Smart park integrated management cloud platform architecture based on microservice governance framework

D Huang, G D Jiang, B H Sun, R Q Feng and J P Wu

1Nari Technology Development Co., LTD, No. 19 of Chengxin Avenue, Jiangning District, Nanjing, 211106, China

E-mail: jiangguodong@sgepri.sgcc.com.cn

Abstract. Nowadays, the park and its enterprises face many challenges in the global environment that is service-oriented, customer experience centered and customer demand oriented. ICT is becoming a major promoter and partner for smart parks and modern businesses. The smart park takes advantage of the results of data integration and sharing, and provides decision-making through data analysis, making the park more "intelligent" for corporate and individual services. An integrated management cloud platform architecture of smart parks based on the micro-service governance framework is proposed in this paper, which adopts the springclass micro-service governance framework to solve the problems of "data island", "power island", and "user experience" among the information subsystems of smart parks. The information and services will be integrated well to make quick response to changes in the extensional needs of the park's operational services, and to achieve high scalability the system.

1. Introduction

In the past decade, the construction of IT systems in the park has encountered the common problems and situations as following. When the business department proposes some needs, the ICT department will start system bidding, with the project cycle of requirements collection, demand analysis, development, testing, and commissioning. This approach of constructing systems adapts to business-demand reponsed, has become the standard process for parks or companies to build IT systems over the years. This is where many parks or companies face the difficulties of Internet transformation today. In the industry field, the SOA concept with loosely-coupled services is often proposed to realize business reuse through the arrangement of services. It helps facilitate rapid response and innovation in business and solve the interaction problems between different heterogeneous systems [1]. Originally this could be an advisable way for the enterprise's IT system construction. However, in the past decade, many parks and companies have not gotten benefits that SOA has promised to provide [2], even they have invested a lot in the internet transformation based on SOA. The reason is that the SOA was normally applied, with ESB (enterprise service bus) together. Each system implemented the business interaction in the form of service encapsulation or service invocation. Actually, it only adopted the form of service and chose a scientific way to achieve system interconnection from the perspective of technology. In addition, the trade-off between loose coupling and agility requirements during SOA architecture remains a challenge, especially with regard to integration and personalized requirements development across systems and even across enterprise boundaries.
Now APIs (Application Programming Interface) and micro-service paradigm have become the "next big thing" to support smart parks and modern companies [3]. Many technology suppliers have joined the trend. Microservices themselves are described as a (small) application that can independently evolve and select its own architecture, technology, and platform, and can be managed and independently deployed and expanded using its own release life cycle and development methods [4]. Since 2014, micro-services have received extensive attention from practitioners such as software architects and developers, and subsequent research activities have increased significantly by 2015 [5].

To sum up, this paper puts forward a method based on the micro-service governance framework, considering the performance, advantages and disadvantages of micro-services, APIs, SOA, etc.. It can effectively solve the problems of "data island", "business island" and "user experience" in the smart park, and achieve the goal of flexible response to ensure the scalability and extensibility of the platform as the number of users and operations of the park continues to increase.

2. Microservice architecture based on springcloud

The micro-service architecture is a lightweight architecture model that advocates the division of a single application into a group of small services. Services emphasize coordination and cooperation with each other to provide users with final value [6]. Each service runs closely around a specific business in a separate process, forming a highly cohesive autonomous unit. Services and services communicate with each other using lightweight communication mechanisms (usually the Restful API based on the HTTP protocol) [7]. It can be seen that the micro-service architecture emphasizes the implementation of true componentization within a single business system to achieve loose coupling, agility, and elasticity [8]. The minimum elements of a micro-service architecture are services, service invocation, registry, service registration, and service discovery. With these basic components, you can achieve a simple service architecture. The difference between micro-service architecture and traditional architecture is shown in figure 1:

![Figure 1](image)

Figure 1. The difference between micro-service architecture and traditional architecture.

Because a large number of micro-services call each other, their relationship organization becomes a complex relationship network, which is difficult to maintain and manage. Spring Cloud is a micro-service management framework. It provides configuration management, service discovery, circuit breaker, intelligent routing, micro-agent, control bus, global lock, decision campaign, distributed session and cluster state management, etc. for the development of micro-services. Various component tools needed.

Advantages of a micro-service architecture: 1) Independent deployment of operations. The micro-service architecture ideally defines each service as a separate unit based on the principle of responsibilities, where each service is independent (that means no sharing of code or infrastructure), independently scalable, and subject to rapid deployment and change. 2) Effective isolation of resources. Each microservice ideally has an independent data source. Different microservices do not need cross-database access to ensure the independence of microservices. This effectively avoids the problem of competing databases and caching resources between services. A service memory leak will not paralyze the entire system. At the same time, since each microservice instance runs on the "container", the server resources (memory, CPU resources, etc.) are effectively isolated [9].
Weaknesses of the micro-service architecture: 1) Developers design communication mechanisms between services and resolve multiple service requests at the back end (Distributed transaction based can solve this problem). 2) To manage multiple instances of different services, increase the complexity of management, and require developers to make overall considerations (The appearance of the dock container can solve this problem). 3) There is no need to consider the micro-service model architecture for a campus where business is not yet clear and business data and processing capabilities have not yet exploded.

3. Platform business architecture based on microservice architecture

The platform is divided into three layers: IAAS layer, PAAS layer and SAAS layer based on the cloud architecture. Among them, IAAS layer provides virtual storage management of IT hardware resources, which can be flexibly deployed on a rental or self-built basis; PAAS layer deployment of unified data center and platform public services; The SAAS layer applies the platform business to form a microservice cluster module [10]. Monitor external requests such as large screens, mobile terminals, and desktop terminals to access SaaS microservices through APIs. Microservices are called through the Spring Cloud suite to access databases and data centers in the PASS data security zone. The platform business architecture is shown in figure 2:

![Figure 2. The platform business architecture based on microservice architecture.](image-url)

On the one hand, the platform collects information such as video information, alarm information, talk information, patrol information, parking information, visitor information, face access information, one-card information, energy consumption information, and equipment electronic assets information through physical equipment and networks, the data is stored in a distributed database, and then pushed to the unified data center via ETL. In the process of designing a data warehouse, data from various sources are first and for most extracted, then cleaned eventually standardized before being stored into the data warehouse. This very important step in the decision-making process is called ETL (Extract-Transform-Load) [11]. The unified data center is based on ODS-& GT; DW-GT; DM conducts data governance, storage, and operation at three levels. This ensures that each microservice ideally has an independent data source and realizes unified collection, distributed storage, and effective isolation of resources [12].

On the other hand, when considering the type of business in the smart park when microservice splitting, Based on the micro-service architecture, the construction subitems such as investment
management, energy management, information management, and intelligent transportation are divided into smart security, smart access, energy control, public information, visual UI and other micro-service cluster services. Microservice clusters and microservices under the cluster perform business calls and visits through lightweight communication mechanisms to achieve device-system-regional-panoramic operational status monitoring, field-related linkage control, and emergency response automatic response from different perspectives. Implement large screen + APP + PC multi-linkage display. Users can easily realize the integration experience beyond conventional systems through only one API address, solving the "data island" and "user experience" problems.

For example, the smart security micro-service cluster consists of video surveillance, anti-theft alarm, wireless talk, Bluetooth patrol, fire protection, elevators and other services; The smart pass micro-service cluster consists of parking lot management, visitor management, face recognition and access control management, and one card. The energy control micro-service cluster consists of new energy, distributed power supply, energy network, energy equipment, and built-in source-network-charge-storage panoramic monitoring, energy diagnostic analysis, metering, asset management, transportation maintenance, and microgrid coordination control. The public information micro-service cluster consists of user management (virtual one-card account), permission management, information distribution, and online meetings of the management-enterprise-employee three-level account system; The UI micro-service cluster consists of three-dimensional visual management, graphic management, alarm management, and report management.

In addition, the platform provides account password management, basic data management, role permissions management and other functions by establishing a virtual card account to provide account management, identity authentication, user authorization, permission control, and other functions for all microservices. Realize the management information, enterprise information, personal information and energy, access, security, access control, other business unified data interaction and identity authentication definition, and solve the problem of "island of authority". For example, enterprise information belongs to the account of the organizational entity, and the enterprise unified social credit card is used as ID identification; Management information belongs to the accounts of non-organizational entities and is divided according to roles. Personal information belongs to personal entity accounts. It is recommended to use ID cards as the main data that interacts with traffic, security, and access control. At the same time, it supports Face ID, two-dimensional code, license plate number and other overlapping authentication methods to ensure the uniqueness of the account, and data behavior. Attribution. One card account function includes: 1) to achieve user identity information integration; 2) Achieve integration of identity authentication methods; 3) Achieve user rights system integration; 4) Implementation of consumer module integration(including ordering services and support for expansion); 5) Multidimensional information cleaning and user portrait generation, that is, all the behaviors that occur through the virtual card account, all the data produced, are automatically collected and organized under the account due to the application for authentication of identity information and permissions to the system. These data automatically generate basic user portraits and support multi-dimensional filtering to generate user demand behavior tags.

4. Platform business data streams
As mentioned earlier, the smart park integrated management cloud platform stores data in distributed databases through device acquisition, and then pushes it to the unified data center through ETL. The unified data center is based on ODS-& GT; DW-GT; DM conducts data governance, storage, and operation at three levels. Microservices are then accessed through the lightweight resource call suite. Data flow flows in both directions according to the device-perceptual control terminal-campus network-large data distributed storage-unified data center-micro-application-registration center-display end. Based on the above business data flow, the business resource call and sharing integration are realized under the micro-service governance framework.

Take the visualization of UI microservice cluster data streams as an example: the visual UI aims to associate the application of data such as park informationize, intelligentization, and energy control, as
well as the large-scale non-numerical information resources generated by using graphic image technology and methods for visual presentation, helping people to understand and analyze data. Multidimensional support park operation and management, to achieve the status of the park "unobstructed".

Visual data rendering is divided into two ways: active data rendering and passive data rendering:

- Active data rendering: Custom or autonomous rendering of normal running state and normal running scene. The data stream is shown in figure 3:

![Figure 3](image)

**Figure 3.** Data invocation process for a visual UI microservice cluster under normal (active) circumstances

- Passive data presentation: refers to the visual passive presentation caused by the initiation of contact sources such as alarm signals and work order signals. The data stream is shown in figure 4:

![Figure 4](image)

**Figure 4.** Data invocation process for a visual UI microservice cluster under abnormal (passive) circumstances.

5. **Software implementation method for resource calls**

The platform's entire software operation flow based on the SpringCloud micro-service governance framework is show in figure 5:

All micro-services are deployed to each server and registered in multiple Eureka registration centers [13]. Each micro-service performs service health monitoring by establishing a heartbeat mechanism with the registration center. Ensure the normal operation of each microservice. The front end sends Http or Https requests to the Nginx server. Nginx performs an identity check and then forwards a microservice to the Eureka registration center via the Zuul route. All requests can be obtained from the registration center [14]. The addresses and ports of the available microservices are
called. In order to solve the problem of who is asked to deal with when a service has multiple instances, we propose to go through Ribbon's load balancing before requesting a microservice. Ribbon can automatically allocate the request based on the load balancing polling mechanism algorithm. In addition, Hystrix circuit breaker mechanism is integrated into the invocation process to avoid service avalanche caused by a failure of a micro-service and ensure the stable operation of other micro-services. The data generated by microservices are stored in Redis cache or relational database, and are used by distributed transaction centers and real-time distributed log systems through Kafka message queues.

![Platform software architecture based on springclub microservice architecture.](image)

The configuration files of each micro service are managed by Config Configuration Center. In order to better maintain these configuration files, Cloud Bus uses lightweight message broker to automatically refresh configuration when modifying configuration. Due to the different access pressure of each micro-service, it is necessary to monitor micro-service with Turbine, observe the operation of micro-service in real time, and dynamically regulate micro-service [15]. Zpkin collects timing data of microservices to track system latency of microservices architecture [16].

6. Experiment validation
The acceleration ratio is used to verify that the platform using micro-service architecture can effectively improve the execution efficiency of processing requests under different number of user requests. The acceleration ratio is defined as follows:

$$\frac{S}{n} = \frac{T_s}{T_d}$$  \hspace{1cm} (1)

In the formula, it represents the processing time of user requests on a single server and the processing time of a desktop server. Ideally, as the number of servers increases, it will overlap.

Two groups of experiments were designed to verify the processing efficiency of smart Park platform based on micro-service: the comparison of running time between micro-service architecture platform and common monolithic platform under the same number of users' requests, and the accelerated comparison of platforms under different number of users' requests.

Experiment 1
User requests are processed on two computing platforms: smart Park platform based on
micro-service architecture and common monolithic platform. The smart security module of the two platforms can send requests at the same time. The number of simulated users is from 1000 to 10000. The experimental results are shown in figure 6:

It can be seen that with the increase of the number of user requests, the processing time of user requests in the two platforms becomes longer. However, the request processing time of the micro-service architecture platform is much shorter than that of the common platform, which shows that the micro-service architecture platform has more advantages in processing the execution efficiency of user requests.

Experiment 2
The number of analog users is divided into 5000 and 1000 parts, and the acceleration ratio is calculated. The server cluster is taken from 1 to 5, and the experimental results are shown in figure 7:

![Figure 6](image_url)

**Figure 6.** "Run-time Comparison" of microservice architecture platforms and common single platforms under the same number of users request.

![Figure 7](image_url)

**Figure 7.** "Acceleration ratio comparison" of microservice architecture platform and general single platform under different user number requests.
The experimental results show that with the increase of the number of server clusters, the acceleration ratio increases gradually, and the acceleration ratio of 10000 user data sets is faster than that of 1000 user data sets. The acceleration ratio of two data sets increases linearly. This shows that the micro-service architecture has good real-time performance and scalability in processing user requests, especially when dealing with a large number of user requests.

7. Conclusion
Based on the micro-service governance framework, the integrated management of the smart park cloud platform can effectively solve the problems such as "data island" and "power island", ensure the scalability of the platform, and provide necessary conditions for the construction of the open platform and business ecology of the park. Experiments show that the smart Park platform based on micro-service architecture has good execution efficiency and scalability when multi-users simultaneously initiate requests.

References
[1] Gu Q R and Zhang H T 2010 Research of using SOA build enterprise application integration strategy 2010 2nd IEEE International Conference on Information Management and Engineering (Chengdu, China)
[2] Xiao Z X and Qiang X J 2016 Reflections on SOA and microservices 2016 4th International Conference on Enterprise Systems (Melbourne, Australia)
[3] Zhong H, Xie T, Zhang L, Pei J and Mei H 2009 MAPO: Mining and recommending API usage patterns Proceedings of the 23rd European Conference on ECOOP 2009 - Object-Oriented Programming (Genoa, Italy) pp 318-43
[4] Sill A 2016 The design and architecture of microservices IEEE Cloud Computing 3 76-80
[5] Pahl C and Jamshidi P 2016 Microservices: A systematic mapping study Proc. of the 6th Int. Conf. on Cloud Computing and Services Science (Rome, Italy) pp 137-46
[6] Johannes T 2015 Microservices IEEE Soft. 32 1-116
[7] Krylovskiy A, Jahn M and Patti E 2015 Designing a smart city internet of things platform with microservice architecture Proc. Third Int. Conf. Future Internet of Things and Cloud (Rome, Italy) pp 25-30
[8] Yu Y, Silveira H and Sundaram M 2016 A microservice based reference architecture model in the context of enterprise architecture IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (Xi’an, China)
[9] Singleton A 2016 The economics of microservices IEEE Cloud Computing 3 16-20
[10] Miyachi C 2018 What is "Cloud"? It is time to update the NIST definition? IEEE Cloud Computing 5 6-11
[11] Diouf P S, Boly A and Ndiaye S 2017 Performance of the etl processes in terms of volume and velocity in the cloud: State of the art 4th IEEE International Conference on Engineering Technologies and Applied Sciences (Salmabad, Bahrain) pp 1-5
[12] Linthicum D S 2017 Cloud-native applications and cloud migration: The good, the bad, and the points between IEEE Cloud Comp. 4 12-4
[13] Hoffmann T R 1992 Eureka: A hybrid system for assembly line balancing Manage. Sci 38 39-47
[14] Wen Z P, Li G L and Yang G H 2018 Research and realization of nginx-based dynamic feedback load balancing algorithm IEEE 3rd Adv. Inform. Technol. Electron. Autom. Contr. Conf. (Chongqing,China)
[15] Ouyang T, Kusiak A and He Y 2017 Modeling wind-turbine power curve: A data partitioning and mining approach Renew. Energ 102 1-8
[16] Patten B C 2015 Link tracking: Quantifying network flows from qualitative node–link digraphs Ecol. Model. 295 47-58