International Core Journal of Engineering Volume 9 Issue 7, 2023
ISSN: 2414-1895 DOI: 10.6919/ICJE.202307_9(7).0035

Microservices-based Medical Platform Architecture Design

Xianjun Wang[a], Yuanyuan Zhang[b], Simiao Jia[c], and Yusong Liu[d]
University of Science and Technology LiaoNing, Liaoning 114004, China
[a]476690826@qq.com, [b]yuanyuan810713@126.com,
[c]1718949950@qq.com, [d]944860844@qq.com

Abstract

The purpose of this thesis is to explore the design of a microservices-based medical platform architecture. With the development of web platforms, there is an increasing demand for access to medical information and doctor-patient interaction. Microservices architecture provides a flexible, scalable and maintainable solution for medical platforms. In this paper, we study and design a microservices-based medical platform architecture to implement centrally managed medical access and medicine sales functions. In the architecture design, a three-tier architecture with front and back-end separation is adopted, and the concept of microservices is used to organize and manage different functional modules. Specifically, the architecture includes core modules such as doctor-patient interaction and drug purchase mall. In terms of technical implementation, technologies such as WebApi, order microservices, Nginx and Ocelot are used to enhance the performance and reliability of the system. The back-end services are relieved by the routing, load balancing and caching functions of the API gateway layer. In addition, using Rabbitmq for message processing and forwarding requests to the microservice module through the gateway enables horizontal scaling of the system. Finally, by making the order processing function independent as a microservice, it simplifies the development and maintenance of the system, and improves the overall system performance. Through the research and design of this thesis, we aim to provide reference and guidance for the microservice architecture of medical platforms, so as to meet the demand for convenient medical services.

Keywords

Microservices Architecture; Healthcare Platforms; Convenient Medical Services.

1. Introduction

The purpose of this thesis is to explore the design of a microservices-based medical platform architecture. With the development of web platforms, the demand for access to medical information and doctor-patient interaction is increasing. Microservices architecture provides a flexible, scalable and maintainable solution for medical platforms. In this paper, we study and design a microservices-based medical platform architecture to implement centrally managed medical access and medicine sales functions. In the architecture design, a three-tier architecture with front and back-end separation is adopted, and the concept of microservices is used to organize and manage different functional modules. Specifically, the architecture includes core modules such as doctor-patient interaction and medicine shopping mall. In terms of technical implementation, technologies such as WebApi, order microservices, Nginx and Ocelot are used to enhance the performance and reliability of the system. The back-end services are relieved by the routing, load balancing and caching functions of the API gateway layer. In addition, using Rabbitmq for message processing and forwarding requests to the

ISSN: 2414-1895 DOI: 10.6919/ICJE.202307_9(7).0035

microservice module through the gateway enables horizontal scaling of the system. Finally, by making the order processing function independent as a microservice, it simplifies the development and maintenance of the system, and improves the overall system performance. Through the research and design of this thesis, we aim to provide reference and guidance for the microservice architecture of medical platforms, so as to meet the demand for convenient medical services.

2. Microservices Architecture

Microservices architecture is a highly regarded architectural approach that has emerged with the growth of the Internet and cloud computing. Compared to traditional monolithic and SOA architectures, microservices architecture can better cope with the growing demand for data and services, and provide higher flexibility and optionality.

The core idea is to split applications into mutually independent services that can be distributed and deployed in separate processes and interact through lightweight communication mechanisms. Each microservice has the ability to scale and scale independently and has well-defined boundaries that are not limited by development languages, technology routes, or development teams.

Microservices architecture is characterized by automatic job deployment, fast run rates, peer-to-peer reorganization and decentralized data control and language control. Each microservice runs in a separate process and communicates through lightweight mechanisms such as HTTP APIs. These services are designed based on business scenarios and released independently through automated deployment tools. Microservice architectures can use different programming languages and data storage technologies. The use of microservices architecture in healthcare platform design can provide flexible, scalable and maintainable solutions. By splitting the healthcare platform into different functional modules, it can better meet users' needs for medical information access and doctor-patient interaction. Each microservice focuses on a specific function and interacts and forwards requests through an API gateway.

Compared with the traditional monolithic architecture, the advantage of microservice architecture is that it splits the huge monolithic process into small processes that are independent of each other, solving various problems faced by monolithic architecture in the Internet era By splitting into independent services, each microservice can be independently upgraded and maintained by the system without restarting the whole application process, thus improving the availability and reliability of the system. Microservice architecture provides services by splitting the business system into different microservices, each of which is independently compiled and deployed as a separate project and can be deployed with multiple instances on multiple servers. REST interface is used for communication between microservices.

Automation is a key technical challenge when implementing a microservice architecture. As the number of microservices may be high, it becomes tedious and error-prone to compile, deploy, debug and upgrade manually. For this reason, the development of container technology has become one of the key enablers of microservices architecture. With Docker technology, each microservice can be packaged into an independent container, enabling automated deployment and management, greatly simplifying the implementation and operation and maintenance of microservice architecture.

3. System Analysis

3.1 Technical Feasibility

Microservice architectures for healthcare platforms offer advantages such as decoupling and high availability. However, implementing a microservice architecture requires consideration of service unbundling and governance, and requires the use of proven microservice frameworks and tools. In a microservice architecture, communication and collaboration between services are required, so some distributed system technologies such as RPC and message queues need to be used. For rapid deployment and operation and maintenance, the use of container technologies is an ideal choice, the most commonly used of which is Docker, which enables rapid image building and deployment.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202307_9(7).0035

In a microservice architecture, there are important functions that need to be implemented, such as service registration and discovery, load balancing, fusion and fault tolerance. To implement these functions, service governance frameworks, such as Consul, are required. These frameworks can help manage and monitor the operational state of microservices and provide the necessary support to ensure effective communication and coordination between services.

3.2 System Functional Analysis

The medical platform is divided into four modules: user login module, doctor-patient interaction module, medical mall module, and order module.

The first module is the user login module, with user roles including doctor and patient. Users who are not logged in will default to visitor privileges and can only view information but not interact with it.

The second module is a discussion module, similar to a discussion forum where patients can ask questions and doctors can give answers and advice.

The third module is the Medical Mall, which allows users to purchase medications based on a doctor's prescription and add them to their shopping cart. This module is closely related to the disease treatment system and provides patients with a convenient channel to purchase medications.

The fourth module is the Order module, which exists as a separate service. This module has a separate database and message queue function. The order module is responsible for processing the contents of the user's shopping cart, including generating orders, processing payments and delivery and other related operations. By making the order processing function independent as a service, the development and maintenance of the system can be simplified and the overall system performance will be improved.

3.3 Business Process Analysis

After entering the medical platform, first login for identity verification, after successful login jump to the home page, the home page can enter the discussion forum or enter the product page, after selecting the right product can be added into the shopping cart.

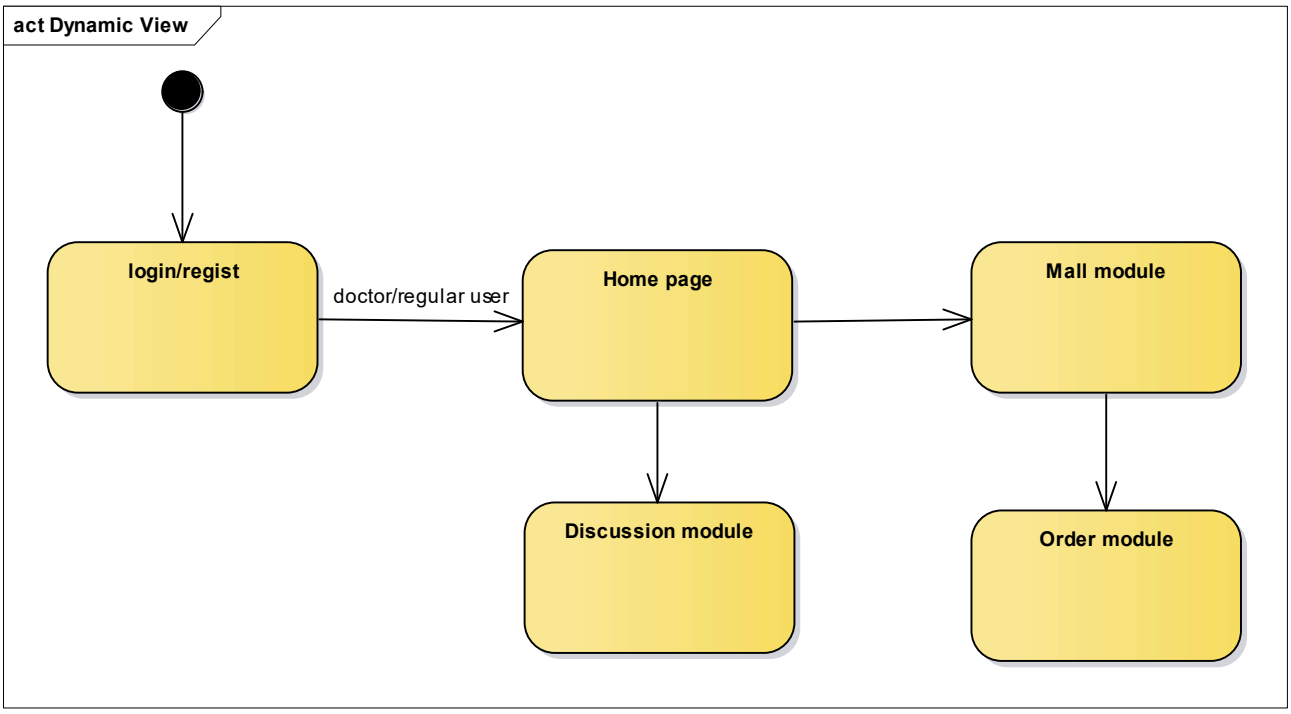

Figure 1. Business flow chart

ISSN: 2414-1895 DOI: 10.6919/ICJE.202307_9(7).0035

4. System Architecture Design

After a series of system analysis, the overall architecture of the system was designed as follows:

Access requests are received at the application portal (web platform) or at the external interface (OpenAPI) and sent to the microservice layer. The request is sent to the OcelotAPI gateway in the access layer of the microservices layer, which sends the request to the RPC service layer and interacts with Consul in the service governance layer, which mainly provides service discovery and service registration functions. In addition, the RPC service layer interacts with the underlying service layer for data, mainly with MySQL and Redis databases.

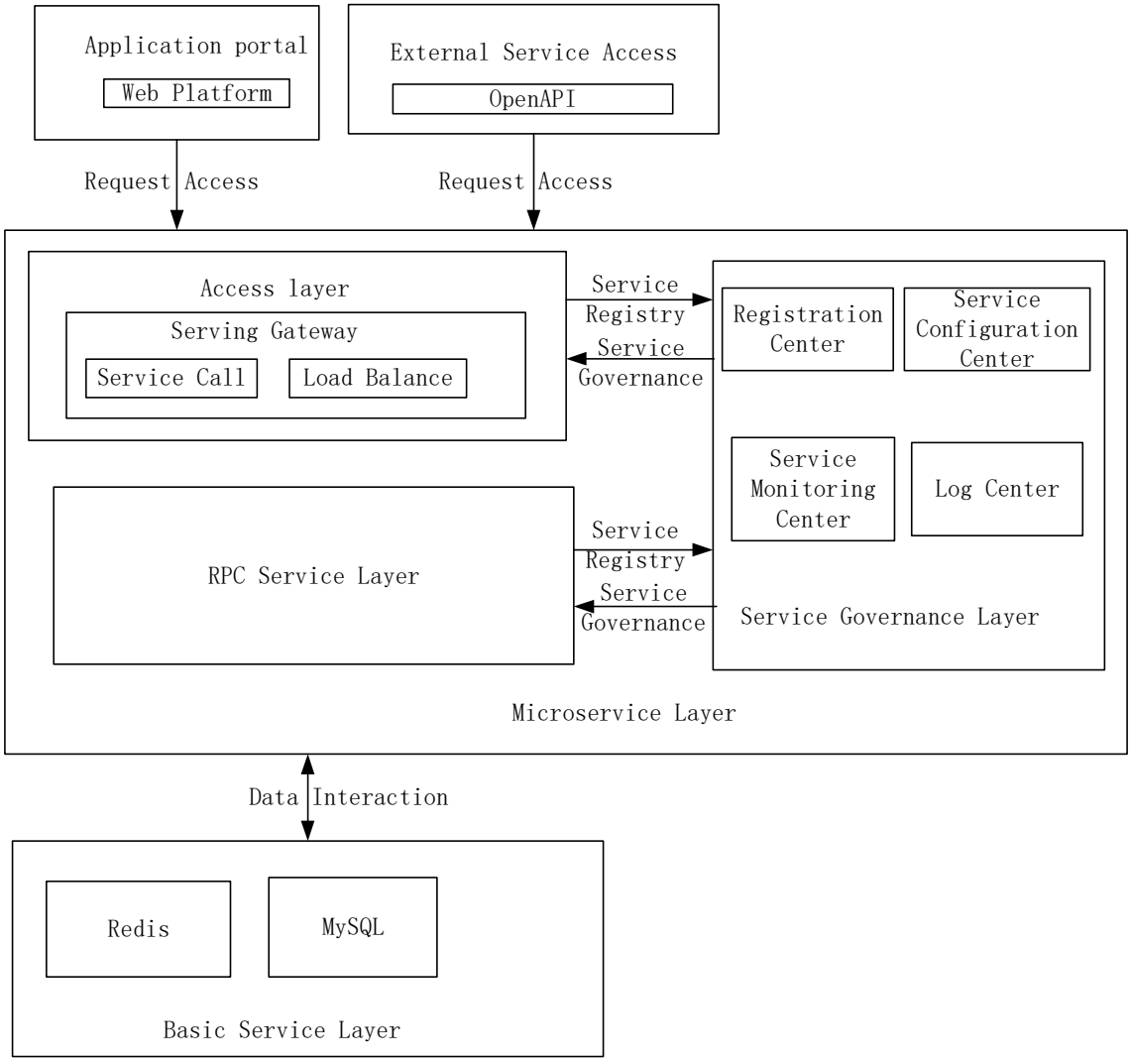

Figure 2. Overall architecture design diagram

In a microservices architecture, each service has its own request processing process. The following is a more detailed overview of the microservice request processing process:

Routing requests: When a client makes a request, the request is sent to a gateway service. The gateway service is responsible for routing the request to the appropriate microservice.

Authentication and authorization: Before routing a request, the gateway service will authenticate and authorize the client. If the client does not have enough privileges to access the microservice, the request will be denied.

Load balancing: Once a request is routed to a microservice, the microservice uses a load balancing algorithm to determine which service instance to send the request to. The goal of the load balancing

ISSN: 2414-1895 DOI: 10.6919/ICJE.202307_9(7).0035

algorithm is to ensure that each service instance is able to handle approximately the same number of requests, thereby increasing the throughput and availability of the entire system.

Request processing: Once a request is routed to the correct service instance, the service instance will start processing the request. Request processing may involve accessing databases, invoking other services, calculations and other operations.

Response processing: Once the service finishes processing the request, it generates a response and sends it back to the client. The response can include data, status codes and other metadata.

Monitoring and logging: To ensure the robustness and reliability of the service, microservices typically log various data during request processing, such as processing time, location and type of error occurrence, and so on. This data is usually sent to a centralized logging system and used to analyze and diagnose problems.

Cache processing: In some cases, microservices may use caching to improve performance. Caching typically includes in-memory caching, distributed caching, and client-side caching.

Specifically, the following chart:

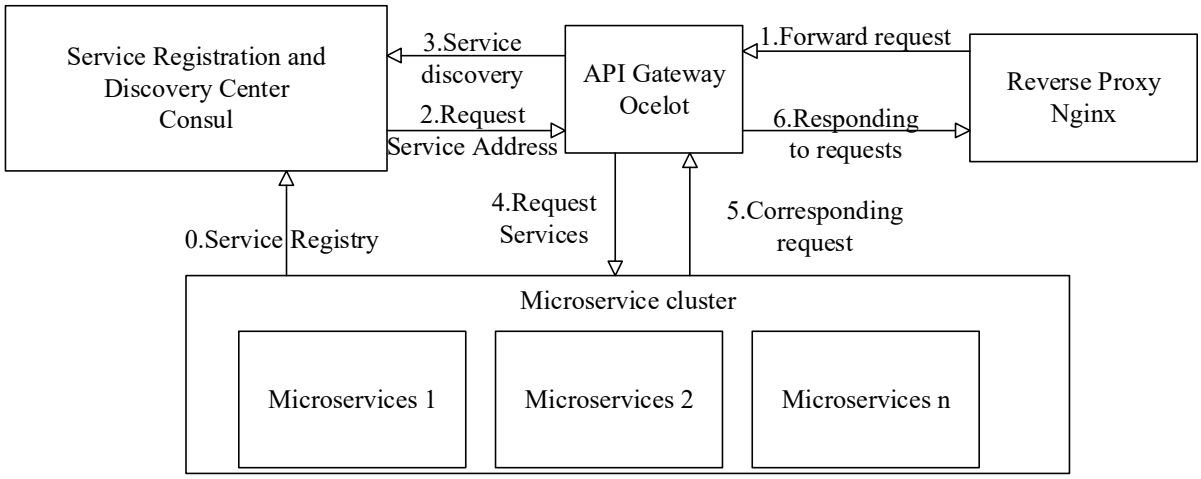

Figure 3. Schematic diagram of the process of processing requests by the microservice governance system

5. Conclusion

By studying and designing a medical platform based on microservice architecture, this paper aims to provide a flexible, scalable and maintainable solution to meet the demand for convenient medical services. By splitting the medical platform into different functional modules and using the concept of microservices to organize and manage these modules, a centrally managed medical access and medicine sales function is realized.

In terms of system architecture design, this paper adopts a three-tier architecture with front- and back-end separation and uses features of microservice architecture, such as API gateway, service governance and containerization technology, to improve the performance and reliability of the system. The pressure on the back-end services is reduced by the functions of routing, load balancing and caching in the API gateway layer. In addition, message queues are used to handle requests and enable horizontal scaling of the system. Separating the order processing function into a microservice simplifies the development and maintenance of the system and improves the overall system performance.

Microservices architecture, a highly regarded architectural approach, has demonstrated many advantages in healthcare platform design. It splits complex applications into mutually independent services, allowing each service to scale and scale independently with clear boundaries. Microservices

ISSN: 2414-1895 DOI: 10.6919/ICJE.202307_9(7).0035

architecture effectively responds to the growing demand for data and services and provides greater flexibility and scalability through features such as decoupling and high availability.

In terms of technical implementation, this paper adopts various technologies, such as WebApi, order microservices, Nginx, Ocelot, etc., to support the functional implementation and performance optimization of the system.

Through the research and design of this thesis, it provides a reference and guidance for the microservice architecture of medical platforms, and provides a feasible solution to meet people's demand for convenient medical services. The application of microservice architecture is expected to further promote the development of medical platforms, provide better medical information access and doctor-patient interaction experience, and make positive contributions to people's health and well-being.

Acknowledgments

I would like to thank everyone on the team and my instructor, Yuanyuan Zhang, for their help.

Fund Project: 2023 Liaoning University of Science and Technology Student Innovation and Entrepreneurship Training Program.

References

[1] Ye Weiyu, Lu Hanyu. Design and implementation of a big data platform for smart campus based on microservice architecture[J]. Science and Technology Innovation and Application,2023,13(15):101-104.DOI:10.19981/j.CN23-1581/G3.2023.15.023.C. Li, W.Q. Yin, X.B. Feng, et al. Brushless DC motor stepless speed regulation system based on fuzzy adaptive PI controller, Journal of Mechanical & Electrical Engineering, vol. 29 (2012), 49-52.

[2] JIANG Hongxiang, CHEN Chengdong, WU Xiaoyuan et al. Design and implementation of a public information service platform for circular packaging boxes based on microservice architecture[J]. Logistics Technology,2023,42(03):135-142.J. Liu, E.L. Chen and Z.T. He: Journal of Shi Jia Zhuang Railway Institute (Natural Science), Vol. 22 (2009) No. 4, p.40-42.

[3] Wang Yuan. Design of microservices-based IoT platform architecture [J]. Industrial Control Computer, 2023,36(04):129-130.