The application management system of railway-river combined transportation under cloud environment

Yunxi Wang¹, Qiang Huang¹

¹ School of Information Engineering, Sichuan Agricultural University, Yanan, Sichuan, 625014, China

*Corresponding author’s e-mail: 750897756@qq.com

Abstract. In the context of the serious information isomerized of China's hot metal water transport system, it is impossible to solve all problems of the same level in a complex cloud environment, resulting in rigid resource management problems and larger energy sources under the traditional "chimney" framework. Combining the technical framework of the iron and steel intermodal cloud platform, researching the key technologies of automated construction, deployment and cluster management of intermodal applications in the cloud environment, and building an intermodal application management system based on virtualization and elastic computing, not only can significantly reduce the heterogeneous heterogeneity of the cloud platform. The management cost of the intermodal service application can also provide a common virtual resource management service for the construction of intermodal information sharing and big data applications.

1. Introduction
The status of railway-river combined transport in China's logistics is becoming more and more important. As an intensive and efficient transportation organization, railway-river combined transportation can improve transportation efficiency and reduce logistics costs. Although China's railway and port transportation have made great progress, there are still major problems of resource management and energy consumption loss, and the technology is relatively backward. Therefore, we propose to adopt an intensive large-scale centralized management mode for the information management system of the railway-river combined transportation system, which is constructed by railways and ports, fully utilizes virtualization technology, and takes advantage of resource intensification to distribute the intermodal information systems distributed over each port. In the centralized management mode, based on the concept of resource sharing, the cloud computing power is used to virtualize the server hardware and establish a unified virtual resource pool[1]. Establish an application management pool on the virtual resource pool to form a shared intermodal application pool, break the technical barrier between ports, and make full use of the flexible scheduling and on-demand allocation characteristics of virtual resources to improve computing resource utilization and economy.

2. Integrated intermodal application management system based on DevOps

2.1. Component design of application management system
According to the construction management system, the application management component should be designed around ITAPP[2-4]. In the virtual resource pool of the resource support system, the DevOps mode is used to seamlessly connect the application delivery and operation process with DUI as the...
carrier. The automatic integration process of development, build, release, deployment to service. The collaborative relationship between its functional components is shown in Figure 1:

Figure 1. Integrated application management system based on DevOps

As can be seen from the above figure, the application management system is mainly composed of the following four components, which respectively correspond to the construction, deployment, management and access control process of an application:

1. Application Continuous Integration (ACI), unlike the current software package delivery model, under the application management system, ACI receives the software code submitted by the software vendor and automatically tests and packages the software according to the configuration. And publishing, forming an ITAPP application model based on virtual images.

2. Application Cluster Deployment (ACD) is responsible for managing a large number of intermodal application deployments. That is, according to the business requirements configuration, the ITAPP is automatically deployed to the ARE environment of the resource support system to form a DUI cluster.

3. High-Available Cluster Management (HACM), responsible for DUI cluster management, and application monitoring, recovery, and scaling mechanisms to achieve high-availability of application clusters in virtual resource environments. Quality of service.

4. Unified Access Control Gateway (UACG), based on the SaaS service model in the cloud environment, is responsible for unified access control of applications to eliminate the application security bucket effects in the business process.

2.2. Application continuous integration model

As the standardized content carrier of application for virtual environment, the mirror resource pool of resource supported system can effectively shield the structural differences between applications, so this paper is based on the virtual resource pool. In this paper, an application continuous integration construction model based on virtual mirror image is proposed. By encapsulating the application, the virtual image is stored in the mirror warehouse, and the application registry is established to manage the metadata uniformly. It can effectively avoid the defect of poor adaptability of heterogeneous environment in the process of package delivery, and combine the construction and release of applications into one. The model architecture is shown in Figure 2:
As can be seen from the figure, the ITAPP continuous integration build model is mainly composed of code warehouse, application building module, application registration center and application virtual mirror warehouse. Different from the current mainstream continuous integration system, the model first uses the mirror resource pool to shield the differences between applications and implements the general construction process. Then, through the logical division of the virtual running environment, the test and release environment is unified, whether it is Both the test image and the production image are tested in the same running environment, which effectively avoids the inability of the application to run when the application goes online, and finally eliminates the inefficient manual deployment process by applying the registry. Cluster deployment The module can be configured according to the metadata, automatically pulls the application image and deploys it to the production environment, which effectively improves the application efficiency. Building the model Automate the construction of the intermodal application in accordance with the following two processes:

The main contents are as follows: (1) construct the trigger process. This process is consistent with the current continuous integration model. Developers submit the code to the code warehouse of the application construction model after the application development and local unit test and integration test are completed locally, which triggers the application construction process. (2) Application construction process. According to the definition of ITAPP, the process outputs the metadata and ITDU mirror set of ITAPP, which is mainly divided into the following steps: (a) test mirror construction. The application construction module pulls the code from the code warehouse, and then applies the construction module to perform the ITDU test mirror construction and pushes it to the mirror resource pool; (b) to perform the test deployment. Pull the test image into the test area of ARE and automatically execute BVT (including functional test, stability test, safety test and stress test, etc.) to build ITDU to verify the; (c) production mirror construction. The application construction module sets up the code dependent environment according to the configuration of the production environment, constructs the ITDU production mirror and pushes it to the mirror resource pool, and sets the; (d) acceptance test of ITAPP metadata information related to ITDU. After the construction of the whole ITDU collection of ITAPP is completed, the application construction module configures the metadata information of the application to the application registry, and deploy the application in the production environment according to the minimum resource quota, and notify the user testers to carry out the acceptance test. If the acceptance problem occurs, it returns (1) to correct and reconstruct the application; if there is no problem, the application example is destroyed and the feedback of the construction results of the computing resource; (e) is released. The relevant information of the construction process and results is fed back to the operation and maintenance personnel, and the application construction is completed.

2.3. Application cluster deployment model
The deployment of intermodal applications mainly occurs in three scenarios: the first use of the application, fault recovery or resource scalability. The application deployment process under the
resource support system mainly includes the following three steps: (1) according to the service requirements or tenant request, the deployment request of ITAPP is obtained; (2) the ARE that conforms to the request is selected in the resource support system. (3) the ITDU unit of ITAPP is deployed on ARE to form DUI cluster.

Although some of the current business cloud platforms have been able to automatically increase resources and deploy (such as Amazonundefineds automatic extension services) according to service requirements [5], these three steps have not yet been effectively integrated. Users can only monitor and discover the application deployment requirements themselves, and then manually complete the application of virtual resources and deployment of applications and other work; At the same time, these cloud platforms can only be managed for a virtual environment, and can not meet the deployment requirements of heterogeneous applications. When the deployment scale is large or the frequency is high, it not only needs to consume a lot of time and manpower cost, but also seriously restricts the application deployment efficiency and flexibility of the resource support system. In order to solve the above problems, this paper holds that the deployment process of the application should be able to automatically select the matching virtual resources from the cloud nodes and complete the configuration and deployment according to the deployment request of the application. According to the idea of constructing the application management system, the deployment of the application is the process of loading the intermodal application image constructed by ACI into the ARE environment and running in the way of case cluster according to the configuration of ITAPP metadata, combined with the architecture design of the resource support system. It can be seen that the ARE target environment of ITDU is as follows: (1) for ITDPU, the ARE is a container environment managed by Mesos; (2) for ITSU, if its RTE property is Container, the ARE of that ITSU is Kubernetes; If RTE is VM, the ARE of the ITSU is OpenStack virtual machine. To sum up, the ITAPP automatic cluster deployment process based on resource support architecture is described as follows:

Algorithm 3-1: Automatic cluster deployment process of intermodal application based on resource support system

Algorithm name: itappACD
Input: itapp:Intermodal application to be deployed;
Output: collection<FAILED_ITDU>:The set of failed ITDU units is deployed. If it is empty, the deployment is successful;
    retrieve collection<ITDU> from itapp where ITDU.AID matches itapp.ID;
    while (collection<ITDU> not empty) {
        pick an itdu from collection<ITDU> satisfied D;
        monitor deployment procedure {
            create hosts and mount virtual volumes for this itdu from Ceph if necessary;
            if (itdu typed ITDPU) {
                cast itdu to itdpu;
                create Container-ARE if environment not ready; //Perform the ITDPU deployment process
                    register metadata of current itdpu to Chronos;
                    while (time satisfied ITDPU.SCH) {
                        obtain input-data path;
                        submit itdpu.C to Mesos-Master with itdpu.QT;
                        pull task image from image-repository;
                        //Configure a DUI cluster
                        dispatch tasks on Mesos-Slave;
                        treat each task as a DUI<i,j,k>; //Monitor the DUI's container running status
                        monitor each DUI<i,j,k>;
                        process and store ouput-data concurrently;
                }
            } else {
                create Kubernetes-ARE if environment not ready; //Perform the ITSU deployment process
                    retrieve container image from image-repository;
                    //Configure a DUI cluster
                    dispatch tasks on Mesos-Slave;
                    treat each task as a DUI<i,j,k>; //Monitor the DUI's container running status
                    monitor each DUI<i,j,k>;
                    process and store ouput-data concurrently;
            }
        }
    }

else if (itdu typed ITSU) {
    cast itdu to itsu;//Perform the ITSU deployment process based on RTE
    if (itsu.RTE typed CONTAINER) {
        create Container-ARE if environment not ready;
        //Create or update an ITSU deployment in POD
        execute itsu.C by Kube-CTL // register metadata of current itsu to
        Kubernetes;//Create a POD instance based on the REP copy requirement of itsu
        while (itsu.QT.REP not satisfied) {
            deploy a POD replication {
                add POD info to Etdc;
                scheduler select a minion for POD satisfied itsu.QT;
                create binding between POD and minion;
                add this bound POD object to Etdc;
            }
            kubectl get bound PODs periodically from Etdc;//Configure a DUI instance set
            pull POD image from image-repository;
            create and start docker containers by POD image;
            treat each container as a DUI<i,j,k>;//Create a service and set up a
            load balancing policy
            create kube-service and set LBC if necessary;//Monitor the DUI's container
            running status*
            monitor DUI<i,j,k>;//Register global service endpoint
            trigger registering service endpoint on Consul if necessary;
        }
    }
    else if (itsu.RTE is VM) { //Create an ITSU deployment in virtual
        machines
        create VM-ARE if environment not ready;
        execute itsu.C {//Create a virtual machine cluster
            while (itsu.QT.REP not satisfied) {
                deploy a VM replication {
                    pull VM template image from Glance for itsu;//Configure a DUI
                    instance
                    boot VM instance of image on VM-ARE with itsu.QT by NOVA;
                    set virtual network for instance;
                    if (itsu not deployed in instance) {
                        deploy itsu package on these instance;
                    }
                    treat this instance as a DUI<i,j,k>;
                }
                //Monitor the virtual machine running status of the DUI
                monitor DUI<i,j,k>;//Register global service endpoint
                trigger registering service endpoint on Consul if necessary;
            }
        }
    } else throw unknown RTE error;
}
else throw unknown ITDU type error;
//If an exception occurs during the deployment process, the itdu is added to
the failure set
add this itdu to collection<FAILED ITDU> if catch any exception;
} //At the end of the deployment process, monitor the cluster of application
instances for failover and scaling control
monitor this itapp for failure recovering and auto-scaling;
return collection<FAILED_ITDU>;

3. Conclusion
Based on the resource support system, this paper focuses on the key technologies of the Integrated
Application Management System (IAMS). According to the DevOps mode, the key technologies of
abstraction, continuous integration, automated deployment, high availability cluster management and
access control of intermodal applications are studied. Finally, the test environment of IAMS prototype
is built based on HCI mode, and finally in different operating environments. The application
performance was compared and tested to verify the performance advantages of the resource support
system.

References
[1] Zheng Li’an. Research on Business Process Design and Information Integration Alliance of
Railway Group Catering Supply Chain [D]. Jilin University.2013.
[2] Tao Zhenwei. Research and application of continuous integration based on Jenkins [D]. South
China University of Technology.2014.
[3] Xu Meng. Research and implementation of continuous Integration in Modern Enterprise
Software Development [D]. China Ocean University. 2013.
[4] Liu wen. Practical research on continuous integration in web componentization project
development [D]. Tianjin university, 2014.
[5] Auto Scaling [EB/OL], http://aws.amazon.com/autoscaling/.