Research on Internet of Things (IoT) accessing platforms for multiple applications and scenarios

Yongjie Lee[a], Shidong Liu[b], Yong Yan[c,*], Li Ning[d]

[a]College of Electronic Engineering, Naval University of Engineering, Wuhan, China; [b]Unit 92228 of Chinese People's Liberation Army, Beijing, China; [c]Beijing Jinghang Computation and Communication Institute, The Third Academy of China Aerospace Science and Industry Corporation, Beijing, China; [d]Beijing Institute of Mechanical and Electrical Engineering, The Third Academy of China Aerospace Science and Industry Corporation, Beijing, China

ABSTRACT

This paper proposes a set of IoT connection management and data service platform. The IoT platform can avoid the disadvantages that hardware systems of multiple manufacturers and multiple systems need to be adapted separately. The unified access method of IoT sensing devices is studied to enable the access work of devices to be carried out in a configurable way and improve the efficiency of device access and management. The platform provides powerful data channels for users and help the dual-direction communications of terminals like multi communication between sensors, actuators, embedded devices and intelligent appliances. It also supports concurrent and massive access of devices, million-level messages concurrent processing performance and provides multiple protections of devices and realizes data extraction, storage, processing and integration from multiple IoT devices.

Keywords: Internet of Things (IoT), terminal device, device events

1. INTRODUCTION

Sensor, working as measurement and control terminal, is the bottom of IoT[1]. But currently there are no unified standards and protocols between the terminals from each company. Terminals from different companies do have different communication protocols and accessing methods[2]. This paper proposes a set of IoTs connection management and data service platform (hereinafter referred to as IoT platform) which not only has the ability of unified access and management on different measurement and control terminals[3], but also can provide unified data service for application systems.

2. CHARACTERISTICS OF THE PLATFORM

IoT platform is designed above perception layer and network transport layer, the bottom of the IoT technology system and under the application layer. It works in the middle of the perception layer and the application layer[4]. The platform can deploy independently and complete the access management of the perception layer devices according to specific application requirements. Application data related to industry applications is handled by the application system in the upper layer[5]. Due to the different industries and businesses, different application systems should be applied to realize corresponding business logic analysis and processing; and these application systems belong to different measurement and control terminals, which are all required to access IoTs for unified access and device management, so as data analysis and data services[6].

The main technical characteristics of the IoT platform are

(1) Multi-protocol access: it supports the access from protocols like MQTT, Socket, WebSocket, Coap, Stomp, Modbus DTU, Modbus TCP, which can help and solve the access management of various heterogeneous devices.

(2) IoT security: it can provide safe protection for each link during the data processing from access to the upper layer presented to users and it works as: link security provided by SSL and TLS, data access controlled effectively by key authentication, internet attacks prevented by hardware devices like firewall, data storage security protected by replication

*13370154501@189.cn

International Conference on Computer Application and Information Security (ICCAIS 2021),
edited by Yingfa Lu, Changbo Cheng, Proc. of SPIE Vol. 12260, 122601R · © The Authors.
Published under a Creative Commons Attribution CC-BY 3.0 License · doi: 10.1117/12.2637687

strategy.

(3) High-reliable device management: By self-registration, REST service or adding terminal devices in batches, it uses batch commands to operate and control devices in batches and provides complete device management.

(4) Optimized asset management: It connects devices with staffs from the identity management platform, also connects devices with asset management in the asset management platform or position between geospatial elements, which can realize the whole life cycle management of the devices.

(5) Advanced multi-tenant pattern: It runs any number of IoT applications on a single application example and provides independent data storage system for each tenant, so as independent and customizable event processing engine for each tenant not affecting other tenants when starting, stopping and reconfiguring tenants.

(6) High-concurrent real-time access: It works with million-level IoT terminals, and handles high-frequency data transmission to the system in real time.

(7) Massive data storage and processing: It provides large-scale scalable device events management and cluster deployment scheme, integrates highly adjusted MongoDB and HBase into realization, provides time series database for device events, and solves the problems of, with the increase of the terminals, database's reading and writing requirements increase sharply together with the reading and writing delay increases in traditional architectures.

(8) Device status display: It captures the current status or the past status over a certain time period of one terminal among the million-level IoT terminals.

3. RUNNING ENVIRONMENT INDICES

The platform can be deployed in various environments, such as Windows, Linux, clouds that support Docker environment. After performance indices test, the basic requirements of the environment are operation system: Windows Sever 2012, hardware configuration: 32G memory, 8-core CPU, 100G hard disk.

4. THE TECHNICAL IMPLEMENTATION

4.1. Global modules

The IoT platform is comprised of several different global modules, which are connected with each other to provide core services. All lessees are sharing the global setting. The technical framework is shown in Figure 1.

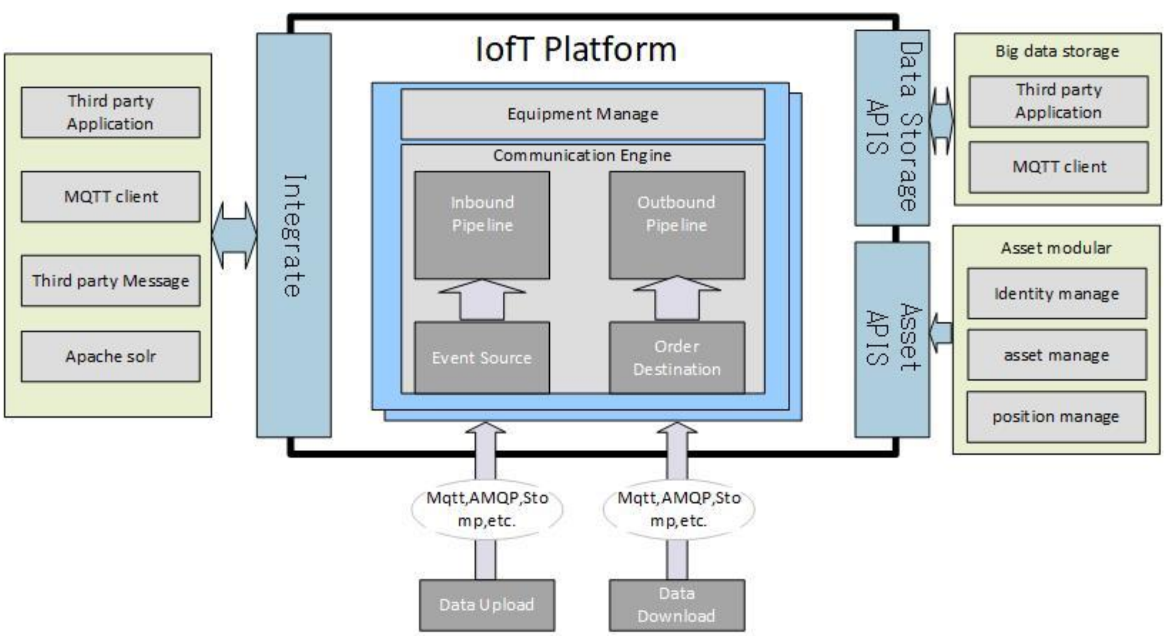

Figure 1. Technical structure.

4.1.1. Web application container. The IoT platform is configured as a Web Application Archive (WAR), and it can run in the web containers[7]. By changing the configuration files, the way how IoT platform deals with device issues and integrates with external service is changed.

4.1.2. Platform server. A server is the central application that control all other modules in IoT platform. It is automatic started in WAR file and uses Spring configuration file in conf/site/site-server.xml to boot. The server is responsible for general system modules management, clients management and REST service.

4.1.3. System management. IoT platform includes HTML5 management application which can be used for managing system functions. Information appeared on the application depends on login credentials and the user authority. In real cases, one administrator with all authorities is in the IoT platform and connects with default clients.

Some aspects in this system, such as the clients and lessees, are manages in global scope. The changes in these entities will be reflected in all lessees. Since these users can create new users and lessees, and shut down running lessees, data in the system will be influenced if current users are deleted. Thus, the administrative access in this level should be restricted.

The most aspects in this system are lessee oriented. Each user account can be connected with one or more lessees and is permitted to manage several lessee information. While logging into the system, a user connected with several lessees will be asked to select the lessees to be managed before processing. All other system data is managed in lessee level, thus, the majority data in this management interface is depending on the user that logs in and the lessees that user selects[8].

4.1.4. REST service. Most core functions that related to IoT platform API can be externally accessed via REST service. By using REST SERVICES, external entities can create, retrieve, update and delete entities that in the system. Also, it can interact with assets management sub-system. Authentication is a necessity for calling REST and Spring Security is adopted to verify if the user has corresponding authority.

A formal version Swagger is included in the IoT platform and it adds a user interface around REST service. By using Swagger interface, the running IoT platform server can execute REST API alternatively and check JSON responding. The default Swagger URL of server is http://hostname/servername/, where hostname is for the running IoT platform server.

4.1.5. Global data storage. When store and retrieve data, IoT platform does not deal with database. Conversely, the system defines service provider interface (SPI) for the data manipulation of the third party, and expects that data storage implementations satisfy all interfaces. User management data is stored under global-level configuration and based on following API, IUserManagement: including all cure user management API, CRUD of users and authorities.

While configurating a new IoT platform server example, by changing the settings in core Spring configuration file, the specific data storage for storage implementations is defined. Currently, data storage in MongoDB and Apache HBase are supported.

4.1.6. Hazelcast service. Hazelcast is an in-memory data grid (IMDG). Its design involved in high performance. In defaut situation, each Server example can act as Hazelcast example. In default global configuration, Hezelcast is loaded from conf/servername/hazelcast.xml.

IoT platform can adopt Hazelcast to interactively broadcast event data to other systems of interest. By adjusting Hazelcast configuration in IoT platform server, the access to Hazelcast client side is controlled, thus, only computers in specific IP range will receive the event data.

4.2. Tenant modules

IoT Platform modules are configured on a per-tenant basis, enabling data and processing logic to be differentiated between different users. This technical approach can realize the access and control of multiple different hardware systems by a unified Internet of Things platform.

4.2.1. Tenant engine. Multi-tenant systems can provide multiple IoT applications for each instance. Each tenant is configured with its own storage space to ensure data independence. Each tenant is configured with an independent processing pipeline, which can be customized without affecting other tenants. When the IoT Platform Server starts, a default tenant user is created in the default tenant folder, and the processing logic can be configured in the corresponding configuration file. Add and create new tenants in the IoT Platform administration application that can be dynamically started and stopped without shutting down the server.

4.2.2. Tenant data stores. Tenant data stores can be implemented by configuring an SPI to provide a persistence service for tenant level information such as device and asset management. The realized service interfaces are

IDeviceManagement: Manage and allocate core equipment resources, including sites, specs, devices, events, and so on.

IAssetManagement: Manage and allocate core asset resources, including asset classes and ASSET CRUD methods.

4.2.3. Communication engine. IoT Platform communication engine handles all functions related to device interaction.

Register new or existed equipment. Devices can be manually created by APIs, or they can be registered in batches. Devices provide a unique hardware ID number and specification mark to the system, and each device has a unique ID in the system. At startup, the specification tag indicates the type of hardware the device is using and references device specifications that already exist on the system. A device sends a registration event when it boots or connects to the network. IoT Platform creates a new device record or finds an existing device record and returns a response message to the device indicating the registration status.

The event was received from the connected device. Once registered, devices can report any number or type of events to the IoT Platform and store them. Event types include location updates, sensor measurements and other collected data, or alerts in response to special events.

Transmits commands to connected devices. Each device registered in the IoT Platform has a device specification that is related to the type of hardware running on the device. The IoT Platform allows you to add any number of commands in a standardized manner, and each command can carry any number of parameters. Commands and parameters can be added through the administrative user interface or through calling REST.

4.3. Object model

IoT Platform provides a comprehensive object model for capturing relationships between concepts in tracking device data. The object model is used to realize unified device access, so that the access tasks of various IoT devices can be configured, and the efficiency of device access and management is improved.

4.3.1. Site and region. Sites are used to organize related devices and view events. An example of a site application is a location-aware device. Each site provides an entity associated with the map. When you create a site in a management application, you can specify the map type and initial location to render the site's location event on a given map[9]. IoT Platform sites group devices in the same physical location. IoT Platform management applications allow areas to be defined based on a map associated with a site. Set the boundaries of the region through the region editor.

4.3.2. Device specifications. Specifications are used to capture the characteristics of a given hardware configuration.

Device specifications command. Device specifications can be invoked by the IoT Platform from the command list, or they can be added, updated, viewed, and deleted from the admin interface or through REST services.

Device command invocation. IoT Platform provides API for invoking commands on a device based on a list in the device specification. Each command invocation is captured as an event associated with the current device association. The admin user interface and REST services allow commands to be invoked and previous calls to be searched.

Device command response. After the device processes the command invocation, it may return a response to the IoT Platform. The command invocation message carries an originating event ID and can be sent back through any response to correlate the response events and confirm device events, measurements, or alarm information. Users can list responses to a given command and initiate other actions based on the response.

4.3.3. Devices and device groups. The device refers to connected sensors. Each device can be addressed with a unique hardware ID number. New devices can be registered with this system by using a hardware ID number and device specification token. The IoT Platform can create new device records through APIs (Application Programming Interface) to allow events to be collected for devices.

A device group is a logical unit organized by multiple related devices or subgroups. These groups can be used to perform operations collectively rather than individually. Each group can have zero or more roles, allowing arbitrary grouping based on needed application. A device may belong to multiple groups, and zero or more roles may be associated with the groups.

4.3.4. Device association. Events are not directly recorded on devices. Device association is the association between devices, sites, and related assets.

Current device status. The device association also acts as a repository for recent device state. When an event is associated with processing, it keeps the numerical record of the recent event. By default, associations store the most recent location metrics, the most recent values for each metric, and the most recent alerts for each alert type. Using this storage state, IoT Platform can infer the current state of the device. The status information contains the stored date, and you can intelligently choose when to request updated data.

Associate status indicators. Each device association also preserves the state of the association itself. By default, associations are marked as active immediately after they are created. With REST services or administrative user interfaces, if a device or related asset is missing, the state can be changed to missing state, and the processing logic can be changed to missing association. The association status is updated to release at the end of the association, which indicates that the device is no longer associated and may be re-associated.

4.3.5. Device events. Device events[10] are core service data generated by the interaction between connected devices and the IoT Platform. The IoT Platform can capture many types of events, including device measurements, device location, device alert information, command invocation status, and command response return values.

5. DEVELOPMENT KIT

IoT platform is based on several open-source technologies. The main open-source modules adopted are as follows:

Eclipse. IT is chosen as the integrated development environment of the IoT platform.

Apache Tomcat 7. It is the core server that support the IoT platform. The platform is configurated as Web Archive (WAR file) which runs when Tomcat server is activated.

Spring Structure. It provides the core configuration frame which allows configuration and extension. By using same interface in customize class and inserting with Spring, the third party can extend it without referring to core code.

Spring Security. It provides the basic structure for core safety. The platform has its own clients management interface and the Spring Security Interface at the same time, which allows it to authenticate the access to system source with existing security modules.

Hazelcast. With the help of Hazelcast, the platform provides access to undergoing events based on subscription. External entities can use Hazelcast app to connect with running IoT platform example, and monitor event source including the location data, measurement, alarm and order execution.

MongoDB. It is a NoSQL database to store the platform data, with the advancement of high efficiency and good performance.

Apache HBase. In this platform, the customized HBase mode can save the device issues as optimized clusters-crossing time series data, thus realizing quick access to event according to the time of data collection.

InfulxDB/Grafana. The platform can store equipment management data in MongoDB and event data in InfluxDB by using mix methods. Thus, the data visualization can be realized using the tools like Grafana.

6. TYPICAL APPLICATION SCENARIOS

Nowadays, the application of IoT terminals has increased day by day. The development of IoT technology has widely changed our daily like and working style. The proposed IoT platform can be adopted in different industries. Almost all industries can benefit from it for it brings convenience in monitoring, automation, and analysis. Some important application scenarios are listed as follows:

(1) Port application. The data collected by equipment of the port can be uploaded to the platform and stored in time series database. The standardized data interface can guarantee the security of data.

(2) Campus Security. The terminal can upload data to applications with television, PC, MMS and videophone. The terminal can upload data to Web server for users to remote browse data with web explorer.

(3) Digital Warehouse. With the help of IoT platform, a series of automatic operation can be realized, such as tag initialization conversion, automatic warehouse in and out, equipment location tracking, hand-held terminal query and security lock.

(4) Smart Transportation. The terminal can collect vehicle information with corresponding equipment and upload it to IoT platform with wireless modules.

(5) Intelligent Agriculture. The data collecting terminals includes temperature/humidity sensor, soil temperature sensor, soil moisture sensor, greenhouse illumination sensor, video camera, etc. The collected data can be transferred to IoT platform via ZigBee wireless communication and 4G network to proceed data analysis and association.

7. CONCLUSIONS AND BENEFITS

There are a variety of sensing terminals and network access devices on the Internet of Things, data models and interface protocols are not unified, and device performance architectures vary greatly. Projects involving multi-system IoT equipment often need to put a lot of effort into equipment access and joint adjustment. By studying the IoT platform for multiple application scenarios, this paper solves the data access problem of highly concurrent IoT sensing devices, and can parse the formatted and encapsulated values in real time and then push them to the business application system, so as to meet the requirements of various typical business scenarios. The IoT Platform can access and parse new device data in a simple and configurable way. It features cross-platform, high concurrency, and high security, and has certain application value.

REFERENCES

[1] Zhao, X., “Development and application of computer Internet of things,” Electronics World, 39(1) 47-48 (2017).

[2] Zhang, L., “Computer communication and detection technology in Internet of Things system,” Information Record Materials, 22(7), 202-204 (2021).

[3] Fan, Y., “Research on computer communication network under the influence of Internet of Things,” China New Communication, 23(6), 36-37 (2021).

[4] Sun, S., “Application of computer hardware and communication network technology in Internet of Things,” Electronic Technology and Software Engineering, (4), 13-14 (2021).

[5] Li, H., “Design and implementation of virtual control panel for testing equipment based on cloud platform,” China Equipment Engineering, (7), 154-156 (2021).

[6] Liu, Z., Wang, D., Yang, J., et al., “Research on intelligent equipment management system based on cloud platform,” Modular Machine Tool & Automatic Manufacturing Technique, (2), 165-168 (2021).

[7] Han, J., “Top architecture design of IoT platform management system for radio monitoring station,” Internet of Things Technology, 11(1), 54-58 (2021).

[8] Lu, W., “Application of intelligent asset management based on Internet of Things,” Electronic Testing, (15), 87-88 (2020).

[9] Hu, X., “Exploring key technologies of Internet of Things and application of computer Internet of Things,” Computer Products and Circulation, 35(5), 63 (2018).

[10] Peng, S., “Research on construction of general Internet of things application software practice platform,” Electronics World, 42(3), 67-68 (2020).