Research and Implementation of Data Providing Platform Based on Internet of Things

Huan Gao[1, a], Bo Cheng[1, b]

[1] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China.

[a]18910316062@163.com, [b]chengbo@bupt.edu.cn

Keywords: Internet of things, data providing platform, push service, publish/subscribe system, load balance.

Abstract. With the development of Internet of things, multi-source heterogeneous data of a large number of sensors exist in the Internet of things. How to refine and encapsulate these heterogeneous data is one of the bottlenecks in the development of the Internet of things. In this paper, the data platform uses the SpringMVC framework as the basis. The data providing platform extracts the sensor data from the database to encapsulate and transmit, which can reduce the coupling between the application layer and the underlying layer. The data providing platform provides HTTP access interfaces for the upper layer applications, the lightweight web based server push service, and publish/subscription system for multiple types of terminals. The platform not only provides multiple dimension data interfaces for the upper layer applications, but also has a high scalability. The nginx server is used to achieve the load balance, which can expand the number of servers flexibly.

1. Introduction

With the rapid development of the Internet, the Internet of things also entered a period of rapid development. China wants to seize the opportunity to develop the Internet of things. The Internet of things industry is regarded as a strategic emerging industry in China. The government invested a lot of manpower and material resources to promote the study of the core technology of the Internet of things. In 2009, Chinese Premier Jiabao Wen put forward the concept of "perception of China" [1]. The research of the Internet of things will become the focus of information technology in the future. Internet of things is gradually forming the industrial chain in China, which involves transportation, medicine, environment monitoring and so on.

Internet of things is connected with the Internet through the sensor equipment to collect information, which achieves the connection between machine and machine, machine and people. The Internet of things architecture is divided into three levels, which are the perception layer, network layer and application layer. Sensing layer uses the sensors to collect data. The network layer transmits the data through the Internet. The application layer is to analyze and process the collected data, and dig out the valuable information for decision-making. The core of the Internet of things is the flow of data. How to transfer data between machine and machine, machine and people is the main point for the Internet of things. In the current architect of Internet of things, the data collected by the sensors are mostly heterogeneous and multi-source. Different sensors collect data with great differences. Once the underlying data changes, all of the upper layer applications need to make a change. There exists high coupling between the application layer and the underlying data layer. This paper aims to realize the data providing platform for application layer, which can provide unified data interfaces. The data is processed and refined by data platform. Data providing platform can provide multi-dimensional service interfaces for the upper application.

2. The Framework Design and Implementation of Data Providing Platform

The perception layer of Internet of things stores the data in the different tables and databases. In application development, various applications are required to extract data from the different databases, which causes the high coupling between data layer and applications layer. The data providing platform in this paper is to extract the data from different databases, encapsulate the data and remove the redundant data, which is used to transform the heterogeneous multi-source data into the information needed by the upper layer application. The data providing platform is the interface design of the upper layer service. When the underlying data is changed, the data layer of the platform would be modified uniformly. The applications of the upper layer are not necessary to modify the data structure, which greatly improves the scalability and flexibility of the application. The overall framework of the data providing platform is shown in Fig.1. The client can access the data providing platform through Internet. The nginx server receives the request and forwards the data request to the tomcat server for processing. After processing the data, it will return the data to the client in JSON format.

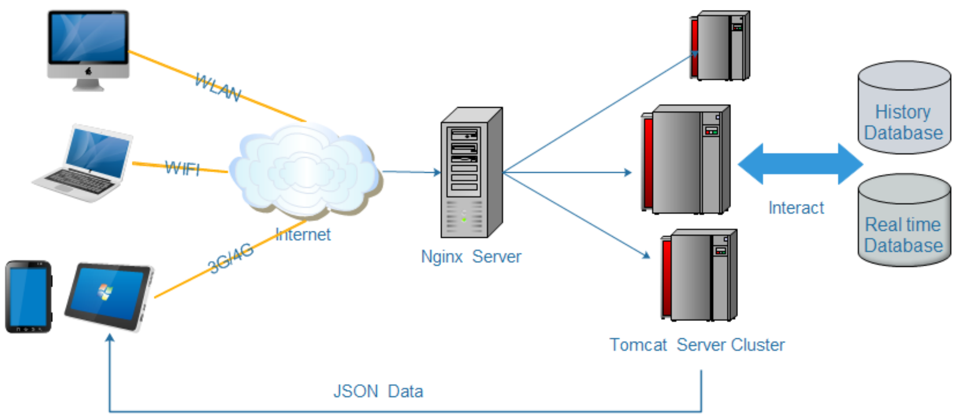


Fig. 1 The overall architecture of data providing platform

2.1 Detailed Design and Construction of the Platform for HTTP Access Interfaces

The data providing platform designed in this paper uses the SpringMVC and Spring framework. SpringMVC is a MVC framework, which can realize the data interface quickly and be able to work perfectly with the Spring framework. The SpringMVC framework separates the roles of the controller, model, dispatcher, and handler objects, which makes it easier to customize and keep the high scalability. When new business logic needs to be realized, it only needs to write the business logic in Controller, and can quickly realize the service composition. SpringMVC framework can directly translate the objects into JSON format through the annotation configuration, which greatly reduces the difficulty of the data encapsulation and makes the combination of services more flexible. The process of handling requests in SpringMVC framework is shown in Fig.2.

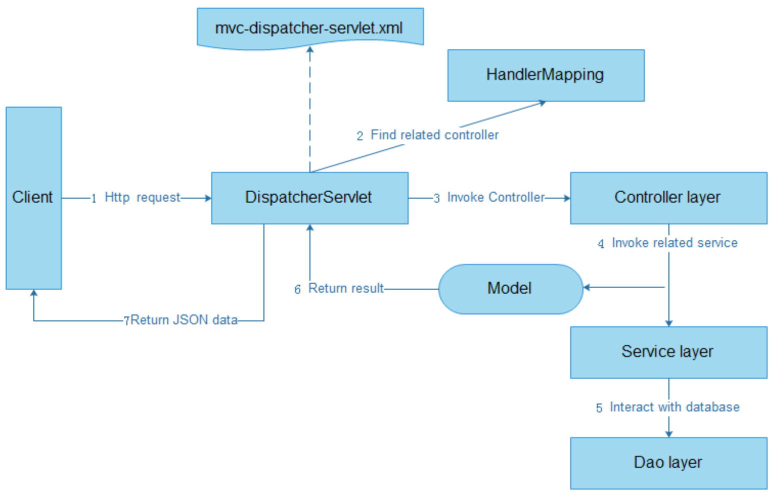


Fig.2 The request processing flow of data providing platform

Data providing platform structure is divided into controller layer, service layer, entity layer, Dao (Data Access Object) layer. Controller layer is responsible for handling all kinds of HTTP request and receive relevant parameters. The service layer is responsible for the implementation of the business logic and the combination of the data extraction and encapsulation. At the same time， the service layer needs to achieve transaction management. In the paper, the transaction management is achieved through the AOP (Aspect Oriented Programming) technology of Spring framework, which can be configured in the XML file. If the operation is just to query data from the database, the transaction configuration can be set as read-only, which can improve the execution efficiency. Entity layer is responsible for object mapping and the establishment of data model. Dao layer is responsible for data acquisition and database communication. Through the Spring injection technology and JDBC technology, the data providing platform and database can achieve the dynamic connection and switch. SpringMVC framework can directly convert the objects to JSON data and return the data to the client through the configuration of the Jackson library. In the controller, the annotation @ResponseBody is used above the detail method code. This method can directly return a Java object, which will be converted to a JSON output by the MappingJacksonHttpMessageConverter. Besides, the GET method or POST method can be selected by configuring the request method in the controller. According to the business requirements, data interface is divided into four main modules, which contains target monitoring module, device information module, communication link module, sensor module. The architecture of each module is shown in Fig.3.

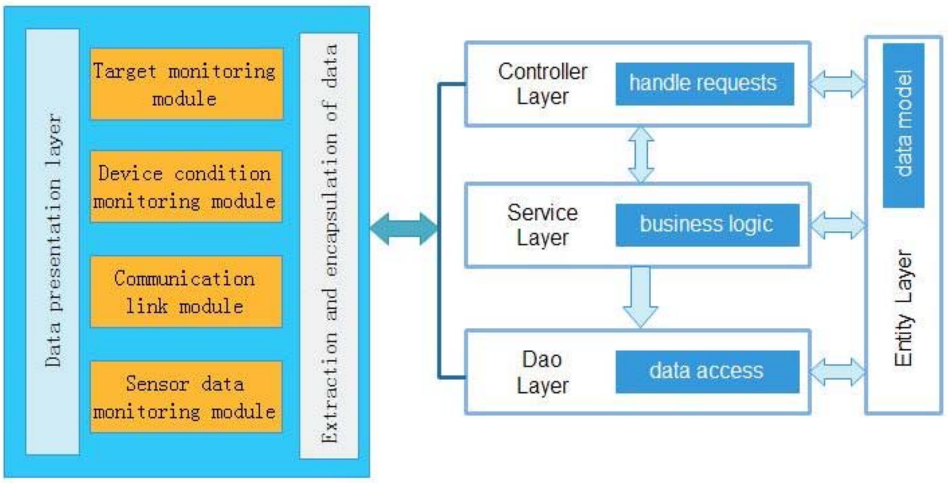


Fig.3 The module partition of data interfaces

The target monitoring module is mainly divided into two parts. One is the real-time monitoring of the target data, tracking the data and encapsulating the target data, which can convert the data into the information needed by the application. Another part is to obtain the historical information of the target. Through the ID (Identification) and the time as parameters, the client can obtain target record. In addition, the query interfaces also provide all the list of targets data within a certain period of time.

In the device module, the device information is obtained from the real time database, which will be associated with the deployment region. The device data will be extracted from the databases and encapsulated into the JSON format data. The client can obtain device information which contains the deployment location, and running status.

Communication link module is to get communication information between devices. Interface information includes the connection of equipment and communication hub. For example, the current sensor through the LTE (Long Term Evolution) base station to transmit data, so the data link will be the sensor ID and LET base station ID as well as the detail information of corresponding sensor and LET. The acquired data is reorganized to form a meaningful JSON data.

Sensor data module is mainly used to transmit the sensor data through a lightweight data push service. The sensor data is pushed to the page in the real time, such as temperature sensors, humidity sensors and so on. In the Internet of things system, there are kinds of sensors. Some sensors will not have the information change for a long time. So the push service is need for the sensor data monitor. When the sensor data change, the client can get the change immediately.

2.2 Lightweight Data Push Service for Web Page

Push service is very necessary in data platform, which can reduce the number of insignificant user access and polling and reduce the pressure on the server. In addition, the user can timely get the data change. The traditional web request mode is the client makes a request first, and then the server will return a response immediately. This model does not meet the requirements of some situation, such as monitoring platform, instant messaging systems and so on. Through the push system, users do not need to refresh the page in real time. Once there is a change of data, the data will be pushed from the server to the client. Comet is a kind of server push technology, which is based on HTTP long connection. If use Comet technology, users do not need to install plug-ins in the browser side. At present, there are three ways to achieve Comet technology, which are AJAX(Asynchronous JavaScript and XML) based long polling, iframe and htmlfile based flow mode [6].

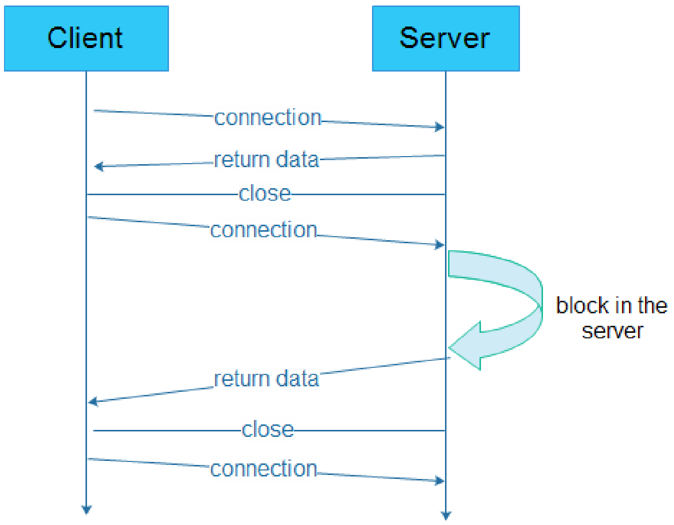


Fig.4 Ajax based long polling communication process

This platform mainly uses the Comet4j framework to realize the push service for the webpages. Comet4j is a pure AJAX based server push framework, which exists long polling, long connection, and automatic selection of three modes operation. Users only need to call the JS function on the page to connect to the push server. The data push process is shown in Fig. 4.The main function in push module is to encapsulate and transmit the sensor data, such as temperature sensor, humidity sensor and so on. Comet4J works in the NIO mode.

2.3 Java Message Oriented Middleware Based On ActiveMQ Framework

JMS (Java Message Service) is a messaging service for Java. The data can be transmitted through the JMS system between clients. In the Internet of things, some data need to be processed before returning to the user. The response time of the client will be increased with the increase of the amount of concurrency, which may lead to server crash. In this case, JMS can be used to realize the asynchronous request processing. After the server receives the request, we can send an order to the message queue, and returns the response to the user immediately. The required data will be sent to the client via the JMS system after the asynchronous processing is completed. In addition, many applications of Internet of things have a lot of heterogeneous software collaboration, which may be developed by different development languages. JMS can achieve the communication between the heterogeneous software. This system mainly uses the ActiveMQ framework to realize the JMS service. ActiveMQ supports a variety of languages to write the client, such as Java, C, C++, Python, PHP, and so on. ActiveMQ also supports a variety of protocols, such as OpenWire, REST Stomp, Notification WS, XMPP, AMQP and so on. ActiveMQ can be easily embedded into the Spring framework and also supports the Spring2.0 version characteristics. So ActicveMQ can be perfectly integrated with Spring framework. The ActiveMQ version of the 5.5.1 framework is selected in this platform.

Through ActiveMQ framework, the platform realizes the multi terminal oriented message push service. The message can be published by the background system and subscribed by heterogeneous client through topics. The publish/subscribe communication system in this platform as shown in Fig.5.

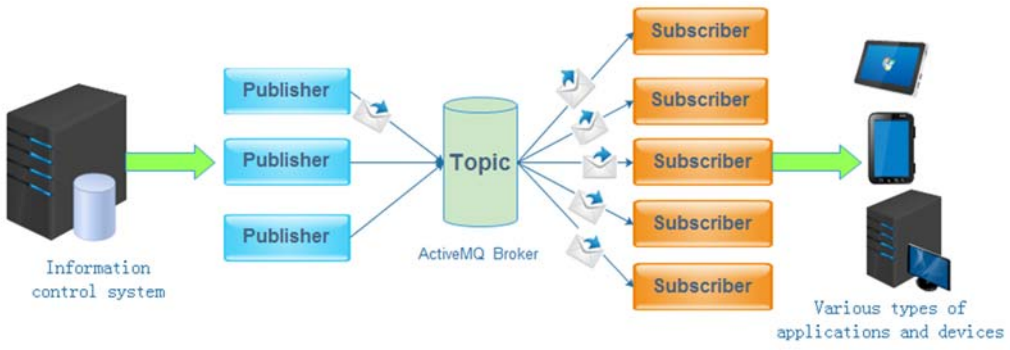


Fig.5 The communication process of publish/subscribe system

Platform can be flexible to configure the theme through the publish/subscribe system, which makes the architecture of the platform more scalable. In the data providing platform, publish/subscribe system is mainly used for the target monitor and shared data transmission. When monitoring target, the publish/subscribe system can push the target information to the clients which subscribe the relate topic. So the heterogeneous upper software applications can realize the linkage of multiple means.

3. Load Balancing for Data Providing Platform

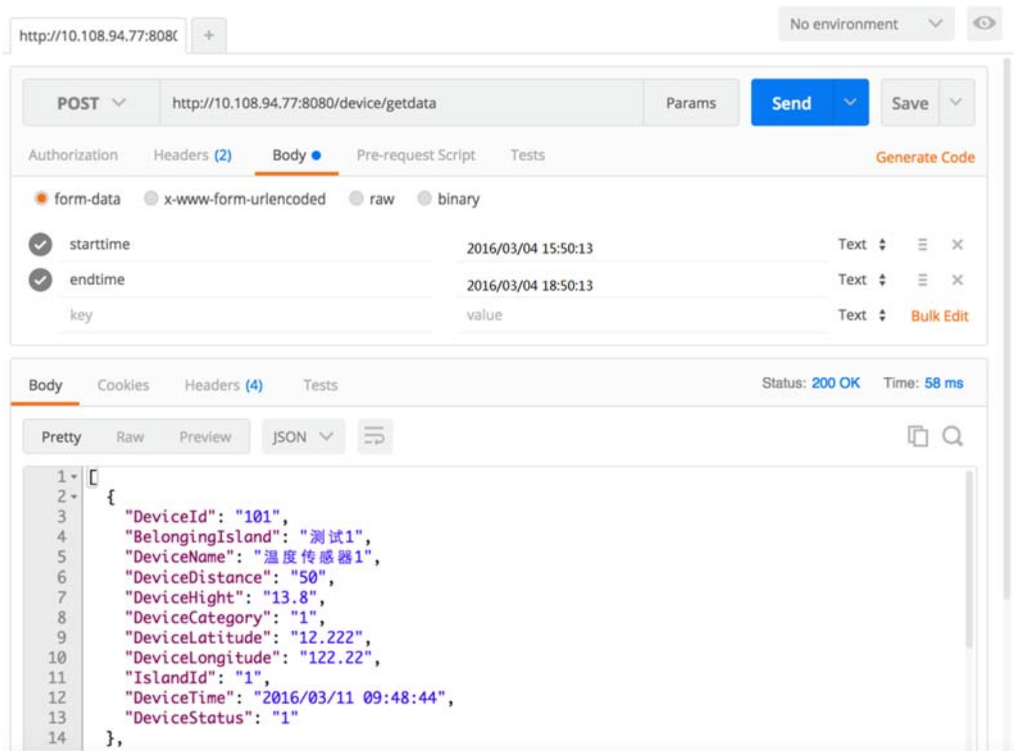
With the increase of interface access and data flow, the processing and computing capacity of the data providing platform of the server are also required to increase, which makes a single server cannot afford. In this case, a single hardware upgrade, which will result in a waste of existing resources, but also face the next business volume upgrade, which will lead to a high cost of hardware upgrades and investment. So the load balancing mechanism must be added to the data platform. Load balancing has two implications. One is that a large number of concurrent access or data flow is shared to multiple nodes to reduce the user waiting time. The other is that a single heavy load shares operation to multiple node devices to do parallel processing. After the end of each node processing, summarize the results and return to the user, which makes the system processing capacity greatly improved. The load balancing technology studied in this paper is mainly to balance the load of tomcat server cluster in high concurrent access.

The load balancing of the data providing platform is realized through nginx server. Nginx not only can be used as a powerful web server, but also can be used as a reverse proxy server. In the case of high connection concurrency, nginx server is a good substitute for Apache server. Nginx server can achieve the separation of dynamic and static page in accordance with the rules, which contains the poll, IP (Internet Protocol) hash, weights and other ways to do a variety of server load balancing [8].

In the HTTP node of nginx configuration file, the IP of load balancing server and the server status can be defined. The weight can be set based on the performance of different servers. The nginx server will forward the request to the server cluster dynamically according to the distribution of the access weight and return the response to the client. The default value of weight is 1. The bigger the weight value the greater the load. Server clusters can be free to expand the number of servers, and delete server dynamically.

4. Interface Test and Performance Analysis

This paper mainly does the function tests and the response time tests for the data interfaces. Functional testing is completed through the black box test. Insert some data to the database, and compare with the results by calling the interface. Testing tools uses Postman, which is chrome plug-in for web page debugging and sending HTTP request. Postman can simulate types of HTTP requests, and can set the required parameters. The correctness of the interface data is tested and determined by the Postman tool. The test result using Postman is shown as Fig.6.


http://10.108.94.77:808(

No environment

POST http://10.108.94.77:8080/device/getdata Params Send Save

Authorization Headers (2) Body Pre-request Script Tests Generate Code

form-data x-www-form-urlencoded raw binary

starttime	2016/03/04 15:50:13	Text
endtime	2016/03/04 18:50:13	Text
key	value	Text

Bulk Edit

Body Cookies Headers (4) Tests Status: 200 OK Time: 58 ms

Pretty Raw Preview JSON

```
[
  {
    "DeviceId": "101",
    "BelongingIsland": "测试1",
    "DeviceName": "温度传感器1",
    "DeviceDistance": "50",
    "DeviceHight": "13.8",
    "DeviceCategory": "1",
    "DeviceLatitude": "12.222",
    "DeviceLongitude": "122.22",
    "IslandId": "1",
    "DeviceTime": "2016/03/11 09:48:44",
    "DeviceStatus": "1"
  },
```

Fig.6 Postman function test result

In addition to testing the correctness of the interface, the response time of the interface is another very important indicator. The response time of each interface is recorded through the log record, and the function of recording time is woven through the AOP technology of Spring framework. Taking the target history interface and device information interface as an example, 10000 track records and 10000 device records are stored in the database. The average response time is shown in Fig.7.

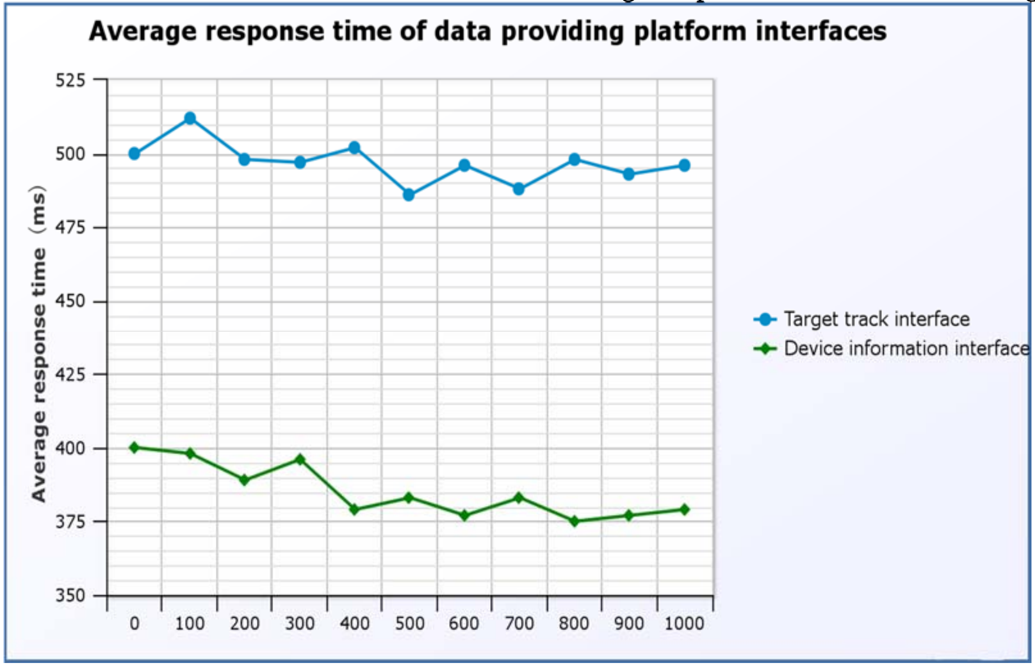


Fig.7 Test results of average response time for data interfaces

5. Summary

This paper mainly introduces the design architecture and implementation method of data providing platform for the Internet of things. With the development of the Internet of things industry, the role of data processing in the Internet of things will become more and more important. How to construct the multi dimension data providing platform is the focus of this paper. In the design of the data providing platform, this paper mainly uses the SpringMVC and Spring framework as the basic framework. At the same time, the lightweight push service and publish/subscribe system are integrated with the Spring framework, which forms a complete platform architecture. The data providing platform is able to provide cross platform interface services for users. The data is in the JSON format so that it is lightweight and easy to parse. The data providing platform can be divided into four layers, which contains controller layer, service layer, entity layer and Dao layer. Each layer has its responsibility, which can realize low coupling and high flexibility. Service layer introduces the transaction mechanism for the insertion modification to ensure the integrity of the operation. In addition, load balancing is achieved by the nginx server in the outer layer of the platform. The data providing platform can dynamically configure multiple servers to ensure the high scalability and the ability to handle high concurrent access.

Acknowledgement

This paper is supported by the National Natural Science Foundation of China under Grant No. (61372115, 61132001), “973” program of National Basic Research Program of China Grant No. 2012CB315802, and National High-tech R&D Program of China (863 Program) under Grant No. 2013AA102301.

References

[1] Meiwen, S.: Theoretical Analysis and Countermeasures Research On The Development of Internet of Things Industry. Jilin University (2015).
[2] Qibo, S., Jie, L., Shan, L., Chunxiao, F., Juanjuan, S.: Internet of Things: Summarize on Concepts，Architecture andKey Technology Problem. Journal of Beijing University of Posts & Telecommunications. 33.3, 1–9 (2010).
[3] Tyagi, S., Darwish, A., Khan, M.Y.:Managing Computing Infrastructure for IoT Data. Advances in Internet of Things. 04.3, 29–35 (2014).
[4] Atzori, L., Iera, A., Morabito,G.: The Internet of Things: A survey. Computer Networks. 54, 2787-2805 (2010).
[5] Hui, Y.: Design and Implementation of Enterprise Work Order Management System Based on SpringMVC and Ibatis. University of Chinese Academy of Sciences (2015).
[6] Mei, L., Xuyang, W.: Study on the Application of WEB in Real Time Based on Comet Technology. Wireless Internet Technology (2016).
[7] Peng, L.: Lightweight Design and Integration on Message Oriented Middleware Based on JMS. Southwest Jiaotong University (2010).
[8] YongHui, W.: The Improvement and Implement of Performance Optimization and Load Balancing In High Performance Nginx Web Server. University of Electronic Science and Technology of China (2015).