Scholar Publishing Group
Distributed Processing System
https://doi.org/10.38007/DPS.2021.020208
ISSN 2790-0916 Vol. 2, Issue 2: 59-66

Design Method of Distributed System Architecture Based On Three Layer B/S Mode

Kothapanlli Lakshmin*

University of Anbar, Iraq

**corresponding author*

Keywords: B / S mode, Distributed System, Architecture Design, Hadoop Cluster

Abstract: The traditional Internet industry mainly uses centralized system to process a large amount of data, but this system has complex structure, poor expansibility, low fault tolerance rate, and is not easy to maintain, so it is difficult to meet the needs of Internet technology. Therefore, in recent years, distributed systems have received more and more attention from the industrial and academic circles. This paper mainly studies the design method of distributed system architecture based on three-tier B / S mode. Firstly, this paper establishes the distributed system development environment, and uses B / S mode and spring + springmvc + mybatis to develop the system architecture. Then, the master-slave architecture mode is used to build the distributed system architecture, and the distributed system environment and system testing are carried out.

1. Introduction

In the common scenarios of the Internet today, massive data computing is not uncommon. There are a lot of massive computing scenarios in the industry, such as Taobao's "double 11 promotion" and Baidu's "Spring Festival red envelope". If these application scenarios want to operate normally, they must rely on powerful servers [1]. For these massive computing scenarios, the traditional solution is usually to use centralized systems, such as IBM mainframes. This system has the disadvantages of complex structure, poor scalability, low fault tolerance, difficult performance improvement and difficult maintenance. Therefore, distributed systems are increasingly valued by the industrial and academic circles [2-3]. A distributed system is a system composed of multiple computer nodes. Its software or hardware components are distributed on different computers, and message transmission and work coordination are carried out through network communication. The distributed system has the characteristics of easy expansion and high fault tolerance. To improve the

performance of the distributed system, it only needs to expand horizontally and increase the number of machines [4]. Therefore, the distributed system is very suitable for the Internet industry. At the same time, machine learning, data mining and other technologies are developing rapidly and their application fields are becoming more and more extensive. These technologies are extremely popular in scientific research and have been applied in many traditional industries such as catering, tourism and manufacturing. Ordinary people can also use these technologies to make efficient applications. With these new computer technologies, applications often generate a large amount of data. To process these data, it is usually necessary to build a distributed cluster. However, the price of building a distributed cluster is very expensive and difficult for ordinary people to bear. Moreover, the distributed system cluster has high requirements on the reliability of nodes, and the nodes must be fully trusted. There can be no malicious nodes in the cluster. Generally, the distributed system can only use the local area network for communication [5].

The current research on distributed systems has been very mature. The well-known distributed computing technologies abroad include Hadoop, a distributed computing framework based on batch computing, spark, a distributed computing framework based on memory computing, strom, and Kafka [6]. The well-known distributed computing technologies in China include distributed service framework Dubbo and distributed database oceanbase [7]. The related research of distributed system is also blooming. Many people have optimized and re developed kubernetes scheduler. For example, a scholar has implemented a k8s container scheduling strategy that comprehensively considers task time, user needs and resource consumption [8]. There are also scheduling strategies for some specific scenarios. For example, an expert has implemented an energy consumption optimization scheduler for IOT devices in the edge computing network environment [9].

This paper mainly introduces and analyzes the B / s technology and the mainstream distributed system architecture, summarizes the relevant research results at home and abroad, and then designs and implements a distributed system architecture based on three-tier B / S mode.

2. Distributed System Architecture Design based on B / S Development Mode

2.1. Distributed System Development Environment

(1) Development mode and development environment

In the client / server mode, the client actively initiates the request, while the server passively accepts the request from the client. In this mode, the data processing is decentralized, while the system control is centralized, and its development focuses on the client part. However, just because the client plays an important role in the development of the whole system, its disadvantages are also obvious. That is, as long as a certain application requirement of the system changes, the entire client and server must change accordingly, which is extremely unfavorable to the upgrading and rapid deployment, installation and configuration of the system [10-11]. In the browser / server mode, the client uses the system through various web browsers, and only needs to install the browser program on the client. All other processing work is completed on the server side. Therefore, the development focus in this mode is concentrated on the server, so it is very easy to expand and upgrade the system, and only needs to upgrade and develop the server side [12-13].

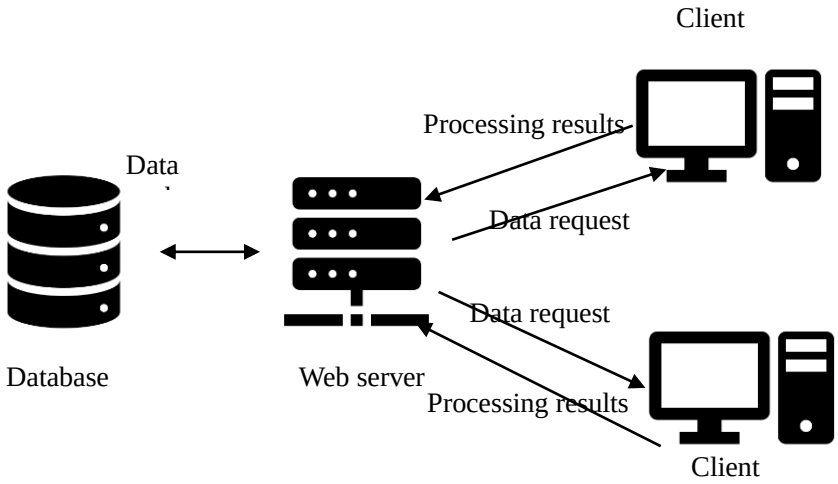

Figure 1. Web service encapsulation and management system development mode

Based on the requirement that all users can use the distributed test system Web service encapsulation and management system anytime and anywhere, this system is designed and implemented based on the B / S architecture. It can be seen from the above that the B / s development mode is a design mode that is easy to develop. Any web browser connected to the Internet can be used to interact with the web server, and the web server returns response data to the web browser [14]. The design and development mode of this system is shown in Figure 1.

Since the development mode of this system is B / S architecture, and it is also implemented on the basis of SOA and web service technology, in order to simplify the development difficulty and use the existing code framework, the system development language is Java. Java language is developed on the basis of the structure of C language. It not only has the advantages of C + + language, but also highlights the characteristics of object-oriented [15]. At the same time, we choose eclipse developed by IBM as the system integration development environment. Eclipse is an easy to learn ide.

(2) System development framework

Based on the division of system program code and combined with the characteristics of various open source frameworks, there are two mainstream combination methods at present, one is the SSH framework, namely struts 2 + Spring + hibernate, and the other is the SSM framework, namely Spring + springmvc + mybatis. In general, the combination of these two different open source frameworks can ultimately show the same functional effect. However, due to the inconsistency of the framework, there are still some differences in details, mainly between struts 2 and spring MVC, hibernate and mybatis [16]. From the perspective of composition maturity, SSH framework has slightly higher composition maturity than SSM framework; In terms of the complexity of the framework, the SSH framework is relatively high for general web projects, while the SSM framework is lighter, and its seamless connection with spring facilitates the management of programs; From the perspective of learning cost, the learning cost of SSM framework is obviously lower than that of SSH framework, so its development efficiency is relatively high. Based on the

above three considerations, the SSM framework with light weight, easy learning and high development efficiency is selected as the basic framework of the system in the development of the system [17-18].

2.2. Distributed System Architecture Design

The master-slave architecture pattern is shown in Figure 2. It is a distributed architecture pattern based on the idea of divide and conquer. It can use the central node to split a single complex task into multiple simple and semantically identical subtasks, and assign them to each child node for processing. Finally, the processing and calculation results of each child node are integrated and output. This architecture mode makes full use of the resources of the child nodes, and improves the computing performance of the system by adopting the parallel operation mode. At the same time, due to the utilization of multiple child nodes, the fault-tolerant processing of the system is enhanced, and the reliability of the system is greatly improved. As the central node, the master is mainly responsible for task scheduling and allocation. There can only be one master; As a sub node, slave is mainly responsible for the calculation of each sub task. It can run multiple nodes at the same time to build a cluster and improve the parallelism of calculation.

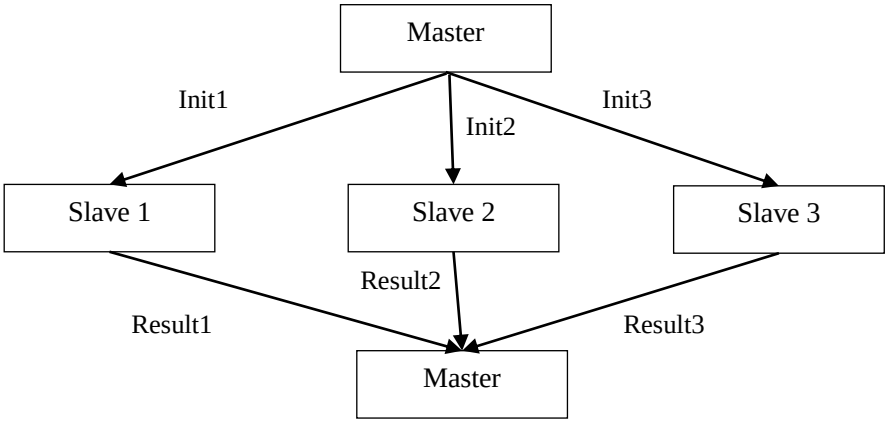

Figure 2. Master-slave architecture schema diagram

The platform adopts the above master-slave architecture to realize the distribution of tasks and the summary of results, and uses local resources or IAAs services to build a SaaS based decision-making platform with the help of cloud computing. Users do not need to care about the underlying deployment and hardware allocation of the system, but only need to be familiar with how the platform operates. Users can fully access the computing services of the decision-making platform through the browser. The architecture diagram of the decision platform is shown in Figure 3. The system is logically divided into master and slave nodes. The master node contains task processing components and routing scheduling components. The slave node can be regarded as different physical machines, with independent CPU, GPU and memory. It is mainly used as a slave node for asynchronous parallel computing. The interaction between the master and slave clusters is a network-based message passing interaction. The messages here can be either calculation data or configuration information, or simple control commands. This requires the business itself to control the specification of message interaction.

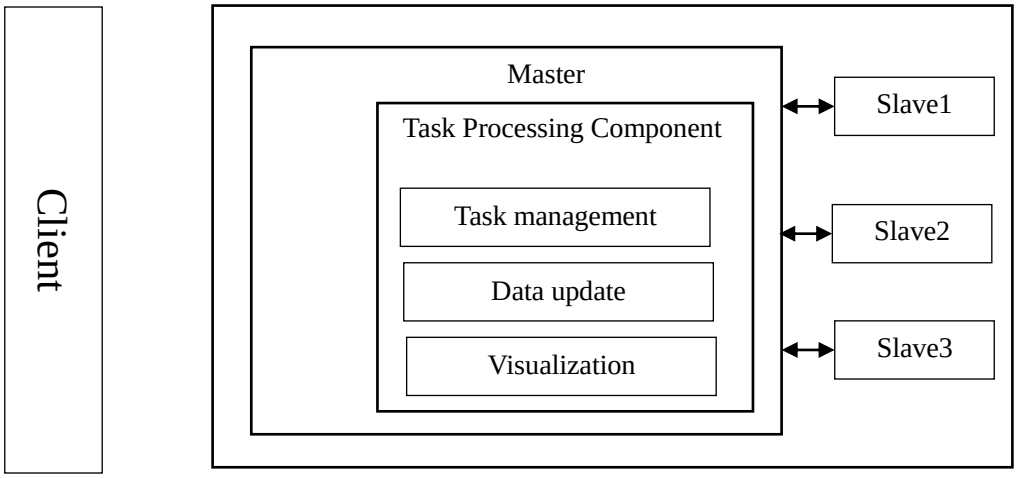

Figure 3. Decision platform architecture diagram

The master node is the focus of the design of the decision-making platform and serves as the entrance for users to access services. It has two core components for task management and sub node management: the task processing component is mainly responsible for the control and management of task status, including business sub functions such as uploading data, configuring tasks, starting tasks, stopping tasks, and summarizing results. It also provides external service access interfaces, Users can access the decision-making platform and call services through the visual interface; The core function of the route scheduling component is the scheduling function. It not only needs to correctly schedule and distribute the tasks to the appropriate slave nodes, but also needs to monitor whether the status information of the child nodes is normal at all times to avoid errors. Besides, the registration, logout and information interaction functions of the child nodes are directly controlled by the module.

The slave node is mainly responsible for the calculation of specific tasks and is an extensible part of the decision platform. Through the container technology, the platform shields the underlying implementation of the algorithm, and the user can freely select the appropriate algorithm or language platform, on the premise that a unified interface and message specification are reached with the master. In order not to affect the interaction between the master and slave nodes, the calculation process of the child nodes should be conducted asynchronously, and the current status, calculation progress and intermediate results should be reported to the master node at regular intervals. In addition, in order to realize the SaaS of the platform, the decision-making platform is required to realize four characteristics, including Internet characteristics, multi lease characteristics, service characteristics and extensibility characteristics.

3. Distributed System Environment Configuration

3.1. Hadoop Cluster Construction

During the large-scale data information processing implemented by Hadoop cluster, it shows advantages such as high efficiency and high fault tolerance, which is also an important reason why the system chooses it as the algorithm calculation and processing of student body test data. It is composed of a single master management node and several slave data nodes.

3.2. Environment Configuration

The Hadoop cluster of this system is built by creating four CentOS virtual machines on the VMware Workstation of this machine. The core configuration of this machine is i7-10700 CPU processor and 32GB running memory. Among the three virtual machines, one serves as the master node, and the remaining two or three serve as slave nodes. Each virtual machine selects application bridging to realize functional connection support. Hadoop 2.8.4, JDK1.8 and idea 2019.3.4 will be selected as the software platform of the system experiment. The planning of Hadoop cluster nodes is shown in Table 1.

Table 1. Node planning for the Hadoop cluster

Node name	Host name	IP address
Master	BS1	192.168.235.110
Slave1	BS11	192.168.235.111
Slave2	BS12	192.168.235.112
Slave3	BS13	192.168.235.113

4. System Performance Test

System performance test is an important part of the test. Good system performance can make the system more reliable in operation and improve the user experience. In the performance test, the system response time, memory and CPU utilization are important detection indicators. Here, the LoadRunner system test tool is used to test the system performance. The performance test of the system is shown in Table 2.

Table 2. System performance test

Test type	Operation process	Expected results	Test results
Test of stability	Simulate 1000 users, log in to the system every 3 seconds, and run for 5 hours during the system	The system is running properly. The CPU memory usage does not exceed 80%	Pass
Testing for concurrency	Simulate 1000 users and log in to the system at the same time. The system will run for 5 hours	The system is running properly. The CPU memory usage does not exceed 80%	Pass

It can be seen from the results in Table 2 that the system has good stability and compression resistance in performance.

5. Conclusion

This paper uses the framework of sring + spring MVC + mybatis to develop the B / S architecture of the system, and through building Hadoop distributed cluster, and using its

MapReduce distributed computing framework to complete the parallel implementation of some algorithms in the designed model, so that the system can be more efficient and reliable in processing the algorithm calculation of large-scale physical test data. This paper completes the system requirements analysis, including functional and non functional requirements analysis. It lays a good foundation for the design and implementation of the follow-up system. This paper completes the fully distributed setup and deployment of Hadoop cluster. By establishing four nodes on the virtual machine through bridging, the master-slave structure deployment of the cluster is completed, and the designed parallel algorithm can be run in the cluster.

Funding

This article is not supported by any foundation.

Data Availability

Data sharing is not applicable to this article as no new data were created or analysed in this study.

Conflict of Interest

The author states that this article has no conflict of interest.

References

[1] Oulhaci T, Omar M, Harzine F, et al. Secure and distributed certification system architecture for safety message authentication in VANET. Telecommunication Systems, 2017, 64(4):679-694.

[2] Elsen R, Nashrulloh M R, Cahyana R, et al. Microservice architecture design for autograder using distributed architecture. IOP Conference Series: Materials Science and Engineering, 2021, 1098(3):032083 (4pp).

[3] Elhasnaoui S, Chakir A, Chergui M, et al. Communication System Architecture to Integrate Distributed Systems of an IT GRC Platform Based on Agent Technology and Web Services. International journal of soft computing, 2017, 12(2):86-95.

[4] Jahn U, Wolff C, Schulz P. Concepts of a Modular System Architecture for Distributed Robotic Systems. Computers, 2019, 8(1):25-25.

[5] MD Qutubuddin, Gibo T K, Raju B S, et al. Brain Affective System Inspired Control Architecture: An Application to Nonlinear System. IEEE Access, 2021, PP(99):1-1.

[6] Shitole A B, Suryawanshi H M, Talapur G G, et al. Grid Interfaced Distributed Generation System with Modified Current Control Loop using Adaptive Synchronization Technique. IEEE Transactions on Industrial Informatics, 2017, PP(5):1-1.

[7] Vlasov A V, Varyukhin A N, Brailko I A, et al. Modelling of aerodynamical interaction of the wing and propellers of a distributed propulsion system. Journal of Physics: Conference Series, 2021, 1891(1):012012 (12pp).

[8] Alam A, Pant V, Das B. Optimal placement of protective devices and switches in a radial distribution system with distributed generation. IET Generation Transmission & Distribution, 2020, 14(21):4847-4858.

[9] Pereyra F, Valencia G, Cazarez N, et al. Implementation of a Distributed Optimal Predictive

Control in a Quadruple Tank System. Latin America transactions, 2019, 17(01):135-146.

[10] Sz A, Xiang L A, Bz A, et al. Multi-objective optimisation in flexible assembly job shop scheduling using a distributed ant colony system. European Journal of Operational Research, 2020, 283(2):441-460.

[11] Amudhavel J, Kathavate P, Reddy L, et al. Assessment On Authentication Mechanisms In Distributed System: A Case Study. Journal of Advanced Research in Dynamical and Control Systems, 2017, 9(12):1437-1448.

[12] Faiya B A, Athanasiadis D, Chen M, et al. A Self Organizing Multi Agent System for Distributed Voltage Regulation. IEEE Transactions on Smart Grid, 2021, PP(99):1-1.

[13] Acquah M A, Chen N, Pan J S, et al. Securing Fingerprint Template Using Blockchain and Distributed Storage System. Symmetry, 2020, 12(6):951.

[14] Brower R L, Mokher C G, Jones T B, et al. From Democratic to "Need to Know": Linking Distributed Leadership to Data Cultures in the Florida College System. AERA Open, 2020, 6(1):233285841989906.

[15] Raj B S, Chandrasekaran S. Optimal distributed intelligent traffic system for road safety, emergency vehicle clearance using hybrid optimisation algorithm. International Journal of Services Operations and Informatics, 2019, 10(2):122.

[16] Muzammal M, Qu Q, Nasrulin B. Renovating blockchain with distributed databases: An open source system. Future Generation Computer Systems, 2019, 90(JAN.):105-117.

[17] Kumar S A, Bhimarasetti R T. Multiple Distribution Generation Location in Reconfigured Radial Distribution System Distributed generation in Distribution System. IOP Conference Series Earth and Environmental Science, 2018, 164(1):012011.

[18] Dickinson P S, Armstrong M K, Dickinson E S, et al. Three members of a peptide family are differentially distributed and elicit differential state-dependent responses in a pattern generator-effector system. Journal of Neurophysiology, 2018, 119(5):1767-1781.