecture enabling the web of things," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 445–454, 2015.

[23] M. Zacharie, S. Fuji and S. Minori, "Rapid human body detection in disaster sites using image processing from unmanned aerial vehicle (UAV) cameras," in *2018 Int. Conf. on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, vol. 3, pp. 230–235, 2018.

[24] S. K. Datta, J. L. Dugelay and C. Bonnet, "IoT based UAV platform for emergency services," in *2018 Int. Conf. on Information and Communication Technology Convergence (ICTC)*, pp. 144–147, 2018.

[25] B. Klotz, S. K. Datta, D. Wilms, R. Troncy and C. Bonnet, "A car as a semantic web thing: Motivation and demonstration," in *2018 Global Internet of Things Summit (GIoTS)*, 2018.

[26] S. K. Datta and C. Bonnet, "Advances in web of things for IoT interoperability," in *2018 IEEE Int. Conf. on Consumer Electronics-Taiwan (ICCE-TW)*, 2018.

[27] S. K. Datta, M. I. Khan, L. Codeca, B. Denis, J. Härri *et al.,* "IoT and microservices based testbed for connected car services," in *2018 IEEE 19th Int. Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, pp. 14–19, 2018.

[28] S. E. Jaouhari and A. Bouabdallah, "A privacy safeguard framework for a WebRTC/WoT-based healthcare architecture," in *2018 IEEE 42nd Annual Computer Software and Applications Conf*., vol. 02, pp. 468–473, 2018.

[29] R. P. Vargas, M. R. Hortelano, L. T. Abad, J. C. Carrillo and R. H. Berlinches, "Teaching cloud computing using web of things devices," in *2018 IEEE Global Engineering Education Conf. (EDUCON)*, pp. 1738–1745, 2018.

[30] I. Nadim, Y. Elghayam and A. Sadiq, "Semantic discovery architecture for dynamic environments of web of things," in *2018 Int. Conf. on Advanced Communication Technologies and Networking (CommNet)*, pp. 1–6, 2018.

[31] B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: A survey," *IEEE Access*, vol. 8, pp. 201071–201086, 2020.

[32] A. Detti, L. Funari and N. B.-Melazzi, "Sub-linear scalability of MQTT clusters in topic-based publish-subscribe applications," *IEEE Trans. on Network and Service Management*, vol. 17, no. 3, pp. 1954–1968, 2020.

[33] R. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[34] T. Shang, R. Chen and Q. Lei, "Quantum random oracle model for quantum public-key encryption," *IEEE Access*, vol. 7, pp. 130024–130031, 2019.

[35] A. Gyrard, S. K. Datta and C. Bonnet, "A survey and analysis of ontology-based software tools for semantic interoperability in IoT and WoT landscapes," in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pp. 86–91, 2018.

[36] S. Javaid, H. Afzal, F. Arif and N. Iltaf, "Trust management for SOA based social WoT system," in *2018 20th Int. Conf. on Advanced Communication Technology (ICACT)*, pp. 387–392, 2018.

[37] B. C. B. Solomon and S. M. Aral, "WOT based routing protocol for the home diagnostic instruments, in 2017 Third Int," *2017 Third Int. Conf. on Science Technology Engineering & Management (ICONSTEM)*, pp. 429–431, 2017.

[38] A. Sengupta, E. R. Kumar and N. P. Chandra, "Embedding digital signature using encrypted-hashing for protection of DSP cores in CE," *IEEE Trans. on Consumer Electronics*, vol. 65, no. 3, pp. 398–407, 2019.