International Journal of Sciences

Research Article Volume 9 – October 2020 (10)

Study on the Construction of Public Service Platform for Risk Assessment of Consumer Products Based on Microservice Architecture

Zhou Xu[1], Fangkai Zhang[1], Meiyu Wu[1]✎, Yanli Meng[1]

[1]School of economics and management, Beihang University, Beijing, 100191, China

Abstract: Faced with the trend of more and more diversified risks of consumer products under the new situation, the traditional risk assessment platform based on single architecture can no longer cope with the growing and rapidly changing new demands. This paper proposes a public service platform for risk assessment of consumer products based on microservice architecture and studies its key issues. Combined with the idea of flexibility and extensibility, this paper analyses the operation defects of traditional architecture and proposes the construction of the public service platform based on the microservice architecture for risk assessment of consumer products. The granularity and partitioning principles of microservices are also studied. Finally, the construction of the actual service architecture is presented.

1. Introduction

Big data, artificial intelligence (AI), Internet of things (IoT) and other new-generation information technologies have become important innovation supports for all walks of life and even the country [1]. The upsurge of concepts such as integrate IT application with industrialization and intelligent manufacturing has injected new impetus and brought new challenges to the informatization development of traditional enterprises. The responsibilities of enterprise information personnel not only need to be responsible for the current operation of the information system to meet the needs of internal business process management, but also need to think and explore the application of emerging information technology in traditional enterprises and methods. How to quickly implement the internal business needs of enterprises through the new generation of information technology and follow the need of users for continuous optimization and improvement has become the focus of enterprise information workers. Therefore, the risk assessment staff of consumer products has been exploring a way to support the rapidly developing business with limited resources and solve the construction of public service for risk assessment of consumer products in the context of big data [2].

The application of traditional architecture is widely accepted. However, due to the complexity and diversification of online business requirements, it involves significant communication costs and is difficult to develop business logic in the new requirements research stage. A traditional architecture cannot satisfy the scalability and maintainability requirements of complex applications.

Microservice architecture, as a popular software architecture style, can build complex applications in a more flexible, lightweight and loose-coupled way [3]. It provides a feasible technical route for the transformation of the monolithic architecture system. This paper studies the construction of public service platform for risk assessment of consumer products based on microservice architecture. According to the idea of microservice architecture, the public service platform for risk assessment of consumer products is divided into several small autonomous and collaborative service entities [4]. Decomposing large and complex problems to small and simple ones can effectively reduce the coupling between system business logic [5]. At the same time, the flexibility and extensibility of the system are improved so that it can effectively support the work of risk assessment of consumer products in the context of big data.

2. Microservice architecture

The concept of a microservice architecture, which emerged in 2012, describes a specific way to design software applications as a suite of services that can be deployed independently [6]. In 2015, more and more BBS, online community, blog and Internet industry giants began to discuss and apply microservice, which further promoted the development and innovation of microservice [7].

2.1. Definition of microservice architecture

Microservice architecture is an approach to developing a traditional application as a set of small services, each running in its own process and

This article is published under the terms of the Creative Commons Attribution License 4.0
Author(s) retain the copyright of this article. Publication rights with Alkhaer Publications.

Published at: http://www.ijsciences.com/pub/issue/2020-10/
DOI: 10.18483/ijSci.2399; Online ISSN: 2305-3925; Print ISSN: 2410-4477

✎ **Meiyu Wu (Correspondence)**
☎ +86 18396851801

communicating through a lightweight mechanism [8]. These services are built around business functions and can be deployed independently. The centralized management of these services is minimal, and they can be written in different programming languages and use different data storage technologies.

Microservice architecture divides all application systems into multiple manageable services. Each service performs a specific business, while the overall function remains unchanged. Individual services are easier to develop, understand, and maintain. Each microservice can be independently deployed, and the release of a traditional function online no longer requires the coordination and influence of other systems [9]. Each service can achieve independent expansion. According to the actual business needs, the deployment scale of each service can be adjusted dynamically in real time, and different instances of the same service can be enabled quickly to meet the sudden traffic. Therefore, a microservice architecture can speed up deployment and respond to changing requirements rapidly.

2.2. Characteristics of microservice architecture

The microservice architecture has many advantages. Firstly, it solves the problem of systems that are too complex and difficult to develop by decomposing large applications into a set of services. While the number of functions remains the same, the application has been divided into manageable chunks or services. Each service has a well-defined boundary framed by a remote procedure call (RPC) driven or message driven API [10]. Secondly, microservice architectural patterns force a degree of modularity so that individual services can be developed more quickly and are easier to understand and maintain. This architecture allows each service to be developed by a dedicated team, with the developer free to choose any technology that conforms to the service API contract [11]. When writing a new service, they can choose the most current technology, regardless of the existing service technology. Thirdly, because services are small, it becomes more feasible to rewrite old services using current technologies. The microservice architecture pattern enables each microservice to be deployed independently, without the developer having to coordinate other services when changes are made [12]. These changes are deployed as soon as they are tested, which makes continuous deployment possible. Finally, the microservice architecture pattern enables each service to expand independently, and to deploy instances that meet the capacity and availability constraints for each service.

But microservice architecture has some intrinsic disadvantages. Firstly, microservices involves developers choosing and implementing message-based or RPC-based inter-process communication mechanisms. Modules need to call each other through language-level method or procedure calls, which is much more complicated than traditional applications. Secondly, the partitioned database architecture is the challenge for microservices. Business transactions for multiple business entities are fairly common, and the implementation of these transactions in traditional applications is trivial because there is only a single database. In a microservice-based application, it is challenging for developers to keep databases of different services consistent. Thirdly, testing microservice applications and deploying microservice-based applications is also more complex than traditional ones.

2.3. The differences with SOA

Microservice architectures and service-oriented architectures (SOA) support some identical propositions, but microservices and SOA are different in some aspects.

First of all, SOA is ‘reusing’ and microservices are ‘rewriting’. The primary purpose of SOA is to make it easier for enterprise systems to fuse together, to divide the different functional units (or services) of an application, and to link them together through well-defined interfaces between these services [13]. Interfaces are defined in a neutral manner and should be independent of the hardware platform, operating system, and programming language that implements the service. This allows services built into a wide variety of systems to interact in a unified and common way. Microservices typically start with the least coupled modules or the most scalable ones, and then spin them off one by one and rewrite them with agility. You can try out the latest technologies and frameworks, and then deploy them separately.

Secondly, SOA is horizontal services and microservices are vertical services. The design of SOA is layer services, which is very inflexible. Every change in the data layer can affect services in all the business layers. However, each microservice usually has its own independent data store. When the database is splitting, it can be appropriately de-normalized. In microservices, there is no need to rely on the data of other services.

Finally, SOA is top-down and microservices is bottom-up. The SOA architecture begins with the definition of service contracts. It conducts the central management on all the services, including centrally managed business logic, data, processes, and so on. SOA architectures typically pre-define each module's service interface. Communication between module systems must adhere to these interfaces, and services are their callers. Microservices are much more agile. Microservices enable users to develop the service as soon as it is available. Then, users can identify business requirements and develop iterations quickly.

In general, SOA architectures emphasize communication and decoupling heterogeneous systems, while microservice architectures emphasize fine-grained separation and deployment of systems along business boundaries.

2.4. Principles for partitioning services

There is no universal approach to service partitioning, but rather a combination of factors based on actual business. Services are split according to business functions until each service meets the requirement of ‘single and complete’ functionality [14]. To be specific, firstly, the internal process of the business system is decomposed into specific business functional components through the analysis of the system interaction process to identify the most fine-grained business functional units. According to the principle of high cohesion and low coupling, service aggregation is carried out from the bottom up to ensure the least interaction between services. Finally, according to the functional characteristics of each service module, the optimal service division that meets the conditions of high cohesion and loose coupling can be obtained by repeated iterative adjustment. For the common basic data and common modules of the system, the generic subdivision strategy is adopted. In the application of microservice architecture, service partition should be technical, business-oriented and divide-and-conquer. Take the following factors into consideration:

Business factors. First of all, the division scheme should be determined from the business perspective, and the service boundary should fully consider the business independence and professionalism.

Be self-sufficient. Clarify the division of labor of services to ensure that each service only contains its own processing logic, including data storage, business logic, message sending and receiving, etc. So that the service operation can be relatively independent, which reduces dependence on the outside.

System expansion. One of the most important roles of service decoupling is to improve system scalability. Different services have different scalability requirements, and separate the services with different scalability requirements can improve the expansion efficiency of the system.

The data is consistent. If the microservice architecture needs to change the state of business data across services, it needs to deal with complex distributed transactions and data consistency, which has a great impact on system performance. Therefore, keeping as many transactions with data consistency requirements as possible in one service helps to optimize system performance.

Information security. Different services have different information security requirements, according to the characteristics of the service can be targeted to meet the needs of information security.

3. Platform construction

We have established a public service platform for risk assessment of consumer products based on microservice architecture, which is oriented to the government, enterprises, consumers and third parties, and has realized key functions such as risk information collection, risk source identification, risk analysis and assessment of consumer products. The risk sources of consumer products are identified from the collection of risk information on consumer products, and then the risk level is analysed and evaluated. Through feedbacking the risk problem to the designer, the production and the user of consumer products, the risk of consumer products from the source is controlled, and the problem of consumer products risk management is solved. These results provide the basis and direction for enterprises to improve the quality and reliability of existing and newly developed consumer products and effectively reduce the risk of consumer products.

3.1. Division of platform services

Based on the actual business needs of the platform and in combination with the microservice division principle, the division of the services of the public service platform for risk assessment of consumer products is shown in Figure 1. It is divided into three levels: upper business service, system support service and basic data service.

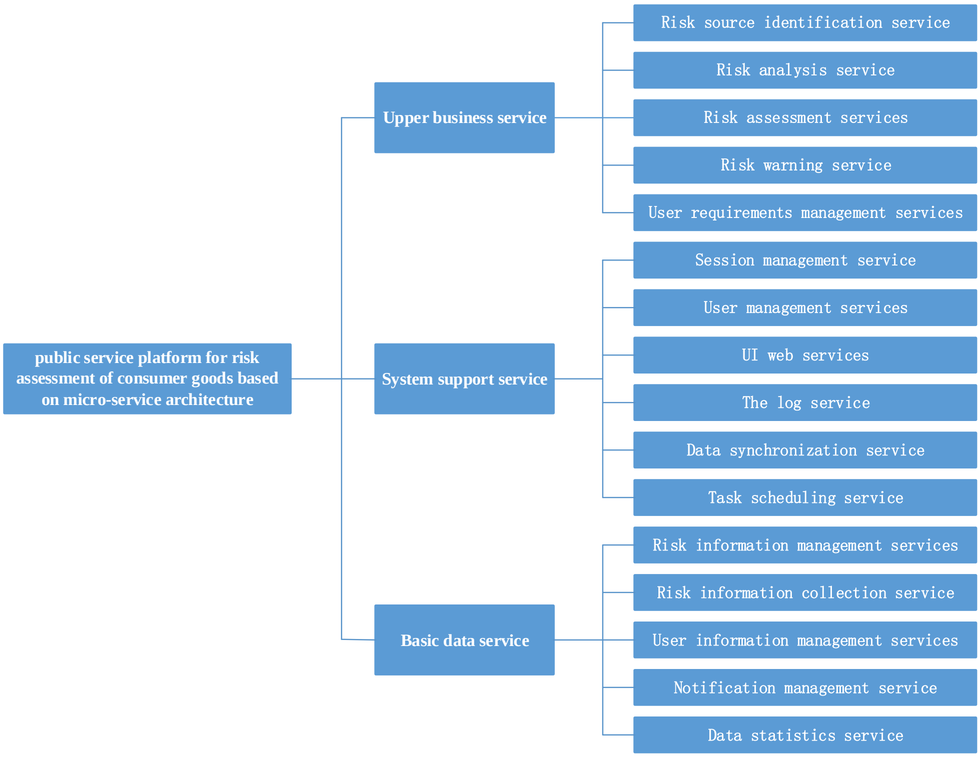
public service platform for risk assessment of consumer goods based on micro-service architecture
Upper business service
System support service
Basic data service
Risk source identification service
Risk analysis service
Risk assessment services
Risk warning service
User requirements management services
Session management service
User management services
UI web services
The log service
Data synchronization service
Task scheduling service
Risk information management services
Risk information collection service
User information management services
Notification management service
Data statistics service

Figure 1. Division of platform services.

Upper business service. It is an upper level service oriented to specific business behaviors, including risk source identification service, risk analysis service, risk assessment service, risk warning service and user requirements management services.

System support service. It is a system-level public service, that is, a common service supporting the operation of business system, such as session, user management, front-end Web, log, data synchronization, task scheduling and other essential basic functions of management system.

Basic data service. Refers to the underlying common resource class services, which are biased towards the business domain model. It includes risk information collection and management services, user information management services, notification information management services and data statistics services.

3.2. *Platform architecture*

Microservice architecture implementation requires a number of complex technical support. In order to facilitate rapid business development, we use the industry's mainstream microservice PaaS platform to reduce the cost and complexity brought by the construction of infrastructure automation tools in the early stage. By investigating and comparing the functional integrity and reliability of various major platforms, the public service platform for risk assessment of consumer products finally chooses Huawei’s ServiceStage microservice platform. The system technical architecture is shown in Figure 2.

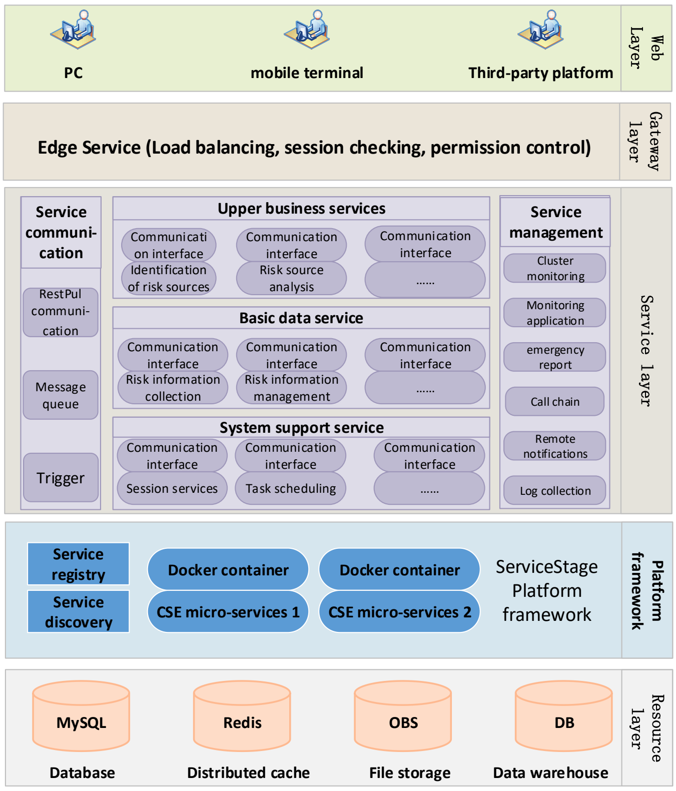

Figure 2. The platform's microservice architecture.

Resource layer. Responsible for data persistence storage, including MySQL database, Redis distributed cache, file storage and data warehouse.

Platform frame layer. The ServiceStage microservice platform is used. With the help of its CSE microservice engine, the service is deployed through the Docker container to improve the system resource utilization. Service registration, discovery and other functions are provided by the platform. Because each service runs as a separate process, communication between services needs to be done through network calls. However, network calls cannot be guaranteed to be completely reliable, so the platform layer provides fault-tolerant mechanisms such as retry, degradation and circuit breaker to ensure the reliability of inter-service calls.

The service layer. With the ServiceStage microservice platform, developers only need to focus on the service layer. According to the service division scheme of the system, each level contains its own specific service module to achieve business capability output. In addition, the microservice platform provides the service communication and service management ability to guarantee the mutual communication and monitoring operation and maintenance of services.

The gateway layer. It is the entry point for service access. Access to services by the Web layer requires an API gateway for request delivery. In microservice architecture, API gateway is also called edge service, which is usually implemented by Nginx, Zuul and other reverse proxy. The platform specific NodeJs + ServiceMesh implementation is adopted here. NodeJs is responsible for receiving external requests, and ServiceMesh, also known as service grid, can be understood as a lightweight network proxy. ServiceMesh routes external requests for different services to the corresponding service by maintaining a long connection with the service registry to obtain information about each service instance.

3.3. *Results*

By establishing a public service platform for risk assessment of consumer products, data collection, data analysis and risk assessment are realized. The platform will serve as a centralized management platform for risk information resources of all consumer products, realizing refined and standardized management. The platform is more flexible. Each microservice component is simple, flexible, and can be deployed independently, without the need for a large application server to support it. Each service is loosely coupled, microservices are highly cohesive, and each microservice is easily scalable on demand.

4. Conclusion

In order to solve the problem of risk data analysis of massive consumer products encountered in practice, the application architecture research based on microservice architecture is actively carried out. In the application architecture, the ServiceStage is used to realize the microservice of the application. Service registration, service discovery, gateway routing, service high availability and service load balancing are realized. Through the design of related architectures and the practice of emerging technologies, the need for real-time analysis and assessment of risks of consumer products is satisfied. It also provides a good foundation for the future development of the system. It is well proved that the application of microservice in enterprise informatization construction is feasible and has a good prospect.

Acknowledgment

This research was supported by the National Key Research and Development Plan (Grant No.2017YFF0209604 and No.2016YFF0202604), the National Key R&D Program of China (No. 2018YFC0807500), Liaoning Social Science Planning Fund (No. L16ATQ001).

References

1. Chin J, Callaghan V, Lam I. Understanding and personalising smart city services using machine learning, The Internet-of-Things and Big Data[C]. IEEE International Symposium on Industrial Electronics. 2017. 10.1109/ISIE.2017.8001570.
2. Xia L, Wan-Jin T, Yue-Xiang Y, et al. Risk Assessment: China's New Trend in the Consumer Goods Safety Management[J]. Standard Science, 2009.
3. Bob. Microservices, iot and azure[J]. 2015. 10.1007/978-1-4842-1275-2.
4. Krylovskiy A, Jahn M, Patti E. Designing a Smart City Internet of Things Platform with Microservice Architecture[C]. 3rd International Conference on Future Internet of Things and Cloud (FiCloud 2015), IEEE. IEEE, 2015. 10.1109/FiCloud.2015.55.
5. Garcia L M, Aciar S, Mendoza R, et al. Smart Tourism Platform Based on Microservice Architecture and Recommender Services[C]. International Conference on Mobile Web & Intelligent Information Systems. 2018. 10.1007/978-3-319-97163-6_14.
6. Rajasekar A, Wan M, Moore R, et al. Micro-Services: A Service-Oriented Paradigm for Scalable, Distributed Data Management[J]. 2012. 10.4018/978-1-61520-971-2.ch003.
7. Viennot N, Bell J, Geambasu R, et al. Synapse: a microservices architecture for heterogeneous-database web applications[C]. 2015. 10.1145/2741948.2741975.
8. Asik T, Selcuk Y E. Policy enforcement upon software based on microservice architecture[C]. IEEE International Conference on Software Engineering Research. 2017. 10.1109/SERA.2017.7965739.
9. Dong G, Wei W, Zeng G, et al. Microservices Architecture Based Cloudware Deployment Platform for Service Computing[C]. Service-oriented System Engineering. 2016. 10.1109/SOSE.2016.22.
10. Alshuqayran N, Ali N, Evans R. A Systematic Mapping Study in Microservice Architecture[C]. IEEE International Conference on Service-oriented Computing & Applications. IEEE, 2016. 10.1109/SOCA.2016.15.
11. Hasselbring W, Steinacker G. Microservice Architectures for Scalability, Agility and Reliability in E-Commerce[C]. IEEE International Conference on Software Architecture Workshops. 2017. 10.1109/ICSAW.2017.11.
12. Amaral M, Polo J, Carrera D, et al. Performance Evaluation of Microservices Architectures Using Containers[J]. 2015. 10.1109/NCA.2015.49.
13. Erl T. Soa: principles of service design[M]. 2007. http://dx.doi.org/.
14. Pautasso C, Zimmermann O, Amundsen M, et al. Microservices in Practice, Part 1: Reality Check and Service Design[J]. IEEE Software, 2017, 34(1):91-98. 10.1109/MS.2017.24.