Elastic Application Container

Sijin He, Li Guo, Yike Guo
Imperial College London
South Kensington Campus
London, United Kingdom
{sh107, liguo, yg}@doc.ic.ac.uk

Abstract—The computing resource level architecture allows end-users to directly control its underlying computer resources, such as VM (virtual machine) operations, scaling, networking, etc. However, setting up and maintaining a working environment is complex and time consuming for end-users and resource management is also a heavy-weight task for the providers. In contrast, the application resource level architecture automatically controls its underlying computer resources so that end-users can concentrate on their core business. In this paper, we propose a new architecture called Elastic Application Container (EAC) that enables the end-users to efficiently develop and deliver light-weight, elastic, multi-tenant, and portable applications. The EAC is an abstract representation which hides all its abstractions of the underlying VMs. We believe that our EAC architecture has the potential to become the foundation of future application resource level model in this research area.

Keywords-component; Cloud Computing; Elasticity; Multi-tenancy; PaaS; Portability; Resource Management; Scalability;

I. INTRODUCTION

Service providers at computing resource level normally provide computing resources directly for end-users in the form of VMs. The end-users have to directly control their VMs, such as scaling, networking, etc. However, setting up and maintaining a working environment is complex and time consuming, and resource management is also a heavy weight task. Furthermore, the resource management of the providers in the computing resource level is still a heavy-weight task. This is because migrating an application in a VM from one resource server to another is equivalent to the VM migration, i.e. all data in the VM including the application and other middleware are transferred to another resource server. This could cost a huge amount of unnecessary I/O and network traffics.

Service providers at the application resource level normally provide platforms which allow end-users to directly manage their applications without considering the underlying computer resource consumption. The resource management of the providers becomes easier and more flexible as it is no longer based on VMs. It is based on each individual application. Migrating an application from one resource server to another only transfers the application. This provides a light-weight solution in comparison with the computing resource level architecture.

II. ELASTIC APPLICATION CONTAINER

We propose a novel architecture at the application resource level called Elastic Application Container (EAC). Similar to the concept of a VM, an EAC is a much less weight abstract representation which has its own configurations, i.e. CPUs, memory, and I/O etc. It offers lightweight resource provision mechanisms for end-users.

Figure 1 shows an overview of the EAC system. At the top of the stack, there are three EACs: two of them are hosting applications while one of them is vacant. One EAC can only host one application at a time and it provides application processing service for application execution, and data storage service for retrieval and storage respectively. The EACs are controlled by Elastic Application Container Space (a.k.a. container space) which directly communicates and controls its underlying VMs in order to provide real computing resource that EACs request. At the same time, it also manages the lifecycle of EACs including EAC deployment, scaling up or down, EAC removal etc.

A. The Architecture of the EAC System

Figure 2 shows all main components which constitute the EAC system. An EAC provides application processing power which is generated by a collection of application servers on VMs. We call them as Auto-scale Application Servers (AAS). This is because the application servers can be automatically scaled up or down according to demand of the overall system performance. The AAS are responsible for dealing with computing processing tasks of applications hosting on EACs. Each application server in the AAS is a software framework that provides an environment for running applications. End-users can upload applications to the AAS for deployment via the web-based user interface. An EAC also provides data storage service for application data storage and retrieval using a collection of databases. We call them as Auto-scale Databases (AD) which provides APIs for end-users to connect their applications to the elastic databases for data retrieval and storage.
The monitor serves for two different purposes, metering and monitoring. The monitor meters every application usages in an EAC, such as CPU cycles, memory used, data transfer, etc., for pricing and billing purposes. The monitor also keeps track of the performances of AAS, AD and their underlying VMs in the EAC system. Once the monitor detects some abnormality in the performance of a component, it will notify the resource manager to make appropriate resource management actions where the resource manager directly controls all resource management in the EAC system. It manages application placement, dynamic load balancers for an application in an EAC, and auto-scaling for the overall EAC system.

B. Features of the EAC system

1) Portability
Applications running on EACs are portable. End-users no need to fear being locked into the technology. It allows the end-users to reuse existing code instead of creating new code when moving an application from an environment to another. In order to make EAC portable, we use some popular existing computing paradigms, such as popular programming languages, traditional relational databases, instead of inventing a new computing paradigm. Further, the EAC system does not provide programming interfaces and abstractions that end-users need to know. In this way, the end-users can use their favourite programming tools to develop their applications and upload them via a web-based portal which give an easier and more flexible way for application development.

2) Multi-tenancy
Multi-tenancy of the EAC system allows end-users using EACs operate in virtual isolation from one another and manage an application as though they each have a separate VM, yet their data remain secure and insulated from the activity of all other end-users. An end-user allows having multiple applications depending on the number of EACs he has. However, one EAC can only be managed by one end-user. Furthermore, each application securely hosted in an EAC is a complete and independent unit. EAC separates the application in its own protected and reliable environment which is independent of the operating system and hardware and only has a limited access to the underlying VMs.

3) Elasticity
Elasticity of an EAC enables its computing processing power and data storage to be increased or decreased instantly in response to demand of an application running on the EAC. We deploy a dynamic load balancer to distribute workload of an application across multiple application servers or databases to achieve optimal resource utilisation of VMs, maximise throughput, minimize response time, and avoid overload of the application as shown in Figure 3.

4) Auto-scalability
Auto-scalability of the EAC system allows the computing resources of the system increase seamlessly during demand spikes to maintain the overall system performance, and decrease automatically during demand lulls to minimise costs. Increasing computing resources allows the system to create more virtual units for computation, and decreasing computing resources allows the EAC provider to save operating costs of VMs. As we mentioned earlier, both application servers and databases in the EAC system provide auto-scalability for maintaining a healthy EAC system performance.

III. CONCLUSION

In this paper, we propose a new architecture called Elastic Application Container that is an abstract representation which hides all its abstractions of the underlying VMs. One of the key issues that our proposed architecture attempt to solve is the complexity of using the computer resource level service to host and scale application. Our solution provides computing capabilities in terms of application servers and the storage capabilities in terms of database server for supporting a certain type of applications, mostly web applications. It may not be enough to cover large varieties of applications that can be deployed using IaaS. However, supporting web applications in the EAC system is used for demonstrating the EAC concept. Our ultimate aim is to provide more generic application containers which can host main stream applications using the EAC concept.

In the future work, we aim to provide more generic application containers which can host main stream applications. For example, we can take advantages of JVM’s architecture for deploying Java applications on the cloud. We believe that our EAC architecture has the potential to become the foundation of future application resource level model in this research area.