Research on Integrated Deployment Method of Digital Earth Thematic Application Model

Zhao Danpu¹,², Tai Xianqing¹,² and Ma Zhijie²

1. Key Laboratory of Technology in Geospatial Information Processing and Application System, Institute of Elec-tronics, Chinese Academy of Sciences, Beijing 100190, China
2. Research Center of Intelligent City and Intelligent Transportation, CAS R&D Center for Internet of Things, Wuxi 214135, China

dpzhao@mail.ie.ac.cn

Abstract. Nowadays, the digital earth not only relates to the technologies of surveying and mapping geography, but also includes the analysis and cross-application of various scientific data related to geographic information. It requires a computing environment and management for the corresponding thematic application models which supports the online analysis on the digital earth. So we proposed an integrated deployment method for digital earth thematic application models based on container cloud, which provides a flexible and scalable information integration and analysis architecture for digital earth. Firstly, a Docker image building module is designed to facilitate the user to quickly build the application model into a image. Then the Docker technology is used to implement the containerization of the application model to provide a stable running environment for the application model. Finally, the Kubernetes is used to dynamically manage the cluster resources to realize rapid expansion of computing resources. In addition, the HDFS is deployed, and the image files and various scientific data respectively store with Hbase and Apache file systems. It can ensure the read and write speed of the application model and the related data and log files. It also used zabbix framework to manage the monitoring index of the container cloud cluster to ensure the execution effect of the application model. Through the tests of typical application model, it is verified that the method can integrated deploy and apply the cross-domain, hetero-architecture, multi-version and multi-class application models on the digital earth.

1. Introduction
The "Big Earth Data Science Engineering Project" takes the earth big data as the starting point to carry out data integration, sharing, mining analysis, visual application, information decision support in resources, environment, ecology, biology and other fields, and provide scientists with a digital earth platform (CAS earth) for data sharing and cross research[1]. Different from the traditional digital earth, the platform can not only provide a multi-modal visualization method for all kinds of spatiotemporal data, but also support cross field data online analysis, image processing, situation deduction and other operations, which is convenient for researchers to carry out their work with low cost and high efficiency. But at the same time, it puts forward new requirements for digital earth platform, which needs to provide storage and interface for all kinds of data involved in calculation, provide calculation environment for all kinds of thematic calculation and analysis models or microsystems, provide task
scheduling and resource coordination mechanism for thematic model or microsystems involved in calculation, etc. Therefore, based on container cloud technology, this paper studies the technology of multi-model system instant application service architecture, rapid deployment and service, version management and service call, and proposes an integrated platform of digital earth thematic application model which is designed into four parts: cluster side, storage side, monitoring side and service side to meet the application requirements of cross domain, different architecture, multi version and multi-user.

2. Various Thematic Models of Digital Earth

Big Earth Data has many characteristics such as multi-domain intersection, various types, large differences in attribute, and significant spatial-temporal scale differences, etc. Compared with the typical data mining analysis in a specific field, the cross-domain mining analysis of big data is closely related to the mining methods used and the types of knowledge discovered, and the selection of mining analysis model and algorithm will directly affect the quality of the discovered knowledge. At the same time, in the multi-disciplinary and cross domain scenarios such as resources, environment, biology and ecology and so on, it is difficult to effectively promote scientific discovery by using the existing mining analysis theories and methods in a single specific field. It is necessary to comprehensively select and use the corresponding data mining models, algorithms and tools according to the specific needs, and make full use of self-learning technologies such as machine learning and artificial intelligence to improve the degree of automation and reduce the participation of human-computer interaction. To this end, not only need to provide a unified data access, storage, access and computing environment in the data layer, but also need to break the domain barriers in the mining analysis (model, algorithm) layer, provide cross-domain model and algorithm sharing environment, develop new big earth data mining analysis method, form a research and innovation work environment based on cloud model, and support the innovation design of big data analysis model and the exploration and discovery of high-value interdisciplinary knowledge association.[2]

In this paper, the digital earth thematic application model is divided into two categories: one is a general-purpose algorithm, which refers to the code corresponding to the logical process of data analysis and processing, including various scientific calculation functions such as basic numerical operations, array or matrix operations, and commonly used Reduction devices and so on, and general machine learning algorithms such as regression, prediction, classification, clustering, timing analysis and other traditional machine learning algorithms and deep learning algorithms such as convolutional neural networks; the other category is dedicated services, referring to the dedicated module or system developed, which is researched and developed by scientists in specific fields for prediction, early warning, evolution analysis, trend evaluation, and so on, and can independently run external output services, such as remote sensing image processing services, tile services, elevation data services, 3D model data services, basic vector services, etc.

3. Container Cloud Technology

Cloud computing has developed for more than ten years since it was first proposed in 2006[3]. As a new model for computing resource provision and usage, it can virtualize hardware resources such as computing, storage, and network into a multi-user shared resource pool and provided to users on-demand and scalable through the Internet[4].

3.1. Container Technology

Container technology originates from virtual machine technology. At first, hardware resources are virtualized based on virtual machine management program, and a complete hardware system is simulated by software to realize the quota and isolation of computing resources[5]. Later, with the continuous enrichment of application programming interfaces, large-scale virtual machine containers can be managed in an automated way in the cloud computing platform, thus forming an operating system level virtualization technology, namely container technology.
Docker is a representative open-source container management tool in 2013\textsuperscript{[6]}. It uses Cgroups technology to achieve resource quota, namespace technology to achieve resource isolation, and built in image construction and distribution functions based on the joint file system. By sharing the operating system kernel with the host, the resource cost is saved, which makes docker lighter than virtual machine technology.

3.2. Container Cloud Technology Based on Kubernetes

Kubernetes is an open source container choreography engine of Google\textsuperscript{[7]}. It is built on top of Docker technology to manage containerized applications in the cloud platform, providing an application deployment, planning, update and maintenance mechanism \textsuperscript{[8]}. Kubernetes uses Pod as the basic management unit and contains one or more closely related containers in each Pod. In Kubernetes, you can create multiple containers, and run an application instance in each container. Then it can management, discovery, and access to the application instance through the built-in load balancing strategy, eliminating the need for complex manual configuration and processing of O&M personnel.

By building application programs such as thematic models and microsystems into container images, and then building container cloud based on Kubernetes, the whole set of functions such as resource scheduling, deployment and operation, service discovery, capacity expansion and reduction are provided for containerized applications, so as to realize the cloud management of digital earth thematic application models.

4. Design of Integrated Deployment Platform of Digital Earth Application Model

In this paper, an integrated deployment platform of digital earth application model based on container cloud is designed. Kubernetes is used as the deployment, management and scheduling platform of container cloud cluster. The efficient management ability of Kubernetes to container cluster is used to solve the problems of complex cluster creation and insufficient management, storage and monitoring ability of application model in large-scale distributed computing. The design of integrated deployment platform of digital earth application model based on container cloud is divided into four parts: cluster side, service side, application side and monitoring side. The overall architecture is shown in Figure 1.
The integrated deployment platform of digital earth application model is oriented to multiple hosts running application containers. Through the open-source cluster management framework Kubernetes, and based on the B/S architecture, the application container management system is built to manage and service the thematic application model.

The platform as a whole is based on the encapsulation and visual management of Kubernetes and Docker clusters. Docker provides containers for Pods to run, and Kubernetes is used for resource allocation and Pod management.

The cluster side encapsulates the functions of Kubernetes and provides the management of Kubernetes and Docker, such as image repository, container arrangement, deployment management, cluster management, run-time management, global configuration and other back-end functions, which is suitable for quickly inspection, configuration and the use the whole system.

The server side is more user-oriented. Through a set of process steps, the user can quickly start the application model layout, release, management, etc. The server-side management is to directly call the container-side interface. The DNS management of the system is directly implemented by the domain name service of Kubernetes\[9\], and users can set the domain name and access model directly without worrying about the specific IP and port of the model running machine. The specific implementation is to use kube-dns for domain name resolution, and ingress monitor the ingress-controller to update the configuration file dynamically and reloads the configuration file without restarting the model.

The monitoring side monitors the host, cluster, service, container and event status in an all-round way, and provides early warning and fault location to ensure stable and reliable operation of the platform.

There are two types of users for the integrated deployment platform of digital earth application model, one is to use the model through the server side, the other is to create and store the container image of the thematic application model through the platform, so an application side is needed to support such users for corresponding operations.

4.1. Cluster Side

The cluster side is the bearing platform of application model, which provides strong model integration ability, and can integrate various models in different languages and architectures; it provides refined management of the whole life cycle of application model, and supports the safe and reliable operation of various models. The cluster architecture is shown in Figure 2. Kubernetes cluster and UNIX system/localized system are deployed on the unified infrastructure respectively. Because not all thematic application models can be containerized, it is necessary to provide the system environment of deployment and operation for these models separately, and deploy host cluster and virtual machine cluster on top of it.
According to the thematic application model that can be containerized, this paper designs the container management subsystem. Based on the Kubernetes cluster, it carries on the refined management of the whole life cycle of the application model written in different languages, including six parts: repository management, container orchestration, deployment management, cluster management, runtime management and global configuration.

There are multiple nodes in the container cloud cluster, each of which runs the application model containers, and the running of these containers needs to be instantiated into containers by images, so each node stores multiple images. The repository management module manages these images, including the basic image management function and the repository image management. The basic image management provides the version list, image search, image addition, and image deletion function of the base images, and the repository image management provides the list display, image search, and image version view of the repository images.

Container orchestration refers to the process of organizing the work of a single application model, including resource orchestration, workload orchestration, and access orchestration. Resource orchestration is responsible for the allocation of resources (cpu, mem); workload orchestration is responsible for sharing workloads between resources; access orchestration is responsible for service discovery and high availability.

Deploying an application container in a cluster requires consideration of the resources of the entire cluster, and the module can manage multiple clusters. The container deployment process is to deploy the application images in a cluster according to a certain configuration (including cpu, mem, and number of copies). The deployment management module provides container deployment support and management, and supports deployment detail, deployment upgrade, instance information, and network/access information view.

The integrated deployment platform of application model is oriented to the management, operation and maintenance of multiple clusters, supporting different application model business directions. So a cluster management module is needed to manage different application clusters. Cluster management provides the creation function of container cluster and virtual machine cluster, as well as the function of viewing and managing container and virtual machine cluster. It also includes the functions of viewing cluster host, cluster setting, namespace setting, cluster member setting and deleting cluster.

The runtime management module provides service operation monitoring management, including service running status monitoring, operation event recording, and service log viewing and provides exception recovery, version rollback, and other functions for exception services.
In addition to providing container cluster related functions, container management needs to parameterize the platform to better support the normal and stable use of the container cluster. The global configuration module provides user management, private repository configuration, server configuration, monitoring configuration, web SSH configuration, and service center configuration. Among the six configuration functions, user management creates and edits the system login user; the private repository configuration configures the private repository address; the web SSH configuration is for the web access address of the container management subsystem; the monitoring configuration configures the reporting agent component access address, the storage component access address and query component access address of monitoring resource information of all nodes in the cluster, and provides the function of accessing the specified application container. Service center configuration provides the back-end access address of the configuration service center.

4.2. Server Side

The server side is a web page that interacts directly with the user, providing a simplified way of application model deployment and invocation, unified management services, data units, and DNS, etc. Through service orchestration, application models can be associated with dependent data units and service resources to complete application model deployment. The server architecture is shown in Figure 3. It includes service orchestration, service publishing, data unit management, cloud service management, and external service management.

![Figure 3. Server Side Architecture Diagram](image)

The service orchestration module introduces service templates and data dependency templates based on the application deployment, and encapsulates some of their interfaces, so that users can deploy in a functional module. The service orchestration module is the core of the entire platform function module, including the functions of creating, deploying, running, upgrading, expanding/reducing, uninstalling, deleting, stopping, connection testing, and dependency testing of the application model container.

Service publishing realizes the function of service access address (IP: port or domain name) external exposure, and unifies the integration and management of heterogeneous service address through the address book mechanism of service directory subsystem. External clients initiate service addressing through the relevant interface of service directory subsystem.

Data unit management describes the data resource configuration required by the container running in the cluster, including the dependent storage resource service type, connection path, link resource name and so on, and provides upload, download, delete, and view functions for data resource configuration.

A cloud service is a service in which a service instance runs within a platform. The cloud service management module is responsible for managing its lifecycle, including running, stopping, upgrading,
expanding/reducing, connection testing, dependency testing, and more. In addition, the cloud service management module provides functions such as service list view, keyword search, sorting, and service detail check (including instance status, multi-port case, operation information).

Due to reasons such as language or model architecture, not all thematic application models can be containerized on the digital earth platform. Therefore, these thematic application models are regarded as external services, and unified hosting mechanism is provided through external service management module. The functions include external service addition, domain name creation, service list display, keyword retrieval, etc. In addition, external services can participate in container service choreography.

4.3. Monitoring Side

When the nodes and container services running in the cluster, it cannot guarantee that there will be no problems after running. In order not to affect the user experience, it is necessary to view the health status of the whole cluster in real time. When there is a problem with the nodes or container services in the cluster, it can quickly locate and find the fault and solve it, or prevent the occurrence of the fault to reduce the loss, so monitoring is very important. In the monitoring management module, the monitoring and viewing function of the running status of a host or the service instance of a cluster in a production or test environment is provided. The Monitoring architecture is shown in Figure 4.

![Monitoring Side Architecture Diagram](image)

The data collection module not only needs to collect the CPU, Memory, Disk, and network resource status of the container cloud, but also collects the status of the host, the virtual machine cluster, the service status, and the system log information. At present, Kubernetes mainly monitors resources for CPU and memory, with limited monitoring indicators. Therefore, this paper uses the open-source zabbix-agent to collect the information about the memory and CPU of the hardware and operating system, and uses C language to write a dedicated collector for the indicators of the statistical classes in the cluster middleware and operating system, and uses shell and python language to design the indicator collection script.

In the aspect of state data organization and storage, firstly, the time sequential database OpenTSDB is used to store state information. At the same time, the relational database MySQL is used to store the basic data, task data, configuration data and analysis result data. At the same time, in order to ensure the efficiency and reliability of the system, this paper uses the memory database Redis as the cache database.

In order to ensure the accuracy of system status monitoring, early warning and alarm, this paper also designs a data mining, data association and analysis module. If the rules are triggered during runtime, the platform will send information to the administrator's mailbox.

In order to improve the user experience and achieve the visual operation effect, this paper extracts monitoring data sources and tasks from the database, analyzes the data, and integrates them into the Grafana plug-in. Grafana is a front-end page of display forms, which supports user-defined data
sources. Users can set monitoring indicators according to their needs, which is convenient and intuitive for resources monitoring.

4.4. Application Side

The application side is oriented to the users who designed the Thematic application model. Through the human-computer interaction interface, the containerized model is configured and mirrored according to the corresponding service capability template, and added to the image repository to provide the basis for the application model containerization. The Application Side architecture is shown in Figure 5. It includes five parts: repository service, offline packaging, service capability model, configuration center and service catalog.

Offline packaging provides software package, enabling the application to run and manage on the model integration deployment platform. The package is the final delivery form of the application, and contains the application images, metadata description, and other configuration (such as data unit, cpu/mem resource) description information. In addition, offline packaging is responsible for detecting the availability of the application images. By running the images, the application API is tested at the interface level, and the tested software package is pushed to the repository service subsystem for unified management.

The repository service backend integrates image repository and software package repository, and is responsible for package-related resource management and remote image backup. A package is the smallest unit of service center orchestration deployment. The repository service receives the software package pushed by the offline packaging, pushing the application image to the image repository, and resources related the package are persisted to the software repository.

The service capability model defines model specifications, including namespaces, service interface constraints, data storage constraints, and so on, so as to standardize the process of business system development and deployment, data access and so on.

The configuration center centrally manages the configuration of applications in different environments and different clusters. It supports the server side to pull the service configuration information automatically for software deployment, and supports the configuration pull and configuration change notification in the client mode. It supports configuration hot update and has the functions of configuration authority management, audit release, operation audit, configuration service monitoring alarm, operation log recording, etc.

The service catalog is mainly responsible for service registration and discovery, and unified integration and management of service addresses through the address book mechanism. It supports service provider to initiate service registration, service update, service offline and so on to the service directory when starting, changing, or terminating the service, and supports service caller to obtain the service list information by querying the service directory, and initiate service call request to service provider through service addressing.

5. Experimental Verification and Results

In the previous chapter, an integrated deployment method of digital earth thematic application model based on container cloud is presented. This chapter tests and compares the feasibility of the the
models written by different languages deployed on the container cluster and the effectiveness of different types of models deployed on the digital earth platform.

5.1. Experimental Environment

The server used in the experiment was 9 Inspur servers (CPU48 thread 128G, memory 256G, hard disk 30T), one of which was used as DNS server and time server, three of which was deployed MFS/Zookeeper/ Kafka/Hbase components, three of which was deployed Docker clusters, one deployed LLTS, grafana, ims components, one of which was deployed a digital earth base platform. The servers use Gigabit Ethernet to connect to each other and are in the same network segment and can communicate with each other. The cluster runs in the internal network. The operating system uses CentOS 7.2, the database management tool uses Navicat Premium, and the remote tool uses Xshell, xftp. On this basis, start deploying components and system modules in the following order: time service, ETCD, DNS, DFS, Zookeeper, Kafka, Hbase, LLTS, MySql, PostgreSql, Docker, Kubernetes, container management module, server side, application side, ELK, monitoring side.

At the same time, digital earth related services such as Web Map Server (WMS), Web Feature Server (WFS), and Web Coverage Server (WCS) are deployed on the container cluster, so as to view the effect on the digital earth based platform after being arranged together with the application model container.

5.2. Support for Language Verification Testing

Firstly, the map slice test model is written in three different languages: Java, python and js. In this paper, we selected some land images of China for slice test. The image is built by the application side and placed in the image repository respectively. Then through the Server Side, test model, WMS and WCS are carried out service publishing, data management, service choreography. Finally, the loading display of the corresponding images is viewed through the digital earth base platform as Figure 6, which verifies that the system can be loaded and deployed normally for services written in different languages.

![Image](image6.jpg)

Figure 6. The display effect of some land tile data in China

5.3. Support for Model Category Verification Testing

Through the application side, the place name association algorithm written by Java is built into the repository. Then, in the service side, the association algorithm of place names, WMS and WFS are used for service release, data management and service arrangement settings, and the basic platform of digital earth is refreshed to see the connection effect of direct traffic association with the target point as Figure 7.
Figure 7. the connection effect of direct traffic association

At the same time, a three-dimensional model of an aircraft and its parameter configuration information are written into a three-dimensional data service and an image is constructed. Through the server side, 3D data service, WMS, WCS and model trajectory data are published and orchestrated together. Finally, the loading and configuration information box of the aircraft model is viewed through the digital earth platform, as shown in Figure 8.

Figure 8. the loading of the aircraft model

In addition, in the platform designed in this paper, we have realized the deployment and release of algorithm model (place name association, target association, ocean current evolution, etc.) and service model (elevation data service, street view data service, 3D model data service, basic vector data service, etc.) respectively. After docking with the digital earth platform, the visual effect is shown in the figures below.
In this test, terrain service model and aircraft control model are deployed at the same time. Figure 9 shows the system supports flight simulation control and the terrain of corresponding coordinate areas.

In this test, digital earth service and roller shutter analysis model are deployed, and current data are imported. Figure 10 shows that the system supports online selection of target area and loading of profile model, online rolling analysis of scientific data on digital earth, and multi profile comparative analysis.

These tests verifies that the platform designed in this paper can support two application models of algorithm class and service class to load services on the digital earth.

Figure 9. the effect of 3D model and terrain service model loading at the same time

Figure 10. the loading of digital earth service and roller shutter analysis model
6. Conclusion

In this paper, an integrated deployment platform of digital earth application model based on container cloud is designed and implemented. Through the application of container cloud technology, the platform can support models written in multiple languages, rapid expansion/reduction computing space, and automatic restart and deployment of models. It is used to deal with the problems of large number, high complexity, poor stability and so on, which are faced by different subject fields and different types of thematic application models in the digital earth platform. And through real-time monitoring, it is convenient for cluster management and maintenance. The platform also integrates a set of development environment required for application model containerization, which facilitates the optimization and debugging of the model by researchers. The experimental results prove that the container cloud technology can effectively guarantee the high automation and high availability of the integrated deployment platform of digital earth application model, and has a good application prospect.

References

[1] Guo H D, Wang L Z, Chen F, et al. Scientific big data and digital Earth (in Chinese). Chin Sci Bull (Chin Ver), 2014, 59: 1047–1054, doi: 10.1360/972013-1054.
[2] Guo H D. Big Earth Data Science Engineering Project [J]. SubjectandField, Vol. 33 Issue (8).
[3] Cloud computing. https://en.wikipedia.org/wiki/Cloud_computing.
[4] Huisheng Xu. The design and implementation of container cloud platform [D]. Beijing: Beijing University of Posts and Telecommunications, 2018.
[5] Lnesh Gunaratne. The Evolution of Linux Containers and Their Future. https://dzone.com/articles/evolution-of-linux-containers-future.
[6] Simei He. Containerized cloud platform-orientated design and implementation of resource scheduler [D]. Hangzhou: Zhejiang University, 2017.
[7] GCE. [EB/OL]. https://cloud.google.com/container-engine/.
[8] Chen Jianjuan, Liu Xingxing. Distributed ELK log analysis system based on Kubernetes[J]. Electronic Technology and Software Engineering, 2016(15):211-212.
[9] Introduction to K8s. Kubernetes Chinese community [reference date 2018-09-09]

Acknowledgments

This work was supported by the Strategic Priority Research Program of the Chinese Academy of Science under Grant XDA19080202.