# Design of a Joint Microservices-based Smart Epidemic Prevention Platform

*Liang Zhang*
*College of Engineering School*
*Shanghai Polytechnic University*
*ShangHai,201209,P,R.China*
*Corresponding author's e-mail: 710761875@qq.com*

***Abstract*—Currently, as COVID-19 spreads around the world, epidemic prevention departments from different provinces and regions often need to carry out cross-regional cooperation. In the paper, some new office platforms based on some office needs of the joint prevention and control work are tried to build. In the paper, the platforms built is capable of visualizing the risk map of regional risks, covering a multi-segment integrated joint prevention and control collaborative business platform system of the CDC, customs, relevant medical institutions and health tracking management departments. For the staff in each business link, they can access the data of cases, close contacts and key health tracking objects through the platform. Based on the work, it's necessary to print relevant certificates online, and at the same time, they also supplement the business data of the link to the data chain, providing better information services for improving the efficiency of joint prevention and control. In the process of making an analysis of the business needs, it is found that when building the platform, it will face problems such as many functional modules, variable demand, and too large amount of concurrency at a certain time. In the paper, the implemented architecture techniques would be explained, the details of the key technology implementation adopted by micro-services in realizing the platform would be clarified, and the significance of the platform to the epidemic prevention work would be finally discussed.***

***Keywords*—microservices, spring cloud, smart prevention platform, distributed system, Collaborative work***

## I. INTRODUCTION

The new crown pneumonia epidemic is spreading rapidly all over the world. Although governments of various countries have successively introduced emergency response measures, there is still a trend of rapid increase in confirmed cases in some countries. my country has made remarkable achievements in epidemic prevention and control, but still faces the import of overseas epidemics and a possible second wave of epidemics in autumn and winter. In the face of the continuous development of the epidemic, the importance of coordination and cooperation has become increasingly prominent. Experts and scholars from many countries have called for pooling forces to fight the epidemic.

Relevant regions in my country, such as the Yangtze River Delta, the Pearl River Delta, Beijing-Tianjin-Hebei, and the Greater Bay Area, strengthen regional cooperation in public health emergencies and major infectious disease outbreaks, and continuously improve the joint prevention and control of major epidemics and public health emergencies. Explore the sharing of information, technology, personnel, materials and other resources, and jointly carry out scientific and technological research and practice and work together to create a first-class regional disease prevention and control system. For example, the Beijing-Tianjin-Hebei region has carried out the collection and analysis of disease information and related data in the three places, and jointly formulated the Beijing-Tianjin-Hebei major infectious disease prevention and control and emergency response plan system for public health emergencies.

With the development of the new epidemic of Covid-19, the construction of an integrated control platform for epidemic emergency preparedness and joint information based on an internet platform can greatly facilitate the prevention of epidemics and the realization of information technology in the prevention and control of epidemics. Due to data security issues, different regional epidemic prevention departments only use their independent system platforms so far, and do not exchange heterogeneous data. In this case, this has created a lot of difficulties for cross-regional cooperation in epidemic protection.

In traditional system architectures, a Monolithic Architecture is generally used for development, considering development cost issues and the number of people using it. However, in the design of the Joint Smart Epidemic Prevention Platform, there are too many departments involved, such as wide coverage of functional services, high security of heterogeneous data. At the same time, new epidemics may occur at any time in the future. Therefore, the Monolithic Architecture cannot meet our needs for the Platform Architecture, both in terms of service functional modules and technical aspects. In response to these problems, this paper proposes an architectural idea for a Joint Microservices-based Intelligent Epidemic Prevention Platform.

- This platform is used to integrate several local epidemic prevention platforms scattered in different areas, and by building a unified comprehensive epidemic prevention platform service data platform, so that several application subsystems can transmit and share information and data.

2021 International Conference on Cyber-Physical Social Intelligence (ICCSI) | 978-1-6654-2621-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCSI53130.2021.9736199

- To avoid the problem of data islands, and to maximize the utilization of information resources has become the basic goal of informatization construction.

- Ensure the interconnection and intercommunication between distributed heterogeneous systems, establish a central database, complete data extraction, concentration, loading, display, and construct a unified data processing and exchange platform, and provide standardized data services to the outside world. Establish a unified and standardized interface between them, exchange and share information through computer networks, and finally realize one-stop, integrated and collaborative office, providing a new integrated business service platform for different epidemic prevention departments. Through this platform, it is of great significance to the current epidemic work.

## II. RELATED WORK

In the development of a platform, a mature framework is not static, but a relatively simple framework gradually evolves into a mature technical framework. In the development process of software technology, it is roughly divided into the following processes. The first is the monolithic architecture based on LAMP.

In this case, as the amount of user requests increases, the data in the database will gradually increase, and io will also encounter bottlenecks. The software technology and solution are the configuration of separate read and write to the database. In a normal system, the number of times the user's reading is greater than the number of writes. you can configure a one-master and multiple-slave solution. As shown below.

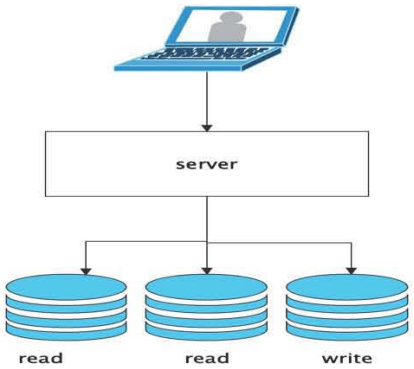


Fig1. Read and write separation

Although the database is configured with read-write separation, with the increase of network users, it still cannot solve the problem of high concurrency of big data. At the same time, the availability of bandwidth resources of the network can become a bottleneck. Software technology is waiting to be updated, and it was not until 2014 that the concept of microservices appeared. Software engineers began to use microservices to solve the problems encountered in the development process.

### A. INTRODUCTION TO THE MICROSERVICES FRAMEWORK

In 2014, Martin Fowler introduced the concept of services [1-2], where complex systems with a Single Architecture are functionally split into services, with each service module being individually independent of each other and solving only a single function of its own module. As a result of the split into smaller functional modules, each single service has its own independent process accordingly, and the individual processes can interact with each other via Communications protocols or inter-process communication. When building the corresponding functional modules, they are written according to the functional business, either using different programming languages or different storage technologies [3]. Both unit tests and functional extensions can be carried out independently.

*a) Benefits of the Microservices framework*

Compared to traditional Monolithic Architectures, Microservices Architectures are integrating the architectural idea of splitting a complex system into smaller granular services by business function blocks [4]. The service boundaries of a single Microservice module are more clearly defined. During the development process, each service module is free to choose its technology platform according to its needs, and a better coupling effect is achieved between the modules. This makes the whole system platform more independent, available and scalable. The advantages of Microservices are as follows:

- Business function componentization:

During the development of the system platform, each business function is seen as a functional component rather than as integration for system development. In this case, the individual functional modules focus only on developing their own business functions. A high degree of decoupling is achieved between the individual service modules. As a result, Microservices have the advantage of high cohesion and low coupling. At the same time, due to the relative independence of each functional module, when new business requirements arise, new functional blocks can be developed directly and independently, without the need to modify the original system, which brings great convenience for later expansion, maintenance and various management tasks.

- Free choice of technology stack:

In implementing their separate business functions, each may have different requirements and different future functional extensions, which may require different technology stacks to implement. It is due to the characteristics of Microservices [5] that there is no need for a unified technology stack during the development process. Instead, each functional module is free to choose its own technology stack according to its business characteristics. This allows for rapid technology updates and reduces the project risks associated with using a unified technology stack.

- Auto-scaling in terms of physical hardware

As the volume of business varies between each service component, the QPS per unit time can also vary significantly. In the traditional Monolithic service architecture, according to the principle of the barrel effect, the deployment volume should be determined at a later stage according to the business module with the largest volume of requests per unit of time. In this case, a great waste of resources is caused. Under a Microservice Architecture, CPU computing resources, storage resources, memory resources and network bandwidth resources can be dynamically allocated according to the data requests of the respective business function blocks [6]. This greatly improves resource utilization and minimizes the waste of resources associated with some of the service modules.

- Continuous Operations and Maintenance

In later deployments, Docker-based Container Technology can be used, so that in the event of downtime, only the corresponding Microservice modules need to be redeployed [7], with no impact on other business functional modules. At some point, there will be a surge in business volume, and operations and maintenance staff only need to move the corresponding functional modules for vertical or horizontal expansion can be. With the Microservice model, the independence of the functional modules can effectively reduce O&M maintenance.

- Less code base capacity

During the development of functional modules, developers only need to understand and focus on the respective business functions, rather than looking at the code of the entire system platform. Accordingly, only the corresponding modules need to be maintained and compiled. Less code in the codebase means shorter compilation times and faster development.

### b) Disadvantages of the Microservices framework

In the course of the development of Internet software technology, a large number of excellent technologies based on Microservices have emerged, bringing great benefits to developers. However, the popularity of Microservices has been accompanied by new difficulties for developers due to the complexity of Microservices themselves. It and its disadvantages are as follows.

- High development costs

Each module is selected independently according to its own business characteristics, which requires a larger and more comprehensive development staff than a Monolithic Architecture and poses more difficulties both in terms of technology and overhead costs.

- More code volume

Although each business function module has a smaller code base, the fact that the modules are developed independently of each other means that each module will have its own technical implementation for some of the smaller requirement implementations, which can whole into a whole system with more code blocks in the realm.

- Increased system complexity

As a result of splitting the Monolithic service into more functional modules, each of which is deployed on a different physical machine. It is in fact a large distributed system. The modules are faced with a series of problems such as communication methods, network latency, data sync, distributed lock and so on. As a result, the complexity of the system increases significantly.

- Increased API adjustment costs

The interfaces need to be standardized whether they are called via RPC remote procedure calls or via restful style HTTP network requests between the various functional modules. When a particular port needs to be adapted, it needs to be adapted for all the functional modules that call it.

B Intelligent Epidemic Prevention Platform

In order to further improve the efficiency of national epidemic protection and safeguard economic development, we are trying to establish a cross-regional collaborative smart epidemic prevention platform based on internet technology. In this platform, we hope to achieve the following innovations.

- Try to build a unified joint prevention and control epidemic intelligence information platform. A new working model of joint epidemic prevention and control by different provinces, cities and departments based on a public platform.
- Heterogeneous data sharing becomes difficult due to the problem of data silos. Due to the high mobility of epidemic personnel, the ability of different regions to share data on epidemic patients in a timely manner in this situation is the key to whether they can be diagnosed in a timely manner. Therefore, the smart epidemic prevention platform should create a unified data warehouse to facilitate data sharing.
- In the face of social issues of public safety, epidemic prevention should not only focus on the current epidemic, but also on possible future epidemics. A smart epidemic prevention platform can report all kinds of epidemic data in a timely manner and inform the public about them.
- The ability to create an epidemic map and follow up on key patients. Observe patients for a period of time after they have been discharged from the disease.

Faced with the complexity of the functions of the smart epidemic prevention platform and the future expansion of business functions, the traditional Monolithic Architecture will face some problems as follows: 1. The smart epidemic prevention platform contains many functional modules, the code volume is too large, various functions are difficult to implement, and the online time will be prolonged. 2. Each module of the platform requires different physical hardware resources, and the use of a unified Monolithic Architecture will cause 3. Putting different business requirements in the same module makes it difficult to maintain and expand later, which is contrary to our desire for the platform to be as scalable as possible.

The Microservices framework is a divide-and-conquer idea. A large system platform is split into single functional modules. Each function is implemented independently. The Microservice architecture has the advantages of good scalability, functional independence and dynamic expansion. Based on these advantages, it can effectively solve the problems we face when building a smart epidemic prevention platform.

## III. Design and Architecture of the Smart Prevention Platform

### A. Unbundling Services

Microservices is a divide and conquer philosophy. When using Microservice architecture, the first step should be to split the functional business, which should be based on certain principles in order to facilitate later expansion and maintenance. The specification for service splitting is as follows.

(1) Single function principle: The split should be refined so that a functional module should only focus on one business and complete only one function. The fewer responsibilities a single module has, the easier it is to maintain and expand vertically.

(2) The principle of service autonomy: modules should remain independent and the operating environment should not depend on other functional modules.

(3) Lightweight communication principle: between each functional module or between different processes often need to communicate across modules, often between each other to carry out a large amount of data transmission. In this case, a lightweight data transmission protocol should be chosen first to minimize the data volume traditionally, avoiding useless data transmission and wasting the resources of network bandwidth.

(4) Port specification principles: When designing a public API, certain specification principles should be observed. For example, include the function module name, version number, port name and the corresponding request method. Uniform API design principles can maintain interface usability and are more flexible.

Based on the above principles of Microservice splitting, the Smart Epidemic Prevention Platform is designed. In this paper, the platform is divided into two main capability layers.

Basic Capability Layer: Based on the principle of Microservice unbundling, the platform is unbundled in the smallest way possible, with each functional module providing a single service function. The functions provided at this level are the most basic capabilities of the platform. These include database storage services, email services, SMS notification services, Redis caching services, file storage services, unified authorization services, unified gateway services, etc.

Capability display layer: The capability display layer is divided according to business capability blocks, mainly to meet the application services for daily work. It mainly includes new data of Covid-19 management capability, cross-region information notification capability, joint risk assessment capability, epidemic map capability, account management capability, departmental management capability, platform management capability, template management capability, etc. Among them, the capability presentation layer mainly relies on the basic capability layer in order to complete the corresponding functions.

Each functional module has a single responsibility and performs some of the functions of the platform on its own. Among them, the various functional modules of the basic capability layer often need to cooperate with each other in order to complete the application services provided by the capability display layer. Take New data of Covid-19 management as an example, when it is necessary to make changes to New data of Covid-19, it is necessary to check the corresponding permissions to confirm whether there is permission to make changes to the data, then persist the data to the database, and finally send an email for notification, through a series of actions before the modification of the data is completed.

### B. Integration Architecture of the system

The entire platform design is shown in the figure below. The platform as a whole can be divided into three layers from top to bottom: client, capability display layer and basic capability layer.

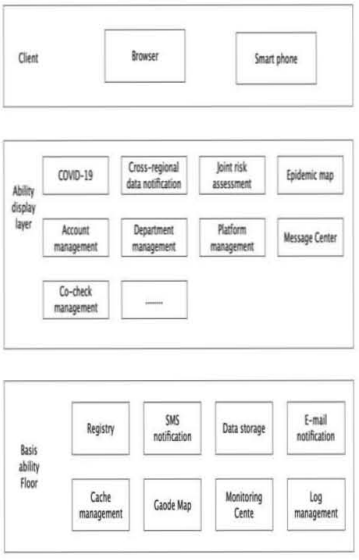


Fig. 3 Smart Platform Epidemic Prevention Platform

1) Client: Mainly as a dashboard background interface display, customers can enter the background through the browser or mobile devices, in our later envisaged, should be able to expand more client access methods, such as WeChat Mini programs, IPAD, etc.

2) Capability presentation layer: This is primarily used as the main in-application for day-to-day work, providing basic functionality but trusting its capabilities to the underlying capability layer in order to perform the corresponding functions.

3) The base capability layer: the smallest unit of Microservice splitting, when Microservices are split, the business splitting is mainly based on the base capability layer.

C. Technical architecture design

Spring Cloud is a complete distributed system solution that uses a series of middleware to solve the problems faced in the development of Microservices. Spring Cloud is now a typical representative of the distributed Microservices community, with a better ecosystem and more powerful features than other Microservices solutions. Spring Cloud is basically the first technology solution used by top tier Internet companies in China to develop web projects with a distributed architecture.

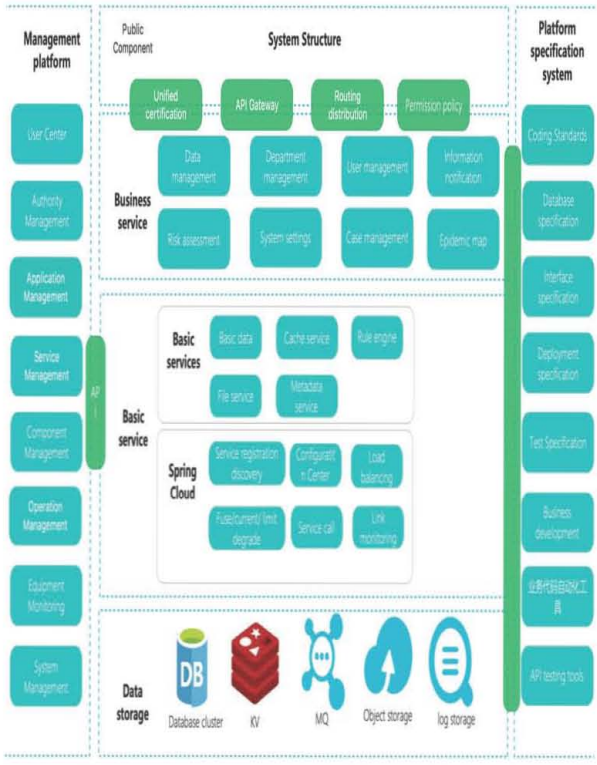


Fig. 2 Platform architecture

## IV. KEY TECHNOLOGIES

A. Configuration Center

When the Smart Prevention Platform is splitting its services, the more granular the functions are, the more functional blocks are split. Each functional module requires its own configuration file. And in the development process of a set of platforms, even the same module requires different configurations during development, testing and deployment. The whole platform will have a bunch of configuration files and the configuration will often have to be modified from time to time during the development process. It is therefore clear that for a system platform, unified configuration management of configuration files is a problem that must be solved during the development process. The idea of building a configuration center for the smart epidemic prevention platform is to centralize the various configurations, platform parameters and switches of the various modules of the platform in the same management center. And provide a unified API interface to the outside world, and each function gets the corresponding parameters through the API. When the parameters of the module need to be modified, simply enter in the configuration center for modification. At this point, the configuration center will send out a corresponding notification, telling the corresponding module to go to the configuration center to obtain the parameters again.

B. Service registration discovery

The platform will be split into service modules, each of which will be deployed with multiple instances on physical machines. The instances will communicate with each other. However, in practice service invocation becomes more difficult due to the large number of service modules that can be invoked against each other. Each service instance is peer-to-peer in the network. We can build a registered service discovery center and register each service module in the center. The main function of the registry is to transfer the previous calls between the service modules to the registry, as the registry has information about all the modules.

C. Load Balancing Mechanism

Microservices are a system platform split into smaller functional blocks, and the number of instances to be deployed is determined by the actual number of requests per unit of time. When making service calls, instances of a particular service functional block are invoked according to a certain invocation policy. When the platform is built, a load balancing layer will be built before the application layer to distribute data traffic, and the specific distribution strategy can be decided according to the production environment.

## V. CONCLUSION

This paper first discusses the new epidemic prevention concept of building a joint smart epidemic prevention platform based on Microservices, followed by a description of the advantages and disadvantages of Microservices and the problems that may exist when using a traditional monolithic architecture platform. The final decision to use Microservices as the architecture for this platform is made, and each functional block at each level of the platform is outlined. The components used are also described. The platform is designed to be independently deployed, highly scalable and easy to maintain using Microservices architecture.

The epidemic is still not over and the use of the new joint platform for epidemic prevention and control is still in the exploratory stage. At a later stage, it is hoped that the platform will be continuously improved to enable better epidemic prevention and control.

## VI. Acknowledgments

This work was supported by the Collaborative Innovation Platform of Electronic Information Master under Grant No.A10GY21F015 of Shanghai Polytechnic University.

## References

[1] Mei, Xuan. Research on Microservice invocation based on Spring Cloud [D]. Wuhan: Wuhan University of Technology,2018.

[2] RuWang.Mubammad Imran,Kashif Saleem.A microservice recommendation mechanism based on mobile architecture[J].Journal of Network and Computer Applications,2019.

[3] Yao Gang, Wang, Congbin., Wu, Haili. An analysis of Microservice architecture based on Docker technology[J]. Information Systems Engineering, 2020(4):PP.127-128.

[4] Singh G, G´omez-Luna J, Mariani G, et al. NAPEL: near-memory computing application performance prediction via ensemble learning. in: Proceedings of the 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, 2019. PP.1-6

[5] Sun, Yu, Zhou,Gang. Research on the construction of a resource discovery system platform based on Microservice architecture[J]. Journal of Library science in China, 2020, 46(1): PP. 114-124.

[6] Wang,Zhishuo. A preliminary study on the common construction and sharing of scientific and technological literature and information resources[J]. China Science and Technology Investment, 2019(16): PP.175.

[7] Zhang, Shifeng , Pan, Shanliang .Application of Docker technology in Microservices [J]. Electronic Technology and Software Engineering ,2019(4):164.