Advances in Computer Science Research, volume 78

3rd International Workshop on Materials Engineering and Computer Sciences (IWMECS 2018)

Analysis on the Distributed Computer Software Bus Architecture

Fang Liu

Xi'an Shiyou University, Xi'an, Shaanxi, 7100654

Keywords: Distributed Computer Software, Bus Architecture, Analysis Method

Abstract: With the development of computer technology and software applications, it has gradually become an important tool in our daily life and in the development and production of various factories. At present, there are still some problems in software development and application. Software development The way also urgently needs to be changed. Therefore, the research on the basis of computer in design mode is put forward. According to the thought of computer software bus, CORBA is adopted as the main standard to construct the system and development thought of computer software bus, The implementation of each module, the realization of the application of JAVA and C + + language, the preparation of the client and server communication between the composition of the final idea to improve the computer cross-language communication, the actual testing, detection good results, with strong The communication effect and solution to past problems of software. </s>

1. Introduction

The application software developed by the traditional method is often a kind of independent and integral system. Software developed in advance through needs analysis and design has various functions and features linked together in a fixed way. An application program as a whole integrates a wide range of usage features prior to release. However, many of these features can not be removed, upgraded, or replaced independently. For other applications, even using the same programming language, and running on the same machine, it is difficult to use the program's data and functionality. Different applications are like strangers, completely separated. With the traditional software development technology, the huge software makes the development more and more difficult, the development cycle is longer and longer, the maintenance cost is higher and higher, and the expansion of functions is a dangerous and difficult task. Application systems are heavily dependent on the operating system and on specific web services, making them poorly open. Also, the structure of the operating system is not modular enough to make it easy to overwrite, upgrade, or replace some service functions in a very concise manner. Both the server application and the client application are of course greatly limited by this environment. Even if the application itself has a strong openness, the applications may also differ in the way they are provided by the operating system and the network Forms show in front of other applications. Developers for insult subjects, different application environments (local applications, web applications, different operating platform applications) to develop different code. The programming paradigm for software development strongly depends on how the service is provided, such as services provided in the same address space via dynamic linking, services provided by different processes on the same machine, services obtained from the operating system, or across the network Services running on another machine. In summary, the design and implementation of complex software is still very expensive and prone to error. Much effort and cost are focused on the same design concepts and the repeated development of code components. However, the diversification of the hardware structure, the complexity of the operating system and the diversity of communication platforms make it difficult to design a correct, portable, efficient and low-cost application from scratch.

2. Distributed Computer Software Bus Architecture Framework Design

The software bus provides a transparent mechanism for communication between components.

The bus provides a consistent interface specification to the outside world, and each newly added component must register its functional service information with the bus and deregister its functional service information when it exits.When a component wants to request Another component of the service, the bus is responsible for querying the functional service information base, locate the components that provide this service, and send this service request.In order to call the remote object instance, the client first get the object parameters.Client used in the remote request The same code as the native request replaces the remote instance with the object parameters. Figure 1 shows a remote invocation procedure. When the ORB detects an object's parameters and discovers that the object is a remote object, it invokes these parameters to transfer the request to the network ORB connected to the remote object. ORB can determine from the object parameters of the target object is remote, the client can not. When the client requests, the object parameters do not distinguish the location of the target object to ensure the transparency of the location These basic principles of COR-BA simplify the design of computer applications for distributed objects.

Plug and play, openness, transparency, reliability and versatility are the tenets of software bus design and development. The underlying software bus is based on IIOP protocol and has the ability to be used on the Internet and Intranet. The upper layer provides users with simple and flexible Message software bus application program interface, easy to program.According to the introduction of CORBA model software bus, multi-layer way to manage, its role is to achieve distributed object calls to adapt to the needs of network monitoring and management.Using the software bus layer of the other One of the aims is to meet the scalability of network information service management system. When new network services are added, it is easy to integrate the new application with the original system as shown in Figure 2. Follow the principle of open design , Providing "plug-and-play" services to applications. Through the communication modules in the bus, any application, regardless of its functionality, can be directly integrated into the system environment as long as the "bus" interface standard is followed. Other applications perform various types of information interaction to achieve data integration and inter-module communication.

The composition of the distributed computer software bus structure shown in Figure 3. The overall structure of a common core as the center, the core includes a Portable Object Adapter (POA), communication transmission protocol (IIOP, GIOP), etc. The upper part of the structure for the development of components Workstations, the left side of the component testing, calculation, development, management system, the right side of the component storage system, the component storage database. Software components may have multiple IDL interfaces, these common interface components for assembly, the interface interface is To handle interface communication interfaces between different language application systems or different language components, including the IDL com-pile and CIDL compiler, the application program API is connected to the right side of the interface The left side of the core is the security interface for application systems and software buses using Secure Sockets Layer Protocol (SSL) and firewall technology. The naming and notification services are on the right. The lower part of the Common Core is the various services at the heart of the software bus communication: Object Transaction Services OTS, Persistence Service PSS, Value Objects ObV, Software Bus Management Control, Directory Services, Authentication Authorization Management. After the completion of the development system integrators, system will be tested and put in an application repository.

3. Software Components Management

Components as the basic elements of the construction system, can be divided into system components, common components and specialized components.Construction components refers to the system development environment or support components of the environment, such as visual development tools in the form, command buttons. It is generally smaller particle size, reusability is better; generic components refers to a variety of applications with greater reuse value of functional components, such as data query components, data statistics components; The functional components designed and developed by a certain type of system have some reusable values in their fields, such as the production quotas and production planning components in the machining industry,

which generally have a large granularity and require a high level of abstraction. After the component is obtained in a certain way, the component library (mainly composed of the basic feature library, interface feature library and network feature library) performs a verification process on the component to ensure that the component meets certain quality standards and then classifies the component And store it in the component library.Users through a specialized search system in the component retrieval library (by component relationship table, component name table, key word table , Component type index table composition) to find the components to meet their needs, if found, it will be combined into a new application.

In the process of assembly of the application system based on open software bus, it is a key link to realize the call of software components in different programming languages.Components usually encapsulate their internal specific implementation functions during development, through the component interface definition language CIDL (compo- nent implementation definition language) on the software bus to connect with a specific application to complete the message sending, providing services or access services, etc. As shown in Figure 5, the open software bus through the CIDL compiler on the upcoming application programming interface Describe and generate the application framework required by the client program, such as authentication, interface description, stub class and so on.

CIDL compiler is divided into C + + CIDL compiler and JavaCIDL compiler, but also generate the corresponding C + + language or Java client stubs, and then write the client file, compiled with the server-side communication. Therefore, CIDL and software Bus is a component and server communication mechanism, and component retrieval is usually done using the SQL (select query language) language.

The design of component assembly structure based on software bus should emphasize the openness of the system and the independence of each component. Development methods of components are usually developed by "black box", each component has its own specific function, The interface part is mounted on the software bus and connected with a specific application to complete the functions of sending messages, providing services or acquiring services, etc. The unified software bus standard and component standard are the prerequisites for implementing the application system assembly.

Component structure is the inheritance and extension of object-oriented structure, it has all the advantages of object-oriented structure. At the same time, in this model, the component is a well-packaged functional components, external consistent interface, communication between components by the bus Unified agent, reducing the interdependencies between components. Software developers can easily define and build new plug-and-play components to expand the system. In order to command, for example, if you want to add new components to the system based on the existing order Of the command components, as long as the command interface components according to the established interface specification to define its various states, inserted into the system, the implementation of automata will be executed as an internal command automatically. Application components to increase more easily, and even dynamic load during system operation And uninstall. Application software package (invocation & encapsulation) allows the application software module into the system's software bus, and by all types of users automatically start the application as needed.In addition, taking into account the Web site is constructed on the distributed component library, Its wide range of areas and the large number of components, so the software components and code into two independent Points, each component library does not save the component code itself, but only save the description of component features and components of the actual storage location of the URL address through the URL address to establish component information and the actual code file remote link (this feature can CORBA IIOP communication and ORB remote call method,

4. Conclusion

The idea put forward by Woodfun is beneficial to improve the long development cycle and high maintenance cost in the traditional software development. The application must rely on the

disadvantages of the operating system and programming language and the disadvantages existing in the modern component-based software development, such as programming language The dependence and the development of stand-alone and other shortcomings. In the application of distributed object technology, the defects that are limited to middleware at home and abroad are improved. For the first time, distributed technology combined with component development is applied to protect software development technology and make software development technology more adapt to the development of network technology. In the improvement of component development method, we abandoned the stand-alone in the simple component development method, and developed and applied the component development in the network development, which is beneficial to improve the efficiency and quality of software development, improve the software development technology and reduce the development cost.

References

[1] Zhou Yue, Wang Hong. Software Bus Technology and Its Application in Enterprise ERP [J]. Computer Development and Application. 2006 (03)

[2] Software Bus at East Alpine School [J]. Software Engineer. 2001 (04)

[3] Sun Zhian. Software Bus: Architecture Analysis and Design [J]. Command and Control and Simulation. 2009 (02)

[4] Li Jianhong, Zhan Chuanjie. Development of control function modules in software bus architecture [J]. Microcomputer Information. 2009 (30)

[5] Wu Kehe, Song Min, Zhou Jing. Software bus technology based on multi-agent system [J] .Power System Technology. 2007 (S1)

[6] Wang Hui, Cheen Shengjian, Hu Xiang. Role-Based Rights Management in Software Bus System [J] .Electric Power Education. 2005 (S2)