

Distributed IoT Technology based on Light Application Development

Chunqiang Li, Xiaoxia Cui

Institute of VLSI Design, Zhejiang University, Hangzhou, China.

Abstract

Based on the research of various IoT communication technologies and the analysis of the working principle of JavaScript execution engine, a scenario-based distributed technology architecture is proposed. Based on this, virtual bus, security authentication and dynamic object model transformation based on JavaScript language are innovatively implemented and applied in smart home scenarios. It is proved that the scenario-based distributed IoT technology improves the convenience of IoT device development, real-time response and security of device access.

Keywords

IoT; Scenario; Distributed Technology; Security Authentication; Object Model.

1. Introduction

With the continuous improvement of chip, radio frequency, communication protocol and other technologies, IoT application has gradually evolved from single product cloud application to scenario-based application [1]. For example, in the scenario of smart home, there are some central devices with strong interaction such as smart speakers, smart TV sets and smart refrigerators. Meanwhile, Surrounding these central devices, there are executive or sensing node devices such as lights, sockets, thermometers, water heaters, etc., which need to be interconnected so as to bring convenience to family life. There are also many such scenarios in our work and life: smart campus, smart storage, smart agriculture, etc.

IoT scenario-based applications bring convenience to customers by providing ubiquitous real-time response to customer through local interconnection between devices, but it requires more complex technical support than a single IoT product connected to the cloud. However, most of the current scenario-based application implementations just simply connect devices in a certain degree. Through the study of mainstream scenario-based application implementations, we have found that the current technologies have the following shortcomings: (1) lack of security when the devices are locally connected, and the devices are easy to be cracked by hackers; (2) lack of unified IoT object model and communication standard, and devices from different IoT platforms cannot interconnect with each other. For example, IoT products from AWS and Xiaomi cannot work together. (3) lack of unified development interface because of the diversity of IoT devices, developers need to master all application knowledge from system software to system hardware. These problems are the bottleneck restricting the development and promotion of IoT scenario-based applications.

In this paper, we briefly introduces the JavaScript language and execution engine, and explains the BLE Mesh protocol working principle [2], then to solve the shortcomings of the current scenario-based application technology mentioned above, we propose and implement a scenario-based distributed framework with light application, and the framework is combined with innovative technologies, such as the distributed security authentication and dynamic object model transformation with light JavaScript execution engine, so that those devices in scenario-based application can communicate each other smoothly and security, at the same time, it simplifies the scenario-based application development. Finally, the feasibility of the technology is verified by a series of experiments in practical scenarios.

2. JavaScript Execution Engine

JavaScript is an interpreted programming language, and it uses object-oriented weakly typed variable types to make programming easier. Because of this simplicity, JavaScript is currently the most popular language for developers. Unlike the compiled and executed approach of the C/C++ programming language, JavaScript programs are interpreted line by line at the runtime, making it having platform independent and requiring only a JavaScript execution engine.

The JavaScript execution engine is responsible for interpreting and executing JavaScript script code. In order to adapt to the constantly improving requirements of web browser and server performance, JavaScript has developed from the early interpretation execution to the JIT dynamic translation execution, such as the Google-led V8 engine. In the embedded system, in order to adapt to the diversity of the embedded system, many JavaScript engines open source have emerged. These engines, limited by hardware resources, still use interpretation execution, such as Duktape, QuickJS, JerryScript and so on. The implementation efficiency of JavaScript execution engine directly affects the execution efficiency of JavaScript code, as shown in Figure 1. Generally, JavaScript execution engine is composed of syntax parser, AST interpretation executor [3] or JIT translator, stack management and garbage collection, etc.

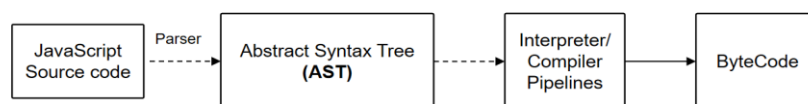


Figure 1. JavaScript Execution Engine

3. Distributed MESH Network [4]

Compared with star network, distributed Mesh network has several advantages: the scalability of network transmission distance, the convenience of deployment, and the fault-tolerance of network device nodes, etc., which can provide fast network access for devices in IoT scenario-based applications. At present, the mainstream implementations of distributed Mesh network include BLE Mesh, ZigBee, etc. As shown in Fig. 2, taking BLE Mesh as an example, devices in the network are generally divided into Relay Node, Low Power Node, Normal Node, Friend Node, etc. All device nodes in the network communicate through publishing and subscribing messages to achieve user functions.

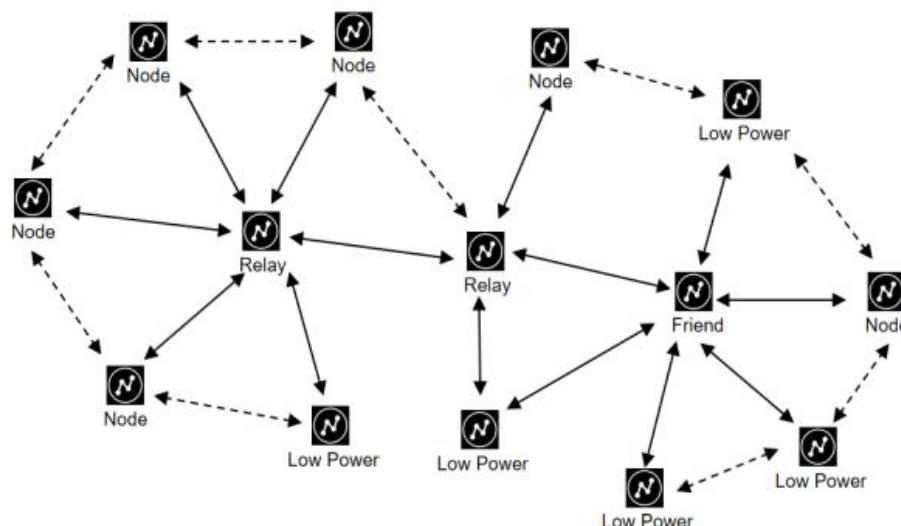


Figure 2. BLE MESH

4. Scenario-based distributed technology

In view of the existing problems of IoT scenario-based application technology mentioned in Chapter 1, we propose a light application scenario-based distributed technology architecture based on JavaScript and BLE Mesh, and implements innovative technologies such as virtual bus, distributed security authentication and dynamic object model transformation on this architecture.

Figure 3 is the architecture diagram of scenario-based distributed technology with light application proposed in this paper. The virtual bus is responsible for abstracting BLE Mesh, Zigbee, WiFi and other underlying communication protocols, so that the application development of IoT device can be protocol independent. Distributed security authentication guarantees the secure access and communication of each device in the scene. The JavaScript execution engine is responsible for the abstraction of the device application programming interface, and the transformation of the object model, so that different object models between devices in the scene can be interconnected.

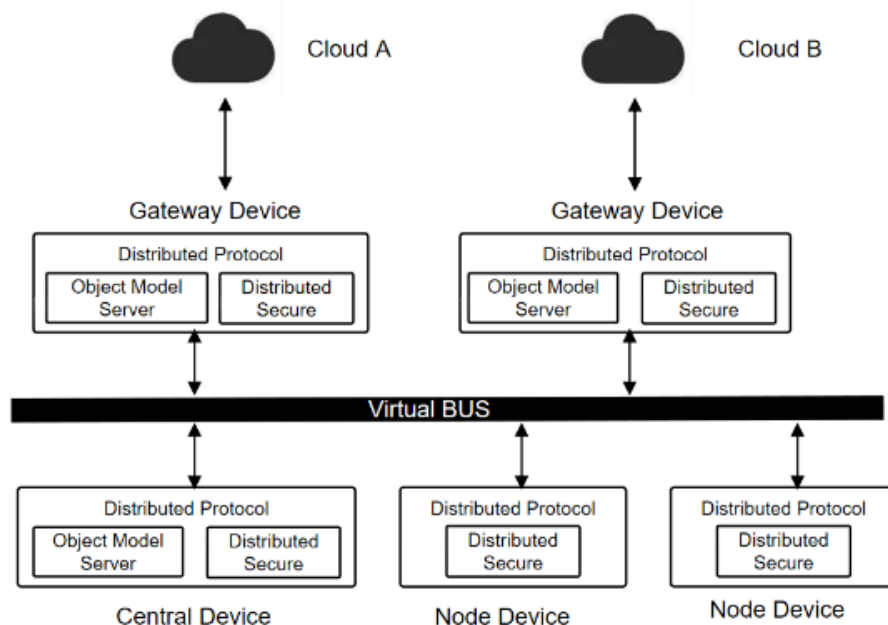


Figure 3. Scenario-based Distributed Technology Architecture

The scenario-based distributed technology with light application proposed in this paper consists of a series of IoT devices (also known as node) with different functions in the actual deployment which interconnect, transmit data and control between each other securely and freely. Devices can cooperate freely to realize the overall functions of the scene together according to their own ability, so as to realize the user requirements of the actual application scene. With scenario-based distributed technology, the user can manage and control IoT devices without mobile phone or computer as center control equipment, and IoT device with UI ability can directly control other devices in the scene, such as smart home scenario, the user can directly turn on the water heater in the bathroom through the intelligent voice panel in the bedroom.

There are typically several IoT device roles in each scenario: central device, gateway device, and node device. There can be multiple devices for each role in the scene, and a device can be multiple roles. The central device is responsible for device management, service management and object model transformation; the gateway device is responsible for cloud platform access, security authentication and conversion and communication between different communication protocols within the scene; the node device is responsible for information collection, command output and action execution. Figure 4 is a distributed schematic diagram of smart home as an example.

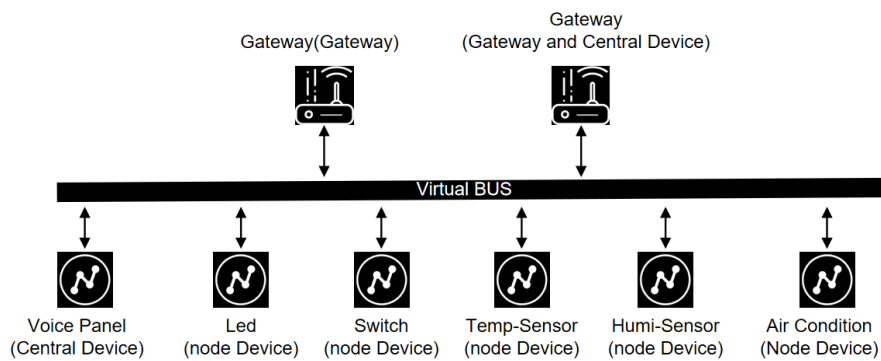


Figure 4. Distributed schematic of smart home scenario

4.1. Distributed Virtual Bus

The diversity of IoT applications results in the diversity of connection and communication protocols. The scenario-based distributed technology must consider the access and interconnection of devices with various communication protocols. For example, the interconnection between BLE Mesh and ZigBee devices, and between WiFi and BLE Mesh devices requires gateway devices, and we need to abstract the different communication protocols to promote the convenience of development. As shown in Figure 5, we innovates to realize the virtual bus mechanism to abstract BLE Mesh and WiFi, and reserve extended interfaces to facilitate the subsequent support of ZigBee or other protocols. The distributed virtual bus consists of basic communication mechanisms such as device discovery, service discovery, networking and connection sessions.

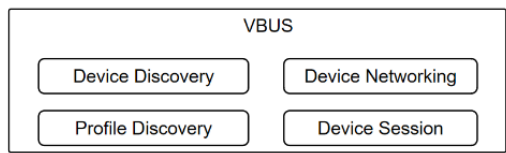


Figure 5. Distributed Virtual Bus

As shown in Figure 6, when a new device appears in the scene, it will broadcast a message "New Device", the gateway device will capture the message "New Device", and finish Authorization and networking for the new device through the distributed authentication mechanism in section 4.2. Once the new device is connected successfully, it will register itself and its services in the center device through the gateway device in order to subscribe to the message from other devices, so that its own function can be called by other devices.

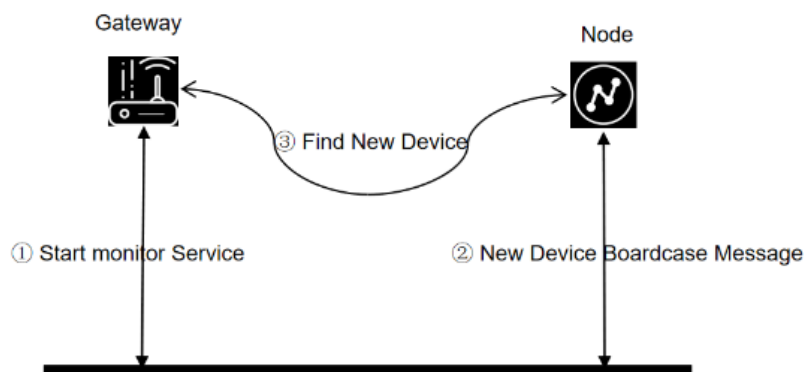


Figure 6. Virtual Bus's new device discovery process

4.2. Distributed Authentication

Through the distributed virtual bus, gateway devices can dynamically discover, network and connect new devices in the scene, which cannot be separated from security guarantee. In the implementation of existing IoT technology, the access of new devices is generally authenticated through cloud platform [6], which requires the cooperation of mobile phones, etc., resulting in poor real-time performance and troublesome operation for users. The distributed authentication technology, implemented in the gateway device, provides the scene local identity authentication methods for new device networking and accessing scenario when the scene finding it. Compared to IoT traditional authentication, The distributed authentication is provided with fast speed and short delay in the case of no less security. Figure 7 shows the specific flow of the implementation and execution mechanism of distributed authentication.

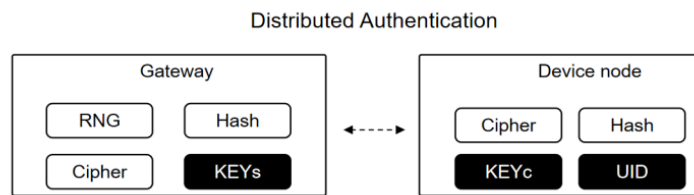


Figure 7. Distributed Authentication Mechanism

1) Preset KEYS when gateway device produced.

2) Calculate and generate KEYc according to Formula (1) and Preset KEYc when node device produced.

$$\text{KEYc} = \text{HASH} (\text{KEYs} \mid \text{UID}) \quad (1)$$

3) Calculate KEYc' locally according to Equation (2) after the gateway device gets the UID from the node device.

$$\text{KEYc}' = \text{HASH} (\text{KEYs} \mid \text{UID}) \quad (2)$$

4) The node device obtains the Challenge (Nonce) from the gateway device and calculates the RSP locally according to Formula (3), and sends the RSP to the gateway device

$$\text{RSP} = \text{ENC}(\text{KEYc} \mid \text{Nonce}) \quad (3)$$

5) The gateway device calculates the RSP' locally according to Equation (4).

$$\text{RSP}' = \text{ENC} (\text{KEYc}' \mid \text{Nonce}) \quad (4)$$

6) The gateway device compares whether the RSP' is the same as the RSP to determine whether the device authentication is approved.

After the new device authentication is passed and networked in the scene, it will register itself in the Shadow Device Management of the central device in the scene through the gateway device, so as to realize the management of the physical node device. In Figure 8, the new device can also publish its own service profile as needed, so that other devices can easily call the new device's service remotely through the central device.

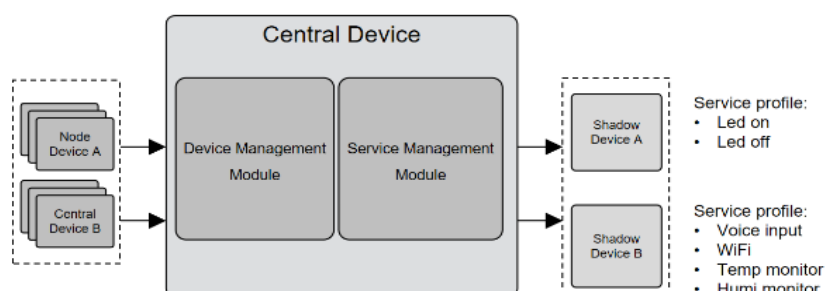


Figure 8. Device Management and Service Discovery in central device

4.3. Dynamic Object Model Transformation Based on JavaScript

At present, IoT platforms are separated from each other, which makes devices from different IoT platforms impossible to interconnect with each other. For example, the IoT products from AWS and Xiaomi cannot be networked, which has greatly restricted the development of IoT scenario-based applications. This paper proposes and implements the Dynamic Object Model Transformation mechanism [7] based on JavaScript. With the characteristics of simplicity and high efficiency, it can be adapted to different IoT platforms [8] and support different IoT devices, and the development of object model transformation can separate from the development of device system due to the use of JavaScript language for development.

As shown in Figure 9, the dynamic model transformation implemented in the central device consists of the Basic Object Model, the Device Management Module, Service Management Module, JavaScript Execution Engine and Model Transformation Application in JavaScript. The Basic Object Model in the central device is used to describe capabilities for devices in scene, such as the 'Turning on', 'Turning off' action to light device, and it can be extended dynamically through the network. The Device Management Module is responsible for discovering new devices through the Virtual Bus, and adding and deleting devices in the Shadow Device Management; The Service Management Module is responsible for discovering new services of devices through the virtual bus, and adding and deleting services in the corresponding Shadow Device; The JavaScript Execution Engine is responsible for interpreting the application code that acts as the Object Model Transformation in JavaScript; Object Model Transformation Application in JavaScript is developed by IoT device manufacturers, and used to transform between their own device object model and the basic object model, and downloaded into the Central Device when the device discovered and networked.

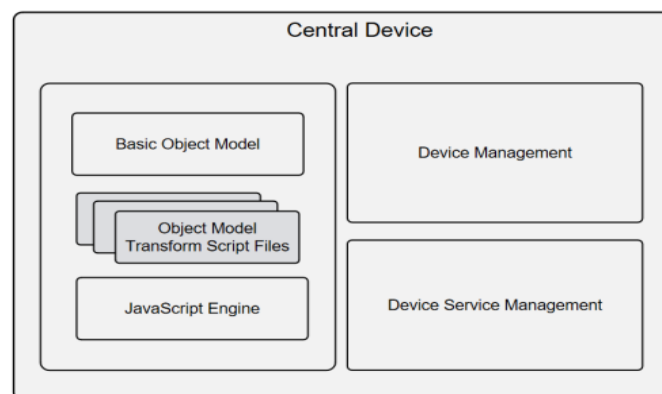


Figure 9. Dynamic Object Model Conversion Module

As shown in Figure 10, The process of the dynamic object model transformation implemented in this paper includes the development stage, the discovery stage and the operation stage.

As shown in Figure 10 (a), the IoT device manufacturer realizes the JS application to transform object model from the device one to the basic one in the device development stage, and registers the device object model and JS application in the cloud platform of the object model.

The discovery stage is shown in Figure 10 (b). When the new device is discovered by the central device in the scene, the central device automatically downloads the corresponding JS object model transformation application from the object model cloud platform and saves it in the corresponding Shadow Device.

The operation stage is shown in the example in Figure 10 (c). The scene includes central device, voice panel (developed by manufacturer A with JS-A application), and electric lamp (developed by manufacturer B with JS-B application). First, when the voice panel recognizes the lamp 'Turning on' command, it sends the lamp 'Turning on' command in object model of manufacturer A to the virtual bus in the scene. Secondly, when the central device receives the lamp 'Turning on' command in

object model of manufacturer A from the virtual bus, it executes the corresponding JS-A application through JavaScript to convert the lamp 'Turning on' command in object model of manufacturer A into the lamp 'Turning on' command in basic object model. Thirdly, the central device analyzes that the target device of the command as the lamp, then finds the corresponding JS-B and executes it. The lamp 'Turning on' command in basic object model is converted into the lamp 'Turning on' command in the object model of manufacturer B, and sent to the lamp, thus 'Turning on' service of the lamp is call successfully.

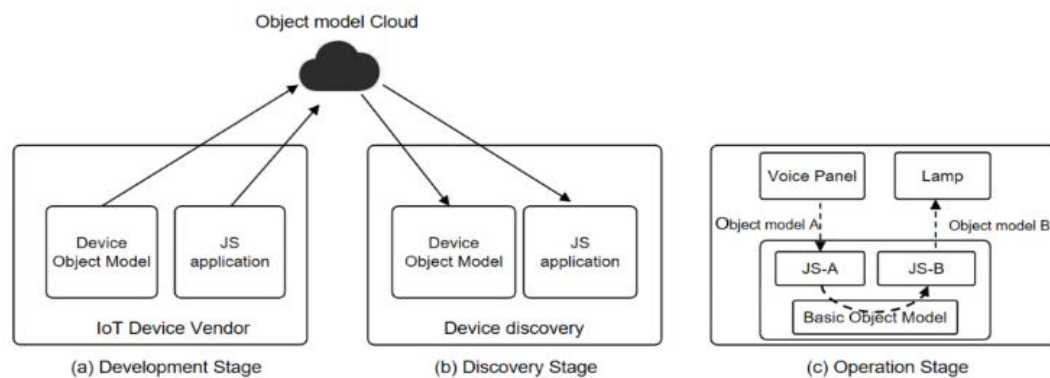


Figure 10. The Process of Dynamic Object Model Transformation

Through the dynamic object model transformation based on JavaScript language proposed in this paper, IoT device manufacturers do not need to care about which device from which manufacturers in the specific scenario-based application, but only need to pay attention to their own device development and object model definition, and use JavaScript language to carry out simple object model transformation application development.

5. The Experimental Results

To verify the scenario-based distributed technology with light application, We have implemented an experimental scene in this paper, which consists of the following devices : (1) 128 Mesh node devices based on PHY6220 Bluetooth chip, such as lamps, switches, sensors, etc.; (2) Smart voice panel node device based on W800 WIFI /BLE dual-mode chip; (3) Gateway device and central equipment based on W800 WIFI /BLE dual-mode chip; (4) Cloud service.

In the experiment, (2) the smart voice panel can issue commands, and (3) the central device can effectively control the lamp after the object model transformation. The experiment proves the feasibility of the scenario-based distributed technology in this paper. At the same time, the experiment also conducted statistics on the real-time performance in the scene. As shown in Table 1, the distributed technology based on light application significantly improves the real-time performance in the scene.

Table 1. Real-Time experimental result

Average Cost(ms)	Distribution Control	Master Device Control
Mesh device control between Mesh node devices	50	560
Mesh device control from the cloud service	100	610

6. Conclusion

This paper focuses on the pain points of the implementation of IoT scenario-based application and proposes the IoT distributed technology based on light application development, it also implements

the distributed device security authentication and the dynamic object model transformation, simplifies the development of object model transformation application by JavaScript language, as well as we make the innovative technology compared with the traditional IoT application through experiments in the real-time performance and running performance of the scene application. In the future, we can further optimize the performance of JavaScript execution engine, adapt to more cloud platforms and object models of IoT device, and promote and apply it in the industry continuously.

References

- [1] C-IoT Cloud-Based Services and C-IoT User Device Diversity, F Behmann,K Wu, John Wiley & Sons, Ltd. 2015(07)
- [2] BLE Mesh: A Practical Mesh Networking Development Framework for Public Safety Communications, Bo Zhang Yufeng Wang Li Wei Qun Jin Athanasios V.Vasilakos Nanjing University of Posts and Telecommunications Nanjing 210023 China Department of Human Informatics and Cognitive Sciences Waseda University Saitama 359-1192 Japan Department of ComputerScience Electrical and Space Engineering Lulea Universityof Technology Sweden. 2018(03)
- [3] From Bytecode to JavaScript: the Js_of_ocaml Compiler, Jerome Vouillon,V Balat 10.1002/ spe. 2187. 2014(01)
- [4] Fairness-oriented routing algorithm joint with power control and channel assignment for multi-radio multi-channel wireless mesh networks[J]. LIU Kai-ming,MA Tao,LIU Yuan-an,KOU Ke-hao. The Journal of China Universities of Posts and Telecommunications. 2014(05)
- [5] A wireless solution for greenhouse monitoring and control system based on ZigBee technology, ZHANG Qian YANG Xiang-long ZHOU Yi-ming WANG Li-ren GUO Xi-shan School of Biosystem Engineering and Food Science. Zhejiang University. Hangzhou 310029 China, 2007(10)
- [6] ID-based authentication scheme combined with identity-based encryption with fingerprint hashing, JIANG Wei-qiang HUANG Zheng-quan YANG Yi-xian TIAN Jie LI Liang Information Security Center State Key Laboratory of Networking and Switching Technology Beijing University of Posts and Telecommunications Beijing 100876 China Key Laboratory of Complex Systems and Intelligence Science Institute of Automation Chinese Academy of Sciences Beijing 100080 China.2014(04)
- [7] A Fog-based Semantic model for supporting Interoperability in Internet of Things, H Rahman,MI Hussain, 10.1049/iet-com.2018.6200. 2019(01)
- [8] IoT cloud platform for information processing in smart city, Xiangcong Chen, Shuqing Zhang, Xiaohui Ding, Seifedine Nimer Kadry, Ching-Hsien Hsu, 2020(8)