A Heterogeneous Distributed Computing Model Based on Web Service

Haitao Li
College of Information Technology, Shangqiu Normal University, Shangqiu 476000, China

Abstract

Based on the in-depth study of the existing database synchronization model, in order to improve the cross platform ability of the system and facilitate the construction of small and medium-sized enterprise information platform, this paper proposes a heterogeneous distributed computing scheme based on Web service. The scheme uses JMS to realize the message transmission between systems, and uses web service technology to realize cross platform data reading and writing. In the aspect of distributed transaction processing, the two-phase commit protocol is improved to reduce the probability of system deadlock and effectively ensure the consistency of distributed database data. In order to improve the performance of distributed database system, cache technology is introduced, and the way of integrating cache and database transaction processing is proposed, which effectively ensures the validity of cache data. The architecture is oriented to program developers, who can develop efficient and convenient distributed database system on the basis of this architecture. Finally, this architecture is applied to the background management system of mobile express service. The running results show that the architecture can well meet the business requirements of distributed heterogeneous database system synchronization.

Keywords: Web service, heterogeneous data, distributed computing, distributed database.

I. Introduction

At present, most application systems are isolated, and it is difficult to exchange and share information between systems, and it is difficult to ensure the consistency of data [1]. How to realize resource sharing and seamless integration of applications across platforms, languages and hardware are the problems to be solved. To solve the problem of system interoperability in this distributed and heterogeneous environment, the traditional solution uses distributed object technology [2]. At present, the most commonly used distributed object technologies include: Sun Java RMI (remote method invocation), OMG CORBA (Common Object Request Broker Architecture), Microsoft DCOM (Distributed Component Object Model).

All of these distributed object technologies have their own private formats and are highly efficient. However, this tightly coupled architecture limits the reusability and flexibility of services [3-4].

(1) Tight coupling: the client and server must be tightly coupled, that is, the same basic structure is required. Therefore, once the interface or execution mode of the server changes, the client will not be able to run, resulting in the collapse of the system.

(2) System integration: each distributed technology defines its own data format and transmission protocol, which requires the client and server to load appropriate middleware on their respective machines. This method ensures the consistency of sharing the same middleware between programs, but it hinders the interoperability between programs using different middleware, making it difficult to realize system integration.

(3) Local computing: there are limitations of "local computing". Due to the lack of existing protocols, there are some deficiencies in the communication between client and server. Most organizations join a firewall between the Internet and their local area network. Distributed object protocols such as DCOM and CORBA use the method of dynamic port allocation, which makes it difficult for all communication based on this protocol to pass through the firewall. In short, the traditional distributed object technology does not use the existing protocol, but defines a set
of data format and transmission protocol, which makes the information sharing and integration between enterprises more difficult.

Therefore, a new form of distributed technology arises at the historic moment, namely the so-called web services. Through it, we can build a general platform independent and language independent technology layer on the basis of various existing heterogeneous platforms [5-7]. Applications on different platforms rely on this technology layer to realize mutual connection and integration. Users don't care what operating system platform developers use, nor what language developers use. They only care about what kind of services developers can provide. They only need to get the services they need anytime, anywhere. This loose coupling characteristic of Web services is the biggest advantage of Web services compared with traditional distributed computing environment, and it is also the basic condition for web services to be used in heterogeneous and complex Internet environment. Therefore, the use of Web services will bring great changes to the heterogeneous distributed computing technology.

II. Web services architecture

Web services use SOA (Service Oriented Architecture) architecture. The architecture consists of three roles and three basic operations. The three roles are service provider, service requester and service agent, and the three basic operations are publishing, searching and binding. The service provider publishes its services to a directory of the service broker. When the service requester needs to call the service, it first searches the directory provided by the service agent to get the information of the service, and then calls the service published by the service provider according to the information. In Web Services Architecture, WSDL is used to describe services, UDDI is used to publish and search services, and soap is used to perform service calls. The relationship between them is shown in Figure 1.

Fig 1: Web service model

The roles in the web services architecture include [8-10]:

(1) Service providers are responsible for using WSDL to describe and define the functionality of applications that can be accessed programmatically. Publish service information in UDDI Business Registry (service agent) so that it can be accessed by other applications.

(2) The service requester is responsible for finding the required service through one or more service agents, downloading the WSDL document defining the service, binding and invoking the service provided by the service provider according to the service definition, and executing these services through the service provider.

(3) The service broker is responsible for publicizing the service published by the service provider and allowing the service requester to query the service definition published in the service broker. The service agent acts as the matchmaker between the service provider and the service requester, which is the same as the yellow page of the phone book. When the service requester finds the required service in the service proxy, the next interaction work only needs to be carried out between the service provider and the service requester, and the service proxy will exit.
the service access and call workflow.

The operations in the web services architecture include:

(1) Publishing: the interaction between service providers and service agents. When the service provider publishes the service in the service agency, it will naturally promote the registered service in the whole web community. The details of publishing API depend on the implementation process of service broker.

(2) Lookup: the interaction between the service requester and the service agent. The service requester describes the search criteria such as service type and quality of service, and the service broker finds the matching service from the published service. The function and quality of the search are determined by the implementation scheme of the service broker.

(3) Binding: the interaction between service requester and service provider determines the runtime relationship between them. The operation function can be static or dynamic. When you use the static model, you need to write a client application for the service requester. When using the dynamic model, the server generates the client agent based on the service definition at runtime to call the service.

The web services life cycle is shown in Figure 2. The web service provided by the service provider is published to the UDDI registry, the WSDL definition is created in the service provider, and the link of the definition is published. The web service finds the required service for the service requester, and obtains information about calling the web service from the published WSDL definition. This includes downloading the WSDL definition to the service requester and deserializing the WSDL into Java classes or other languages. The service requester is bound to the web service at runtime. The application of the service requester uses the java interface represented by WSDL as a proxy to bind to the web service. The service requester invokes the web service according to the service invocation information extracted from the WSDL definition of the web service.

![Fig 2: The web services life cycle](image)

III. Distributed computing model based on web service

Enterprise services may run on different platforms and be implemented in different languages. However, no matter what kind of services are, they do not want to be exposed directly for the sake of business. Usually, they only want to expose some of the methods. Web service proxy pattern is a kind of exposing one or more services that can be accessed through XML and web protocol, and forwarding the service requests to the real service components. The granularity of web service reflects the encapsulation scale of web service to business logic. Web service proxy is a coarse-grained service component exposed in the form of web service. Web service proxy is mainly the design of service operations. After the operations are designed, the parameters and return values of these operations should be defined, and the possible errors or exceptions should be considered. Web service proxy hides the details of calling web service. Jax-rpc packages web service calls into a proxy class, which exposes an interface to hide the details of web service calls from clients. Figure 3 illustrates the structure of this pattern.

Among them, endpoint processor is the first entry for the client to contact the web service, which is responsible for receiving and processing soap requests based on HTTP. It is built into the jax-rpc runtime environment and is
completely transparent to developers. It is also a servlet based on jax-rpc terminal, which is responsible for processing external requests and forwarding them to web service broker. Web service broker is based on a WSDL file and acts as a proxy for one or more services. Figure 4 illustrates the interaction of the Web service proxy mode. First, Client sends a request to EndpointProcessor, EndpointProcessor extracts the request information, and then calls WebServiceBroker. Then WebService broker calls specific business services to complete business operations.

![Fig 3: Web service proxy class diagram](image)

![Fig 4: Web service proxy interaction diagram](image)

There are more customers in Web services than in general client server architecture. Therefore, we must pay attention to its performance problems. There are two cases that will lead to performance degradation: client applications often access the same information of server, and the implementation method is to call web services every time. A large number of client applications access the same data of server by browsing at the same time. But the actual cost of web service call includes: network delay, server CPU operation time and database server SQL operation delay. If we consider these factors, the efficiency of web service invocation is even more unacceptable.

Web services can achieve the maximum performance by using data cache correctly. When the request information of the service is mainly read-only, or when the request information changes more slowly than the required rate, it is a better choice to use caching in Web services. The purpose of using caching technology is to reduce the number of web service calls as much as possible. The solution of this paper is to cache the data with high access frequency in soap client.

(1) Buffer management mode: for web services supporting HTTP, soap clients can cache processing results for future requests to save response time of Web services. In fact, caching technology is suitable for read-only services, which perform queries on the database or provide access to relatively static content, such as patient
history diagnosis information, drug information, historical financial data, etc. For the cache content, we can design a cache with large storage capacity, which can save the most used data items and clear the oldest and least used data items. Figure 5 illustrates the logical process of soap caching through a use case diagram, which describes four use cases around managing a soap cache.

Soap client first accesses soap cache before an RPC call to remote business service. Soap cache can be implemented with hash table in the physical memory of soap client for quick access. If the cache data entry is available and the cache data entry has not expired, the soap client will get the business service information from the soap cache. It is fast to get the cache data from the physical memory of soap client without routing the service request to the remote server, which will greatly save CPU and network resources. If the data entry has expired or there is no cache data needed by the client, the soap client generates a soap request in the form of an XML document and sends it to the soap server. Then, the soap server initiates an RPC call to the business service application. After the corresponding processing, the confirmation or query details are returned to the soap client, and the cache events and business transactions are logged for tracking and management.

Soap cache intercepts a soap request and checks whether the data entry is cached before getting the cached data or calling the soap server. This has a certain risk, that is, if the content of the application is updated or changed, how to ensure the integrity and synchronization of the data. At this time, the synchronization of data is very important, otherwise it will get old data. Here's how to keep the data in sync.

(2) Data synchronization: in this paper, the listener mode is used to achieve data synchroniz

IV. Overall design and implementation

From the perspective of Web services, the business functions of DC are encapsulated as web services and published on Apache soap server. The web service on soap server mainly deals with business logic, and the
connection with underlying database is completed through JDBC. From the perspective of client, the system is divided into three layers: presentation layer, business layer and database layer. This is in line with the web service architecture, but also in line with the idea of three-tier architecture, with web services as the middle layer, which provides a unified business logic and realizes the separation of business and database storage.

Figure 6 shows the JAX-RPC-based web services application architecture. In order to call a remote business service, the client needs a "stub" to enable the client to communicate with the remote application service through the remote "skeleton". His system calls web services through soap over HTTP. The client application uses remote procedure call to call the method on the remote service. The client sends the request message to the server using XML based message protocol. The API provided by JAX-RPC can be used to realize data type mapping between Java and XML. That is, when the client calls JAX-RPC, its XML parameters will be mapped to Java objects. Similarly, when it returns, Java objects will be mapped to XML elements as return values. The specific process steps are as follows:

1. Client application calls local stub: object;
2. Stub object calls program through JAX-RPC system;
3. The runtime environment serializes Java objects into a SOAP message, and then sends the message to the web service endpoint through HTTP protocol;
4. When the web service receives the soap request, the JAX-RPC runtime environment on the server side deserializes the SOAP message into Java classes;
5. Call the method in the skeleton object;
6. Skeleton calls the method to realize the specific service;
7. After processing, the system converts the result of calling method into SOAP message response to customer;
8. The client receives the SOAP message from the response, analyzes it and gets the result of the method call.

The design of the JAX-RPC service is independent of the type of client the application will access.

![Diagram](attachment:conversation.png)

*Fig 5: Simulation result showing energy efficiency of handover algorithm*
V. Conclusion

This paper studies the application of Web services based on JAX-RPC from the perspective of practical projects. Firstly, the development tools and environment are introduced. Secondly, the system requirements are analyzed. Finally, JDK, Tomcat, axis and other tools are used to complete the construction and deployment of Web services based on JAX-RPC. This paper uses two design patterns to optimize the system. Axis is used as the runtime environment of Web services based on JAX-RPC. Using Tomcat server and IE browser to complete the construction and deployment of Web services based on JAX-RPC.

Acknowledgements

This paper was supported by the science and technology research projects in 2018 of Henan Province, China(Grant No 182102410064), This paper was supported by the Science and Technology Department of Henan Province (Grant No 202102210158), and was supported by the Henan Education and teaching reform research and practice project in 2019(Grant No 2019sjglx399).

References

[1] Long Gang, Xiao Lei, Chen Xuequan. Overview of Curvelet Transform in Image Processing. Computer Research and Development, 2005,42 (8): 1331-1337
[2] Yuan Hongbo, Zhao Nudong, Cheng Man. Research Progress and Prospect of Field Weed Recognition Based on Image Processing. Acta Agriculturias Sinica, 2020, V.51 (52): 330-341
[3] Zhang Guoping, Li Yali, Xu Xiangyi. Research on Application of Image Processing in Partial Discharge Ultraviolet Imaging Detection of Power Equipment. Computer Programming Skills and Maintenance, 2020, No.425 (11): 146-148
[4] Cheng Zheng Xing, Lin Yong Ping. Application of Wavelet Analysis in Image Processing. Acta Mathematica Sinica Sinica, 2001, 18 (f12): 57-86
[5] Bai Lifen, Xu Yuxian, Yu Shui. Study on Auto Focusing Method of Microscope Based on Image Processing. Journal of Instrumentation, 1999 (06): 64-66
[6] Chen Bingquan, Liu Hongli, Meng Fanbin. Current Situation and Development Direction of Digital Image Processing Technology. Journal of Jishou University (natural Science Edition), 2009 (01): 63-70
[7] Yang Ming, Song Xuefeng, Wang Hong. Image Processing for Intelligent Transportation System. Computer Engineering and Application, 2001, 37 (9): 4-7
[8] Ji Shouwen, Wang Rongben, Chen Jiajuan. Identification of Weeds in Maize Seedling Field by Using Computer Image Processing Technology. Acta Agriculturae Sinica, 2001,17 (002): 154-156
[9] Jiang Haiyang, Wang Huiwen, Guo Chunrong. Research Progress of Tongue Image Processing Technology. China Science and Technology of Traditional Chinese Medicine, 2021, Vol. 28, No. 1, 164-166, ISTIC CA, 2021
[10] Huang Jin, Zhang Xiaobing, Yin Hanchun. 3d Measurement Method Based on Image Processing. Electronic Devices, 2002, 25 (4): 364-368