Research on Automatic Deployment of Simulation Test Environment Based on Cloud Platform

Ying Cheng, Xiangdong Gao, Xiaochao Wu, Li Chen
Luoyang Electronic Equipment Test Center, Luoyang 471003, China
cychh1999@126.com

Abstract. It is a future trend that SoS confrontation tests develop towards combining simulation tests with realistic exercises, so how to deploy a simulation test environment on demand conveniently, quickly and smartly becomes an urgent problem to be solved. Starting from the architecture and requirements of a simulation test environment, this paper analyzes the deployment solution of Cloudify and Docker open source cloud platform clusters, designs a cloud-based automatic deployment platform for the simulation test environment based on OpenStack, Cloudify and Docker, and realizes the automatic deployment of the simulation test environment with full software stack and full deployment process. The automatic deployment not only improves the efficiency of the deployment and maintenance, but also increases the utilization ratio of resources.

1. Introduction
Simulation test has to face the difficulty of changing test environment frequently. To ensure the smooth implementation of the test and improve its efficiency, the test environment must be deployed and switched efficiently and quickly, and runs reliably and expands elastically. With the implementation of SoS confrontation test, the simulation test platform needs to be compatible with various business components and have versatile architectures, hence its deployment and management become more and more difficult. The IaaS technology of the cloud computing realizes the shared pool of the hardware resource of the business platform, and simplifies the deployment process of the business platform at the hardware level [1]. The PaaS technology of the cloud computing realizes the shared pool of the software platforms such as middleware database and so on, and simplifies the deployment process of the business platform at the software level. But the IaaS mode alone is based on virtual machine image template which doesn’t support the deployment of the basic and application software, and needs a lot of manual setups and configurations, thus has a lower deployment efficiency. Also, the PaaS mode alone cannot select the cluster architecture and dependent basic software, thus lacks deployment flexibility [2]. The emerging PaaS platforms such as Docker and Cloudify make it possible to solve the above-mentioned problems with a network business platform deployment solution featuring high flexibility, high efficiency, full stack and full process.

2. Introduction to Docker and Cloudify deployment platforms
Among all open source PaaS platforms, Docker and Cloudify are two popular platforms that don’t depend on environments and languages and can be used to deploy applications developed with any languages. The Docker focuses on image deployment at software level while the Cloudify places emphasis on all stack standard deployment, the organic combination of the two will be more suitable for the automatic deployment of the simulation test environment.
2.1. All stack deployment platform Cloudify

Cloudify is an all stack platform that fully supports hardware, network and software deployments, coming between applications and the selected IaaS cloud, and can solve the difficulty of the topological and cloud resource dependency of the network business platform cluster. The platform has not only integrated the IaaS-level cloud resource management platforms such as OpenStack and CloudStack and can realize the allocation and configuration of the hardware resource and NFV network, but also integrated software configuration platforms such as Puppet and Chef, and can realize the installation and configuration of the business platform software. Meantime, Cloudify is also a standard cloud orchestration and deployment platform which supports TOSCA (topology and orchestration specification for cloud application) and uses the standard cloud application orchestration model provided by TOSCA. Cloudify can be used to construct all life-cycle management model for all complicated application cluster under the cloud environment, to define the node capabilities, application stacks, and relationship of the application nodes using structural descriptive languages, to support the automatic deployment, scheduling and upgrade of cloud applications through management plans based on workflow, and to provide full process support for the application cluster deployment, including the application cluster orchestration, cluster deployment, cluster monitoring, flexible scheduling, application upgrade, and so on.

Cloudify is a typical PaaS platform orienting automatic application orchestration. From the perspective of automatic operation and maintenance, the automation consists of four levels, i.e. infrastructure automation, middleware automation, application automation and business automation, and Cloudify undertakes the application automation function and part of the business automation function [3]. Its main features are as follows:

1. Flexibility of framework design

Cloudify is programmed with Python language, integrated seamlessly with OpenStack, and can realize the allocation, setup and scheduling of the computing, storage and network resources. Meantime, the framework design of Cloudify is flexible enough to deal with any application stacks and to hide APIs and configurations in applications, which makes it easier to migrate applications and avoid being bound by the cloud (IaaS). It supports the migration of applications to any cloud systems at any time and in any environments, and the deployment of the same application in different environments.

2. Template approach to business orchestration

Cloudify uses TOSCA to make formalized description of the architecture, nodes, relationships and dependent hardware resources, software resources and configuration scripts of the business cluster, and to turn complex business cluster into structural, visual and reusable models which form application cluster template that makes it possible to migrate applications and technical system framework without changing any codes.

3. All process automation of business deployment

Cloudify uses Recipe to describe application, service and interdependent relationship, and to define how to monitor, self-repairing and expanding other services and resources. The workflow engine of Cloudify drives the deployment flow in the TOSCA models, calls the management APIs of each node in a serial or parallel manner, executes the installation, configuration and start of each node, and fulfills the automatic deployment of the entire cluster.

2.2. Container-class deployment platform Docker

Docker is a container-class deployment platform. It can pack an application and its environment dependency in a standard image file just like a container, use lightweight container virtualization technology to isolate resources and deploy applications to run in any environments supporting Docker. Multiple containers can share the same operating system to quickly deploy, launch, destroy and migrate applications and lower resource consumption of each application instance [4].

The emergence of the Docker container technology aims to solve the difficulty of multiple operating systems/application stacks, such as all virtual machine instances on a server to use the same
operating system, deployment optimization with the same application stack, storing a copy of the operating system in local hard disk to make the updating process more complex in a massive cluster application deployment, and so on. The container technology can load an operating system image and an application into memory concurrently. It can also load from network disk and concurrent launching dozens of images won’t bring too many loads to the network and storage. Later image creation processes only need to point to common images and thus reduce internal memory largely. Docker images contain the running environment and configuration information needed when containers run. The images come from repositories and use the file system with hierarchical storage to increase reusability.

Docker platform can address the software dependency and deployment efficiency. Firstly, it can simplify the software dependency of the nodes. Docker can pack in images one layer or multiple layers of software of the business node software stack, and record the hierarchical dependency of the images. Multiple images overlay each other layer by layer just like building blocks. When deploying the software, only the images on top layer need to be designated. Docker can install images from top to bottom one by one according to dependency of the images and complete the installation and deployment of the software stack of the node. Secondly, it can standardize the management and deployment interface of the node software. Docker can unify the package, storage, installation and running interface of all kinds of software by packing business modules and their dependency in standard images, which lower the difficulty of the automatic deployment of the node software. Thirdly, it can increase the deployment speed of the business nodes. Because multiple containers share one operating system on a server and Docker images only pack needed running libraries and configurations without binding the whole operating system, which usually size in MB, the package, transmitting and installation speeds of the images will be much quicker than the conventional overall virtual machine image mode. Meanwhile, after the software stack of the business nodes has been packed wholly and stored in the Docker image repositories, when deploying some pieces of software, all we need to do is to get and migrate them from the image repositories, simplifying the “installation-configuration-running” to “copying-running” and greatly speeding up the deployment of the business nodes.

Testing has proved that the container technology has almost the same running speed as the local platforms, with the performance of the CPU, internal memory, hard disk and network all close to that of the local operating system, thus lowering the total investment of the system. The container technology can make it possible to create two times of the virtual machine instances than before on one server, but it should be planned carefully because two times of the virtual machine instances means two times of I/O loads on the server running them.

3. Analysis of the simulation test environment deployment requirements and design of the simulation test platform

3.1. Analysis of the simulation test environment deployment requirements

As shown in Fig.1, the system architecture of the cloud-based simulation test platform consists of infrastructure layer, running platform layer, generic service layer, common support layer and simulation layer [5–6].

The deployment of the cloud-based simulation test environment does not need to focus on the hardware infrastructure, infrastructure software service, platform service, network transmission and modeling and simulation businesses, but on the automatic deployment on demand expected by the users. The cloud-based simulation test environment consists of multiple server clusters, with each cluster consisting of multiple server nodes and each server node consisting of hardware resources and multiple layers of software stacks of the software. The hardware of the business platforms deployed in the cloud environment depends on the computing, storing and network resources of the underlying cloud resource pool. The deployment of business platforms needs to address three dependency relationships: resource dependency, topological dependency and software dependency.
(1) Resource dependency

The deployment of the simulation test platform based on cloud platform needs to estimate the resource requirements of the business platforms, to apply the virtual machines, virtual storage and virtual network resources and to bind the nodes and clusters of the business platforms with the underlying cloud resources. The deployment of business platforms need not only to adapt to the interfaces of the cloud resources, but also to shield the difference between the interfaces of different cloud resource pools.

![Diagram of cloud-based simulation test platform]

(2) Topological dependency

The simulation test platform usually involves multiple simulation nodes or clusters and the communications between the simulation nodes are realized through service ports. The deployment of the simulation test platform needs to construct the topological structure of the platform and to address the relationship between the simulation nodes, the simulation nodes and clusters, and the clusters. The relationship between the simulation nodes is usually a communication relationship, with communication links between the source and target nodes via a service port. The service ports of the target node such as IP, port and URL etc need to be deployed to the source node when deploying the platform, and to ensure the three layers of links between the source and target nodes are connected. The relationship between nodes and clusters is usually a dependency relationship, including network dependency and node dependency. The network dependency means the nodes of the cluster and the cluster belong to the same sub-net, while the node dependency means the nodes of the cluster are controlled or managed by a certain main node of the cluster and the deployment of the cluster needs to configure the dependency, and meanwhile, conduct bulk operations (such as bulk configuration and bulk restart) of the nodes of the whole cluster through the dependency. The relationship between the clusters is communication relationship which requires to address the routing forwards between different sub-nets of the cluster.

(3) Software dependency
The software of the platform nodes usually consists of software stack with multiple layers and the installation of the upper layer software depends on the installation of the lower layer software. The whole software stack can be divided into four layers from top to bottom: running system software, generic support software, common support software and simulation software. The running system software consists of operating system, database system, virtualization platform, and so on. The generic support software refers to the generic service software related to the simulation technical framework, such as subscription and distribution, resource monitoring, timing service, data access, and so on. The common support software consists of scenario design, management of testing task, system monitoring, middleware, and so on. The simulation software is used to realize the simulation logic and provide the program or software grouping of the related functions of the simulation business, including the simulation of battlefield targets, battlefield environment, intelligence, sensors, weapons and equipment, and so on. The software automatic deployment mainly serves to configure the running environment for the software. Besides the correct configuration of the local environment, there is also a need to configure the network environment for the software according to the dependency of the components of the software, and address the mutual positioning between the software components to realize the network communications. The positioning modes of the components include the positioning based on fixed port, directory, middleware, configuration file, and so on.

For the software automatic deployment of the cloud-based distributed simulation test platform, what matters most is to deal with the dependency relationship of the resource, topology and software, that is to say, the deployment platform need to automatically distribute, install and configure all kinds of resources, clusters, nodes and software of the simulation test platform, load the developed simulation model software to the corresponding node, and run the whole simulation business with perfect accuracy.

The main requirements of the deployment of cloud-based simulation test platform are as follows:

a. Automatically processing the software dependency of the simulation business nodes
Be able to automatically install, configure and start all layers of software of the whole software stack of the business node according to the hierarchical structure of the software stack.

b. Automatically processing the topological dependency of the business platforms
Be able to conveniently construct the topological model of the business platforms, describe the affiliation, communication and dependency relationships between the cluster and nodes of the platform, recursively deploy the whole cluster according to the affiliation and dependency, and automatically deploy the related parameters of the nodes and network according to the communications relationship. The cluster topology with the same structure and deployment flow should be reusable.

c. Supporting any simulation technical framework and topology
Be able to support the automatic deployment of the cloud-based simulation business platform with any simulation technical framework and any topology.

d. With higher deployment density and efficiency
Be able to deploy more nodes on the same server through optimizing and compressing the software stack of the node and reuse the software stack and topology with the same structure, enhance the deployment efficiency of individual node, and increase the parallelism of the platform node deployment to raise the deployment efficiency of the whole cluster.

3.2. Design of the simulation test platform
The deployment platform of the simulation test environment is constructed with the open source cloud platforms OpenStack, Docker and Cloudify, and as shown in Fig.1, the whole deployment platform consists of web portal, PaaS manager, application cluster, image repository and IaaS. Among them, the web portal includes the human-machine interface designed to provide users with management service and model deployment; the PaaS manager provides the core functions of automatic deployment, including deployment model analysis, deployment process execution, resource monitoring and elastic scheduling, and completes the deployment of the whole cluster by scheduling
process engine and TOSCA container to implement the deployment process of the application cluster and by managing agents to call the management interfaces of the OpenStack and Docker engines; the application cluster node loads and runs the application stack by preloaded virtual machine monitor and Docker engine; all the images required by the application cluster, including the images of the operating system, basic software, simulation technical framework and simulation model software, are stored in the image repository; IaaS manager provides the application cluster with the services of computing, storage and the application, configuration, scheduling and destroying of the network resource through APIs.

(1) PaaS manager

The PaaS manager is the core module of the whole automatic deployment platform, and is constructed based on Cloudify to realize the deployment, monitoring and scheduling of the application cluster based on TOSCA models. The whole PaaS manager consists of various management plug-ins such as REST server, TOSCA container, workflow engine, strategy engine, message bus, management agent, OpenStack plug-in, Docker plug-in, and so on.

REST server encapsulates all the services of the PaaS manager, and provides the platform management with lightweight and standard call service interface. The TOSCA container takes charge of the analysis, storage and load of the TOSCA models and provides the workflow engine and upper-layer applications with static meta data and running state data of the application cluster. The workflow engine takes charge of implementing all kinds of management processes defined by the TOSCA models, including the cluster deployment, elastic scheduling and failure recovery etc. The strategy engine makes the operation decisions according to the monitoring indicators collected by the monitoring engine, and implements specific management operations through workflow engine or management agent. The management agent encapsulates the management, configuration, activation and scheduling interfaces of all kinds of software and hardware resources. The management capability of the platform can be expanded through management plug-ins, be aligned with OpenStack platform through OpenStack plug-in to realize the allocation and scheduling of the computing, storage and network resources, and be aligned with Docker engine through Docker plug-in to realize the management of the Docker container and image [7].

Application cluster

The application cluster is the final form to realize the functions of the simulation application. It can be deployed according to either simulation functions or simulation application relationship and the deployment can be optimized according to the monitored running states. The application consists of VMM virtual machine management system, operating system, Docker engine, node agent and Docker image. The node agent, together with the monitoring engine of the PaaS manager monitors the deployment and run of the application cluster and returns the information to the web portal. The Docker engine takes charge of receiving the scheduling of the PaaS manager and conducts the upload, download, start and destroying of the Docker image. Docker image extracts images from the Docker image repository according to the requirement of the application cluster.

(2) Image repository

The image repository consists of OVF image repository and Docker image repository. The OVF image repository takes charge of system-level image of the application node. Its contents include operating system, Docker engine and Cloudify node agent, whose operations being realized through Glance, the image management module of OpenStack. Docker image repository takes charge of storing Docker layered images such as basic software image and simulation technical framework image, which is realized through Docker registry module.

(3) Docker engine

The Docker engine resides in the application node and serves as its background daemon service process. It is the core module driving Docker container to load the running application stack. It receives requests sent by the Docker management plug-in and based on the type of the request, conducts the upload and download of the Docker images and the start and destroying of the Docker container through the scheduling mechanism of the routing and distribution.
(4) Collaboration between automatic deployment components

The Web portal deploys designs and plans configurations, and the information is used to conduct deployment model design include software components, network connections, interactions, and so on. The process engine executes the deployment process of the application cluster to realize the deployment of the database, simulation business application and load balancing clusters and the start of the server. The execution of the deployment process is configured according to topological model loaded by the TOSCA container and the network configuration of the virtual machines is configured when they are created. The load balancer takes charge of the loads between the servers of the balancing database clusters and the servers of the simulation application clusters. The application cluster is realized by loading images, starting containers and configuring routing tables.

4. Deployment process of simulation test environments

As shown in Fig. 2, the deployment of the simulation test environment can be divided into three stages: template design stage, template instantiation stage and template deployment execution stage.

![Diagram of deployment process]

Fig. 2. The automatic deployment process.

In the template design stage, the system architect or system engineer will complete the design of the cluster deployment template, that is to say, designing reusable TOSCA application cluster templates through TOSCA model designer, and distributing the templates on the platform management for the use of the system manager.

In the template instantiation stage, the work to be done includes deployment model configuration, deployment package packing and uploading, new image storage, deployment service call, TOSCA deployment model explanation, and so on. Among them, the deployment model configuration means that the system administrator selects a applicable application cluster deployment template through the platform management, adjust the nodes and relationships on the template to agreeing with the architecture of the application cluster to be deployed, and adjust the node bound system images and top-level Docker images to agreeing with cluster node to be deployed. Then, the TOSCA template files and new local images will be packed in cloud service deployment package and uploaded by using cluster package function of the platform management. After that, the platform management will parse the cloud service deployment package, store new images in the Docker image repository, send TOSCA model file to, and call the cluster deployment service of the PaaS manager. Finally, the
TOSCA cluster topological model will be interpreted as topological object model and stored in the TOSCA container, and the definition of the deployment management process will be loaded to the workflow engine.

In the template deployment execution stage, the workflow engine will implement virtual machine creation, network configuration, node image installation, container run, container configuration, and so on in turns by calling the management agent, and complete the deployment of the whole cluster automatically.

The deployment can be selected through the management platform and it can be the complete cluster deployment process, sub-cluster deployment process or the deployment of a single node.

5. Conclusion
Based on open source cloud service platforms such as OpenStack, Cloudify and Docker, this paper constructs a cloud-based automatic deployment platform for the simulation test environment, which makes good use of the respective advantages of Docker and Cloudify, namely Cloudify deployment model’s support for complex cluster architecture and Docker node’s support for any software stack, to realize the automatic deployment of the complex cluster with full stack and full process, which balances the relationship between the efficiency and flexibility of the deployment and supports the rapid change of the user’s requirements.

References
[1] L. Ji et al., OpenStack Open Source Cloud. Beijing, Tsinghua University Press, 2014(in Chinese).
[2] P. Xu, S. Chen, and S. Su, “Internet Application of PaaS Platform Architecture,” Journal of Beijing Post University, vol. 35, issue 1, pp. 120-124, 2012(in Chinese).
[3] Miglierina M, Application Deployment and Management in the Cloud. Proceedings of the 16th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, Timisoara, Romania, 2014.
[4] H.L. Sun, “Docker Source Analysis Part 1: Docker Architecture,” http://www.infoq.com/cn/articles/docker-source-code-analysis-part1, 2014.
[5] W.D. Cheng et al, Research on and Design of Evaluation Test Platform of Cloud Environment Capability. Proceedings of the 6th China Symposium on Command and Control, Beijing, China, 2018(in Chinese).
[6] S.H. Zhu, Research on Construction Technology of Massive Test Environment Based on Virtualization. Nanjing, Southeast University, 2015(in Chinese).
[7] H.R. Zhang, Design and Realization of Cloud Platform Automatic Deployment Module. Harbin, Harbin Institute of Technology, 2013(in Chinese).