November-December 2019
ISSN: 0193-4120 Page No. 2456 - 2459

A Study on the Kubernetes based Infrastructure Model for Smart Factoring

Cheolhee YOON[1], JangMook KANG[2] Myung hi CHAE[3]
Police Science Institute, Asan Republic of Korea E-mail: bertter@police.ac.kr
Global Cyber University, Seoul, Republic of Korea. E-mail: honukang@gw.global.ac.kr
KT, Seoul, Republic of Korea. E-mail: lan.chae@kt.kr

Article Info
Volume 81
Page Number: 2456 - 2459
Publication Issue:
November-December 2019

Article History
Article Received: *5 March 2019*
Revised: *18 May 2019*
Accepted: *24 September 2019*
Publication: *12 December 2019*

Abstract
An open source-based platform for deploying, extending, and managing automation of container-based applications became commonplace. SmartFactory also uses virtualization technology in its infrastructure. And as virtualization technology evolved, increasingly lighter and more efficient docker technology became popular. Efficient deployment and management of containers in a large-scale manufacturing cloud computing environment has become a key technology in SmartFactory. Currently, Google's internal container services are open-sourced, and kubernetes is evolving into a standard technology. In this paper, we describe the container technology that was introduced to analyze the manufacturing information collectively. In addition, virtualization, which was the emulation of software through the whole process of previous manufacturing processes, was not the structure but the application infrastructure centered on the operating environment for customized manufacturing. The purpose of the study was to apply the kubernetes technology to smart factoring to deploy and manage containers in a large-scale manufacturing process.

Keywords*: Smart Factoring, Big data analysis, Kubernetes, Docker, Container*

1. INTRODUCTION

Kubernetes first presented at the 2014 Google Developer Seminar, and first introduced how Google products work in containers to the public. Kubernetes is an open source flat that automates Linux container operations, providing deployment automation, automatic scaling, and recovery service. [1] That is, components that perform the tasks of hundreds or thousands of services are typically constructed with the strengths of high scalability and high fault tolerance through the micro-service architecture style. Although it is difficult to identify the relationship between the overall complex system structure and each component of the system, Kubernetes makes it easy to deploy micro-service systems and is the architecture required for the necessary step-by-step service transplants and extensions in smart-factoring. Systems at the manufacturing stage require a portability and expandable software platform to manage containerized workloads and services for automating application deployment, expansion and management, which has been the main driver of Google's internal container services since the Borg structure was opened and sourced, and this has become the main driver of the Kubernetes technology currently in use.

2. Kubernetes and Smart Factoring

A. Smart Factoring with Kubernetes

By applying Kubernetes to Smart Factoring, high availability, isolation, and unrestricted placement can be seen as strengths in cost reduction and efficient tasks. The general configuration required for smart factorying is as follows: Each node can be created by a docker-type image and placement, which can be managed using Kubernetes. Applications and data from each stage are distributed using a docker, which can be managed through Kubernetes. [2] The advantages of configuring the entire system and leveraging the detailed docker management of the Kubernetes are that the hardware on the host can be fully utilized through high availability and complete isolation of hosted applications. The advantage of running unrestricted applications is that it is possible to run efficient applications through thousands of nodes, just as every node is a single giant computer. The table below characterizes the advantages of applying Kubernetes. The following table describes the advantages of smart-factoring through Kubernetes[3]

Table 1. Function of Kubernetes

Function	Description
High availability	In a cluster configuration, even if some of the nodes fail, the nodes fail. Container moved to normal node to maintain service continuity
Isolation	Get the most out of your hardware while maintaining complete isolation of hosted applications
unrestricted Application	Just as every node is a single giant computer, thousands of computer nodes. Enables software applications to run
Management	Abstracts the underlying infrastructure to simplify development, deployment, and management of both development and operations teams
Management functions	Containers such as auto scaling, rolling distribution, compute resources, and volume management Provides many management capabilities for applications

B. Structure of Kubernetes in Smart Factoring

The structure of Kubernetes consists largely of a master node and a regular node. The master node serves as the control of the whole Kubernetes. Decisions are made, such as detecting and responding to various events occurring on the node. Because a master node can consist of multiple nodes, it can also be configured in a redundancy or triplet form, rather than a single master. In smart-factoring, configurations can be proposed as follows: The master node consists of API server, controller manager, scheduler, etc. Typically, the API server functions as the front-end role of the cluster, creating pod, controller manager, and scheduler, and performing a series of tasks by receiving all REST requests. Information on the operating status of Kubernetes is stored through the ectd and verified and managed in accordance with the service details of the distributed pod. In addition, the controller manager executes a number of unique controller processes in the background, and when a service configuration change occurs, the controller detects the change and starts working to handle it. In addition, Scheduler applies algorithms to distribute pods to various nodes based on information that investigated resource utilization. It refers to the requirements for service operation and performs the placement on the most suitable node[2]

In addition, Etcd is a repository of simple distributed key values that are used to store Kubernetes Cluster Day, API objects, and service discovery details and is designed to be accessible only from the API server for security reasons. In addition, it communicates with the master through an agent called kubelet and consists of modules such as kubelet, kube-proxy, etc. inside the normal node where the actual container, pod, is generated. Therefore, the generated or changed pods can be utilized mainly through API servers. Finally, Proxy is a proxy service that runs on each worker node to manage individual host subnets and to support external access to the service. The reason for this is that it is necessary to ensure that requests are delivered to the correct pod on a network that is isolated from each other, enabling efficient communication. [4]

Here, the application that supports smart factory, i.e. a group of one or several containers, is called a pod. A Pod is a single process running in a cluster that is a fundamental element for deployment in a Kubernetes. A Pod is the smallest and simplest unit in a model of Kubernetes objects that users create or distribute, and applications that operate on containers inside the Pod share all resources within the Pod. Therefore, the dockers are configured to be on the same network to communicate between containers. Because in Kubernetes, containers within a pod share resources. With the same IP and port space, communication is possible to the local host and storage is also shared. If a container is removed and restarted, the virtual volume assigned to the pod is retained as long as the pod is maintained, and the volume needed can be used to maintain the permanent data of the pod even if one of the containers needs to be restarted. This is because containers can be distributed as nodes where the task of smart-factoring is needed, and automatically recovered if operational errors occur, making it easier to add, replicate, update, and rollback as needed. Here, containers refer to the technology in which processes operate in isolated spaces among several virtualization approaches, enabling an incubating model through the efficient connection of master and walker nodes. The following table describes the essential details of smart-factory[4]

Table 2. Component of Kubernetes

Node	Component
Master Node	Controller Manager
	API Server
	Etcd
	Scheduler
Worker Node	Kubelet
	Kube-Proxy
	Container Runtime
	Pod

3. Smart Packaging Incubation with Kubernetes

A. *Incubation with Kubernetes*

The cloud-based smart-factoring incubation platform is a platform that continuously improves performance by learning and updating data sets of neural network learning in a cloud environment based on actual site-generated data. In order to receive intelligent application technology at a site that requires smart patching, the analysis engine generated through deep learning technology is downloaded and applied to the manufacturing site by connecting to a cloud server

consisting of on-premises. The goal of these cloud platforms is to provide continuous performance improvements using image datasets in real-world environments. Incubating platforms for smart-factoring can be offered in various forms through essential platform incubators at manufacturing sites. In other words, it can be configured as an abstracted virtual machine with artificial intelligence-type neural network learning, and the manufacturing site can be modified with smart-factoring depending on hardware specifications or the cloud environment it has deployed. This is because depending on how you configure it, you can create it with a virtualization instance, a docker contactor, a pod in a Kubernetes cluster, and so on. It can also be developed into a later management model because it can be configured using tools to monitor resources to manage the system by referring to the resource utilization status of Kubernetes. The monitoring structure utilized by Kubernetes requires monitoring of the nodes running the Kubernetes container, which requires monitoring of the CPU, memory, disk, network usage of its platform. Next, information about the container is monitored.
It is designed to monitor individual applications that run inside the container. In other words, it is possible to express as shown in the figure.1 Monitoring of Smart factory by kubernetes.

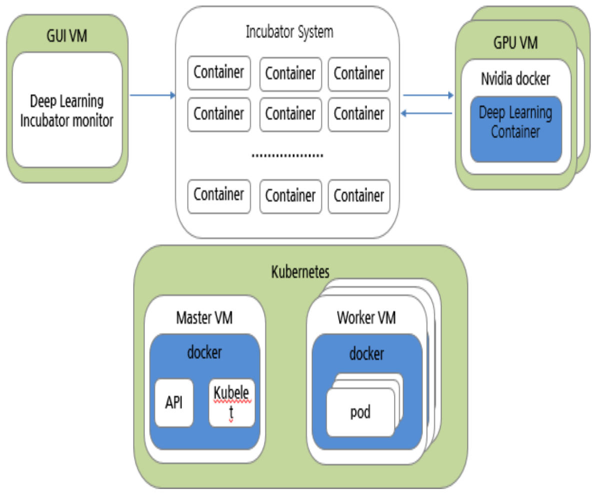

Figure 1 .Incubation with kubernetes

It is a concept that allocates one incubator to users who want to receive certain intelligent data analysis services using an incubation platform based on Kubernetes. Thus, manufacturing plants with thousands of services will allocate and manage one incubator for each node that requires intelligent services. The purpose of the incubating platform for smart-factoring is to provide cloud services that provide the most reliable intelligent imaging capabilities through in-depth online learning from data sets uploaded by users.
Once the manufacturing and data analysis techniques for smart-factoring are learned, and nerve-netting engines are created and verified, they are downloaded from the manufacturing site and applied to intelligent Kubernetes systems. The performance of the intelligent services applied to smart-factoring will be updated with continuous data upload and neural network learning. And the incubating platform will provide an ecosystem for smart factory for datasets and online learning by building an image database for deep online learning based on actual surveillance image datasets at the site.

B. ***Design as a decentralized structure***
The docker in the virtual environment for smart-factoring and the capabilities provided by Kubernetes that automate them can be designed in a distributed architecture, taking into account offloading capabilities offered in the cloud.

First, the ability to monitor the state of factory components and detect abnormalities shares the upper and lower control values determined by data analysis with the edge in the central cloud, and implements real-time monitoring, and abnormal detection. Second, Simulation based on plant change points sends detected or expected change points to the simulation engine in the central cloud as input values to operate the simulation and receive the forecast results. That is a possible On-site response. Third, for pre-predictions of plant element anomalies, long-term, data-driven learning is typically performed with big data storage and learning machines in the central cloud. As described in the requirements analysis, simulation of changes in operators, facilities, logistics, and public law are performed, and the models made in the central cloud perform inferences based on real-time data by the functional module of the edge that can perform the same predictions. However, the edge functions proposed in this paper include small, lightweight learning using the computing resources of the edge and the implementation of the underlying predictive model [5][6]
That is, based on the accuracy needs of the data that needs to be predicted, the learning on the edge node and the learning on the central cloud are configured to choose. For example, in case of examination image quality determination for appearance examination automation, it is effective to transfer image data to a cloud server by pre-treatment at the edge node, and use a judging model learned from bulk samples in the cloud, and determine the replacement frequency of parts based on the sensor data in the facility, the sensor time series data is learned only on the edge node. To perform all of this automatically, the following figure1 shown. In Smart Factory, we are developing an optimal deployment algorithm, researching a system that can be used by applying it to Kubernetes, and realizing it through incubation, which is expected to contribute to the development of Kubernetes system. We looked at Kubernetes, which was developed for container deployment and management when using containers, a lightweight virtualization technology. Kubernetes technology is growing as one of the core technologies of the 4th Industrial

Revolution, combined with the multi-access edge computing technology, which is being developed along with the development of the 5th generation mobile communication technology

4. Conclusion

The infrastructure was presented considering the target of edge computing through Kubernetics in smart factory.[7] We installed a lightweight virtualization engine (Docker) that uses the desktop resources to secure the appropriate capacity for specific production lines, cells, and tiers managed by Kubernetis. Through this incubation, the method of virtualizing hardware resources was also suggested. The virtualized hardware constitutes a cluster, and finally, a method of deploying and operating a docker container-based application image is also presented. The lightweight virtualization of the hardware and the smart factory configuration using Docker containers make it possible to directly implement Continuous Integration & Continuous Deployment under Kubernetes.[7] For this reason, Auto Scale In / Out and Rolling distribution was presented as incubation using Kubernetes' container orchestration. In the smart factoring model, one work area is allocated to each required work service. Each task needs an application to control the case service and the pod state of that service. Each service required for field maintenance needs is tracked and communication ports are required, and dependencies between system services on site can be managed through the entire Kubernetes. In other words, Kubernetes can simplify the analysis process-oriented process, sharing the same type of data among senior decision makers, intermediary reporters, and field workers and making it automated and simplified. In addition, the increased convenience makes it possible to quickly determine the decision-making structure through unified analysis results. Above all, due to the real-time data acquisition and analysis ability of the field staff, you can directly prepare and confirm the statistics and analysis results of the data you want from the field worker's point of view. By using Kubernetes, it is possible to apply statistical work, analysis work, and regular reporting work automatically in order. By automating this, it is possible to succeed by continuously utilizing the advanced system centered on work analysis ability. We expect Kubernetes to upgrade the Smart Factory environment [8].

ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP). (No.2018-0-00705, Algorithm Design and Software Modeling for Judge Fake News based on Artificial Intelligence)
This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2018S1A5A2A03038738, Algorithm Design & Software Architecture Modeling to Judge Fake News based on Artificial Intelligence)

REFERENCES

1. Ho Joon Won, Young Han Kim, "An Implementation of RCA Applied Micro-Service Based on Kubernetes in Cloud Environments," Proceedings of Symposium of the Korean Institute of communications and Information Sciences , 2018.11, 344-345(2page
2. Jaeyeop Jeong, Sanggil Yeoum, Hyunseung Choo, "Study on Container management tools Docker Swarm and Kubernetes " The Korean Institute of Information Scientists and Engineers. 2018.12, 1206-1208(3 pages)
3. Tang Bo, Eun Man Choi, "A Research on Visualization of Kubernetes Based Micro-Service System" The Korean Institute of Information Scientists and Engineers, 2018.6, 508-510(3 pages)
4. Kim kyung- il et al., "Kubernetes Architecture for Cloud Services", The Journal of The Korean Institute of Communication Sciences 35(11), 2018.10, 11-19(9 pages)
5. http://kubernetes.io
6. Jong-Kyung Park, Tai-Woo Chang, "Review of Domestic Research on Smart Manufacturing Technologies", The Journal of Society for e-Business Studies, Vol.23, No.2 pp.123-133, May 2018. DOI: https://doi.org/10.7838/ jsebs.2018.23.2.123
7. Hyung-Sun Kim, Hong-Chul Lee, "Development of Edge Cloud Platform for IoT based Smart Factory Implementation", Journal of the Korea Society of Computer and Information 24(5), 2019.5, 49-58(10 pages)
8. Jabarullah, N.H., Shabbir, M.S., Abbas, M., Siddiqi, A.F. & Berti, S. (2019) Using random inquiry optimization method for provision of heat and cooling demand in hub systems for smart buildings, Sustainable Cities and Society, 47, 101475.