2011 International Conference on Computer Science and Information Technology (ICCSIT 2011)
IPCSIT vol. 51 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V51.13

A Network Services Management Middleware Architecture Model

Zhongqiu Song[a], Zongshui Xiao[a], Fangfang Wu[a]

[a]School of Computer Science and Technology Shandong University,Jinan,250101,China

Abstract. As the coupling degree between network service management system and network element is so high that the system has poor reliability, flexibility, expansibility. This paper propose a network service management middleware model, which adopts Service- oriented architecture, provides plug and play service deployment mechanism meeting JBI standard. Middleware provides necessary core management functions, makes use of thin terminal user interface technology, accomplishes the flexibility and mobility of network service management. The model has been very good in the campus network application.

Keywords: Network Services Mangemwnt, Java Business Integration, Middleware, Service-Oriented Architecture

1. Introduction

With the rapid development of Internet, the network management is developing in a large-scale, distributed and collaborative direction. As the new technologies of cloud computing, virtualization, service-oriented architecture (SOA), middleware etc emerge, the network heterogeneity and complexity not only increase, but also the services show diversity. In order to ensure good network performance and reliability and provide users with high quality services, it's necessary to adopt effective and feasible method to manage these network services. In the new management environment, the traditional centralized network management can't meet the needs and must break through the traditional network management system in the management model and management functional limitations to meet the distribution and intelligent network management features. To meet the changing needs of network management applications, an easy expansion, easy maintenance and customized network management services is provided. Administrators only concern with how to manage, rather than concern about how to achieve. Administrators can management of the implementation of network services by terminal device accessing the Internet anytime and anywhere, and achieve in the "cloud" of management. This paper presents a network services management middleware architecture model that uses service-oriented architecture, uses of IT service management(ITSM) management ideas, learns from the cloud idea [1] "everything-as-a- service" , provides plug and play component framework , is an open, component-based service composition and interoperability platform, and improve the reusability ,integration ,flexibility and scalability of network services management applications, so that it can quickly build on demand, reach the network service effective management and simplify network service management system development. The network conditions can be managed through thin management terminal in a network environment.

2. Related work

ISO20000[2], based on customer-center and IT operation and maintenance management business, is the first IT service management standards by recognized industry .Tele Management Forum (TMF) has done lots of related work and formulated a series of specifications and guidelines. The QoS definition in ITU-T E.860 [3] refers to "degree of conformance of the service delivered to a user by a provider in accordance with an agreement between them". Web2.0 technologies have a good user interaction, which well support visiting

E-mail address: szq1978@163.com.

network service management by thin client in this paper. The concept of middleware is proposed for solving the distributed heterogeneous problem. Currently, there are several industry standard specifications, such as CORBA、DCOM、J2EE [4]. J2EE proposed by Sun Company is complete standard based on developing serve middleware, and has whole application specification based on Java language, released for enterprise, including Java Servlet, Java Server Page (JSP), EJB, JBI for different business requires [5].

JBI is open, message-driven, inserted frame, allow plant the third party's standard component and make them interoperability. The frame has three components: service engine (SE), binding component (BC) and normalized message router (NMR).[6] has proposed a model for cloud service management and monitoring, which only put forward the method for service resource management and monitoring and not give service integration and management method .In order to solve the traditional BPEL business process defects for binding web service static, [7] provide composing web service dynamic system frame based on BPEL and UDDI. [8] Introduce IBM service management architecture from user interface, process layer, information layer, process integration.

3. Network services management middleware framework model

Figure 1 shows the network service management middleware architecture model that uses service-oriented architecture to support the business into a set of linked services, or repeatable business tasks, use framework model of rapid assembly services to respond to changing requirements. In addition, the framework also supports the secondary development , through integrated development environment development services to meet the SOA specification, deployment of services in the form of plug-ins to the framework, assembly existing service components suitable for the business services through process planning, which will be published to the ESB business services, through the thin terminal access thin service management system to complete the network service management and monitoring.

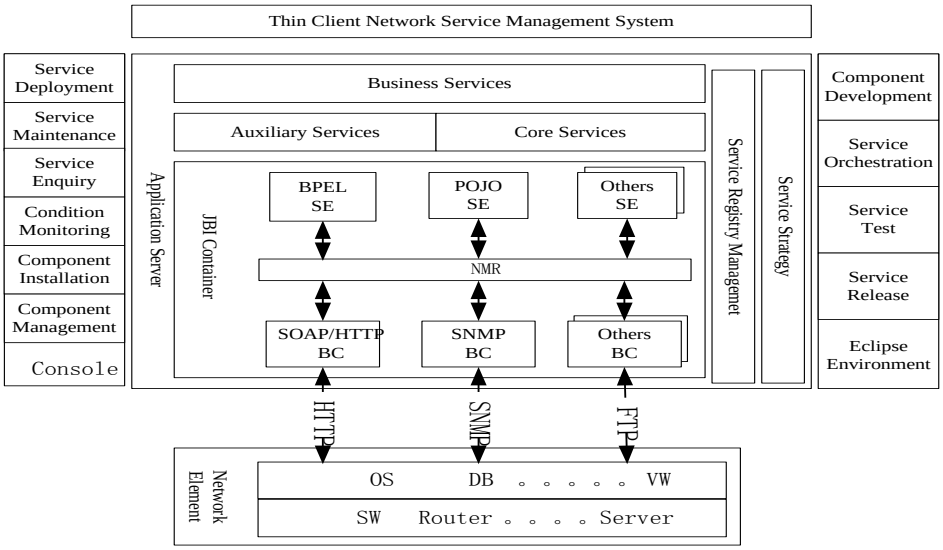

Fig. 1. Network service management middleware architecture model

Network element layer of the framework is the basis of achieving upper service, in order to achieve network service management. First of all, we need monitor network resources that need to be managed. The monitoring of quality of service (QoS) of network services resources mainly divides into four aspects: availability, performance, throughput and utilization of service. We design different indicator system for different objects from the four aspects. In order to achieve the management of network services, management middleware framework model in this paper generally is divided into four parts: the part of network element where the necessary resources data is obtained ; middleware assembled the basic Services into business services by orchestrating the process of basic service of network services management; management and maintenance console and secondary development environment; network services management system for thin client, which build business services into management application, and achieve network service management through the thin client.

3.1. Network element layer

The underlying framework is the network element (NE) layer, which is mainly composed of IaaS platforms that contain the network physical layer devices and various interfaces, and systems and applications deployed in the IaaS platform. The equipment composed of IaaS platform including servers, routers, switches, hosts, etc. Which can be called hard-NE; the applications deployed on the platform include operating system, database, virtual machine, which are called soft-NE [9]. Because these devices and applications from different manufacturers with different technical standards of production and development and different communication protocols. How to eliminate these differences makes that the network service managers do not consider these differences, avoids direct operation on the network elements and achieve simple and efficient management is a serious problem. We shield differences of the network element through the JBI in the architecture.

3.2. Data acquisition and processing

IT service quality management and monitoring system use layer data management in data management, as shown in Figure 2. First of all, the managed object model is needed to build, of which interface and attributes are called the master data. A variety of data collected by monitoring agent is called resource data, also known as real-time data, which is the source data used by analyzing and processing in the system, . By Calculating and processing real-time data, we can get business data. For example, to get the switch port traffic in real-time t time TF by the way of polling, the time stamped signature is stored in the database, if you want the port changes per unit time F in the flow , you can calculate the following equation(1):

$$F = TFt2 - TFt1 \tag{1}$$

A variety of IT resources - platforms, servers, services and applications have a certain degree of monitoring indicators, which is referred as index data. According to the indicators data and monitoring need to filter and processes resource data, we can get the raw data to be monitored, referred as monitoring data. Historical data can be get by accumulating monitoring data, and important events arise during monitoring are registered in the way of log, known as log data. System get analysis of data by analyzing monitoring data, which can be showed to user in the way of graphics, animation, alarm, etc. the data is referred to as user data.

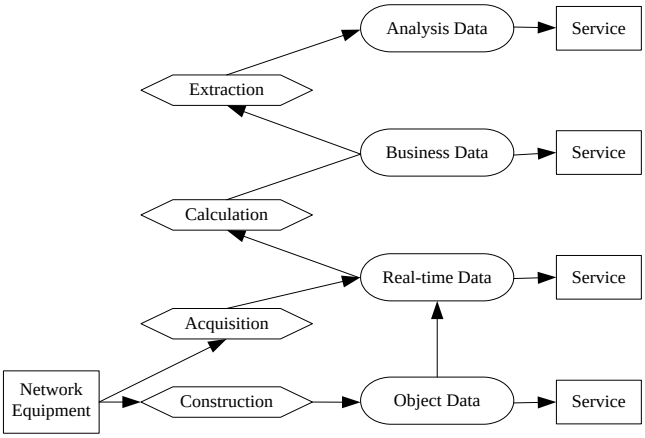

Fig. 2. Hierarchical data processing

3.3. JBI implementation environment monitoring network services

The aim of network services management is to provide stable, reliable, high-quality services for users. To achieve this goal, managers must fully understand the operational status of network service resources, which requires the monitoring of network resources to provide services together, as shown in Figure 3. Monitoring agent will introduce monitored node information into JBI container through different protocol binding components where data can be process through the engine .Data processing focus on monitoring data contained in the information and knowledge using of data mining, business intelligence (BI) analysis techniques . Engine can schedule the service consumer and services, can provide simple services, the service engine can constructer new services by aggregating other services. JBI automatically create monitor resource directory through the directory automatic discovery engine. The monitoring engine inspect the data that

returned by monitoring agency and compare alert alarm conditions set in the alarm engine to determine whether the alarm is triggered. Resource control engine communicate with monitoring agency, and control resources monitored by monitoring agent. Data processing engine can obtain the corresponding business data and analytical data by calculating, extracting and other processing of the data collected. The safety control engine is for user authentication and access control. Event management engine manage various events within the system, and log. XSLT engine complete the convert between a text file, SQL statements, HTTP message, a certain sequence of data calls, etc. and XML documents. BPEL engine complete weaving business process by deploying BPEL service unit the deployment. WS-BPEL language is used in the synthesis of this important mode of service via multiplying the processing of complex structures. POJO engine will put the basic service's POJO class into the JBI. SQL engine complete the operation of database. RULE engine is for processing the alarm occurred. All functions of JBI provide service in the form of web service externally.

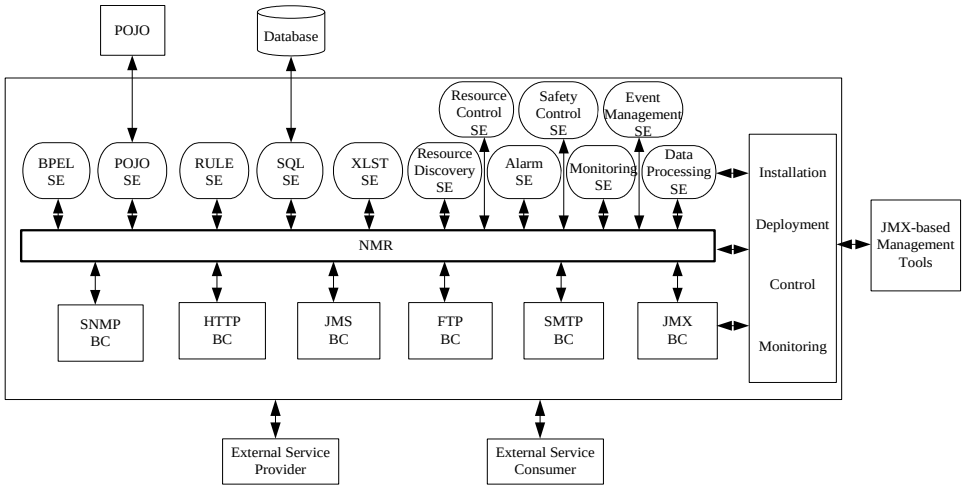

Fig. 3. Implementation of the JBI environment monitoring network services

Binding component send and receive messages based on a specific protocol and transmitter. Binding component convert between the message format and protocol-specific format so that the system only needs to deal JBI normalized message and separate between the JBI system and a specific protocol. Java Management Extension (JMX) binding components achieve management control of JBI environment, SMTP binding components implementation mail forwarding , JMS binding components achieve distributed management of network services, HTTP Binding Component achieve Web service access, SNMP Binding Component achieve monitoring resources management control and data acquisition.

The nodes outside the JBI system is external service consumers and service providers, on behalf of external entity that JBI need to integrate. These external entities can interact with JBI binding component through a variety of technologies.

3.4. Basic service

Basic services is the basis of realizing business services .In order to facilitate the composite service, this article uses simple, formal JAVA object POJO, to reduce the service size. POJO do not need to follow specific external interfaces or third-party API. To reduce duplication of code and the coupling among modules, and contribute to the future operability and maintainability, we use of "cross" idea in the programming and divide the basic services into core services and ancillary services. The ancillary services are crosscutting concerns of aspect-oriented programming(AOP), such as logging, rights management, transaction processing, persistence, etc, which emerge in many places of core service, but are resemble and not much relate with the business process. We can encapsulate public behavior of these packages into a reusable module, use a dynamic agent technology, use of intercepted messages way to replace the implementation of object behavior. Core services are the core concerns of AOP, which is the main flow of business processes. In the monitoring of network services, the network topology algorithm, resource monitoring, resource discovery is core service to realize the automatic monitoring. The POJO class of ancillary services and core services are

put up to the JBI container through the POJO engine, and make complete the service portfolio for business services through the BPEL engine.

3.5. Business services, service registration management and service strategy

The business service get by the framework model are ultimately up to enterprise service bus, the administrator call these services through the management terminal to implement network management. ESB is a product of combinations traditional middleware technology with XML, Web services technologies etc. which can eliminate technical difference between different applications, make different application servers do well and achieve communication between the different services. We publish the business services to ESB, which make these services available to the network services in the form of web service. Administrator's obtained a series of charts, animation, alarm by the thin management end and have an easy and intuitive understanding of the operation of the network.

Service registered management published function modules up to Service Broker in the way of Web Service .Manager Server find Service Broker by looking for various types of IT and get these resources management information, analysis the management information, percept and call the functional modules to achieve the purpose of reuse modules. Web services are XML-based application that supports communication between different operating systems through the network. The open standards XML, WSDL, SOAP and UDDI are used to describe a web application integrating standardized methods. WSDL describes the availability of services, XML is used to package the data, and SOAP is used to transmit data, UDDI lists which services are available. Figure 4 describes the basic model of web services [10].

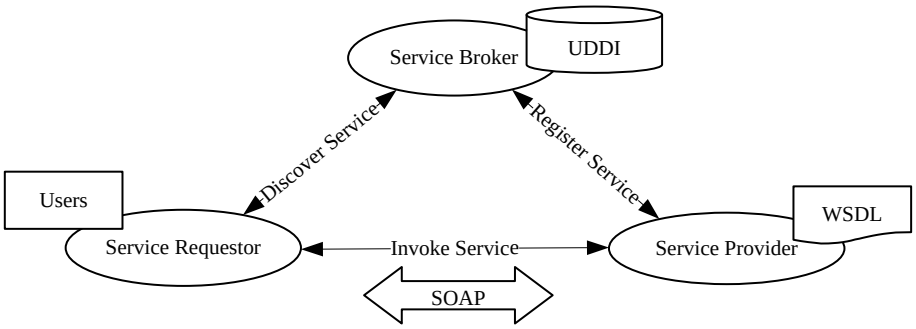

Fig. 4. Web services basic model

Services strategies automatically decompose complex business service requests into a set of linked services, or repeatable business tasks. These services can be achieved through POJO class and can guarantee the quality of service of these objects。It can adjust dynamically QoS strategies by analysis the information of the business implementation.

4. Console and the development environment

Console and development environment is an auxiliary tool of middleware, to complete maintenance and management of middleware and secondary development. Console completes the component installation, component management, service deployment, service maintenance, service inquiries, status monitoring and other middleware management. The development environment mainly completes component development, service orchestration, service testing, service releases, and other secondary development. It can meet the needs of different users through the console and the development environment, and the user can easily develop and deploy personalized service to meet their own needs, and enhance the maintainability and scalability of middleware.

5. Thin client applications

To create thin client network service management system middleware-based, thin client services complete customization and network services management through the system. Figure 5 show the network topology services to map out the campus network topology by thin client calling for adopting service management system. To complete the function of application system also needs to build an object virtual model,

performance model, computational model, analysis model and data processing models, these models will be given in another article. The main development environment is mainly used to develop thin client applications。In order to provide easy use, efficient visual development environment, we use the JavaServer Faces (JSF) new standard Java framework developed by Java Community Process (JCP) to develop a Web application. The framework not only provides Web designers for drag and drop user interface components, but also provides systems developers for the rich and robust JSF API, in order to achieve power and programming flexibility. JSF also constructed Model-View-Controller (MVC) design pattern into its architecture, to ensure that the application has a higher maintainability. Thin client is the main equipment that administrators use, so they can use a Web-enabled phone, workstations and so on to manage network services.

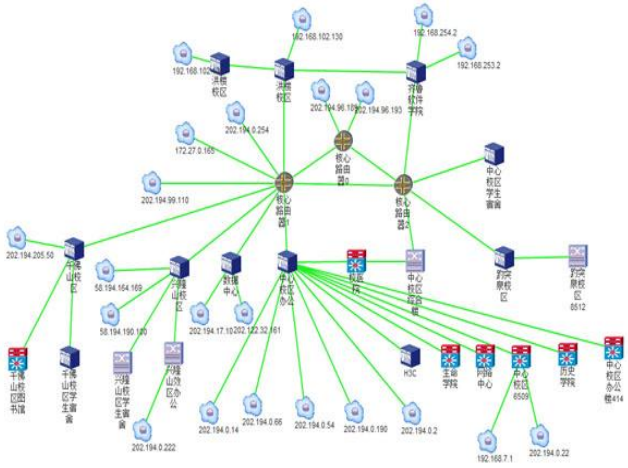

Fig. 5. Campus network topology

6. Conclusion

The network service management middleware framework model is proposed in this article，which can meet the new needs of the network service management for the rapid development of the current network ， provide a Web2.0-based self-service, to achieve the customization for the network service management functions through thin terminal so as to carry out the effective management for the places where there is internet. The component architecture provides plug and play framework that enables component-based service composition and interoperability, to improve the reusability, flexibility, scalability of integration, and basically reach "on demand, quickly build "to achieve effective management of network services while simplifying network service management system development. JBI Binding Component can convert protocol-specific message formats into normalized format, so that the system only needs to deal JBI normalized message ， make the JBI system separate from specific protocols to realize shielding the differences in NE , and avoid the traditional network service management operating directly with network element inconveniently.

7. Acknowledgements

This material is based upon work supported by the Natural Science Foundation of Shandong Province of China under Grant No.Y2007G38 and Specialized Research Fund for the Doctoral Program of Higher Education under Grant No.200804221031.

8. References

[1] AARON WEISS, Computing in the clouds, ACM netWork, 2007, 11(4) pp.16-25.

[2] G. Disterer, “ISO 20000 for IT,” Business&Information Systems Engineering, vol6, Jul. 2009, pp. 463-467, doi:10.1007/s12599-009-0076-x.

[3] ITU-T Recommendation E.86, Framework of a service level agreement,2002.

[4] P. Adirake, F. Kazuhiro, O. Koichiro,"Integration of component-based development-deployment support for J2EE middleware," The 4th International workshop on software Engineering and Middleware,Dec. 2005, pp. 230-244, doi:10.1007/11407386-17.

[5] Y.Guo Jiang, Liao Yuefaong, P. Behzad, "A survey of J2EE application Performance managemnet systems," Proc. IEEE Symp. IEEE International Conference on Web Services, (ICWS 04), IEEE Press,Jun. 2004,pp. 724-731.

[6] Yuanhui Sun, Zongshui Xiao, Dongmei Bao, Jie Zhao, "An Architecture Model of Management and Monitoring on Cloud Services Resources," Proc. IEEE Symp. Advanced Computer Theory and Engineering (ICACTE 2010), IEEE Press, Aug. 2010, pp. 3207-3211, doi:10.1109/ICACTE.2010.5579654.

[7] Huaping Zhang, Xiaosu Chen, Huiyu Liu, Jian Liu, "A BPEL and UDDI Based Dynamic Web Service Composition System Architecture," Computer System Applications, vol 2,Feb. 2007,pp.123-125.

[8] D.Lindquist, H.Madduri, C.J.Paul, B.Rajaraman, "IBM Service Management Architecture,"IBM System Journal,vol 46, Mar.2007, pp.423-440, doi:0018-8670/07/$5.00.

[9] Wei jie, Wentong Cai, J. Stephen, "Dynamic Load-balancing Using Prediction in a Parallel Object-oriented system," Proc. of the 15th IPDPS,Aug. 2002, pp. 345-353,doi:10.1109/IPDPS.2001.925024.

[10] Web services architecture: http://www.w3.org/TR/ws-arch/#webarch.