Towards a Description of Elastic Cloud-native Applications for Transferable Multi-Cloud-Deployments

Peter-Christian Quint, Nane Kratzke

Lübeck University of Applied Sciences
Center of Excellence for Communication, Systems and Applications (CoSA)
Lübeck, Germany

extended abstract

Elastic container platforms (ECP) like Docker Swarm, Kubernetes (k8s) and Apache Mesos have gotten more and more attention by practitioners in recent years [1]. Elastic container platforms fit very well with existing cloud-native application (CNA) architecture approaches [6]. Corresponding system designs often follow a microservice-based architecture [8, 5]. Nevertheless, the reader should be aware that the effective and elastic operation of such kind of elastic container platforms is still a question in research – although there are interesting approaches making use of bare metal [1] as well as public and private cloud infrastructures [6]. What is more, there are even more severe open issues like how to design, define and operate cloud applications on top of such container platforms. This is especially true for multi-cloud contexts. Such open issues in scheduling microservices to the cloud come along with questions regarding interoperability, application topology and composition aspects[1] [7] as well as elastic runtime adaption aspects of that kind of cloud-native applications [2]. The combination of these three aspects (multi-cloud interoperability, application topology definition/composition and elastic runtime adaption) is – to the best of the authors' knowledge – not solved satisfactorily so far. These three problems are often seen in isolation. In consequence topology based multi-cloud approaches often do not consider elastic runtime adaption of deployments (see [7] as research type representative). But to be honest, elastic runtime adaptive solutions focusing multi-cloud capability do not make use of topology based approaches as well (take [6] as an example). And finally, (topology-based) cloud-native applications making use of elastic runtime adaption are often inherently bound to specific cloud infrastructure services (like cloud provider specific monitoring, scaling and messaging services) making it hard to transfer these cloud applications easily to another cloud provider or even operate them across providers at the same time [5]. Furthermore, Heinrich et al. mention several research challenges and directions that come along with microservice focused monitoring and runtime adaption approaches [3]. All in all, it seems like cloud engineers (and researchers as well) just trust in picking only two out of three options.

In our further research we have developed a possibility for deploying CNA on a transferable, auto-scaling multi-cloud-platform. Figure 1 illustrates such an ECP based CNA deployment. The ECP (Layer 4 according to the CNA reference model [5]) enables multi-cloud deployments. For avoiding vendor lock-in, the ECP has to be transferable at runtime. While a descriptive cluster definition model can be used for describing the elastic platform [4], we have identified the need to also describe the application topologies without dependence to a special, static ECP with a domain-specific language (DSL) (see the DSL mark in the figure). The focus of the presentation is on the identification of description possibilities for the Layers 5 and 6. First, the particular characteristics of such a deployment have to be identified. Therefore, we analyzed and compared the architectures and concepts of container cluster-platforms. As representatives, we have chosen three most often used container cluster platforms Kubernetes, Docker Swarm

[1] These approaches are mostly TOSCA based.

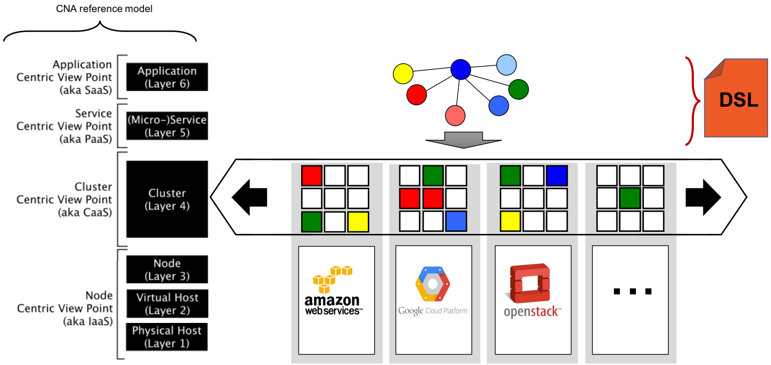
CNA reference model
Application Centric View Point (aka SaaS)
Application (Layer 6)
Service Centric View Point (aka PaaS)
(Micro-)Service (Layer 5)
Cluster Centric View Point (aka CaaS)
Cluster (Layer 4)
Node Centric View Point (aka IaaS)
Node (Layer 3)
Virtual Host (Layer 2)
Physical Host (Layer 1)
DSL
amazon webservices
Google Cloud Platform
openstack
...

Figure 1: An ECP based CNA deployment.

Mode and Apache Mesos with Marathon. After defining requirements for the DSL, we identified, investigated and analyzed approaches like TOSCA, CAML, MOCCA, MULTICLAPP, CoudMIG, CloudML and MODACloudsML.

In our presentation we will give an introduction about ECP based CNA and why they can help small and medium sized enterprises to use cloud services without vendor lock-in. We will talk about creating a DSL for CNA based on transferable ECP and our investigation about existing approaches. After presenting our requirements, the analysis and evaluation, we will derive our results. In the end we will present our conclusion and outlook, including several lessons learned.

References

[1] Carlos de Alfonso, Amanda Calatrava, and Germán Moltó. Container-based virtual elastic clusters. *Journal of Systems and Software*, 127(January):1–11, 2017.

[2] Maria Fazio, Antonio Celesti, Rajiv Ranjan, Chang Liu, Lydia Chen, and Massimo Villari. Open Issues in Scheduling Microservices in the Cloud. *IEEE Cloud Computing*, 3(5):81–88, sep 2016.

[3] Robert Heinrich, André van Hoorn, Holger Knoche, Fei Li, Lucy Ellen Lwakatare, Claus Pahl, Stefan Schulte, and Johannes Wettinger. Performance Engineering for Microservices: Research Challenges and Directions. In *8th ACM/SPEC on Int. Conf. on Performance Engineering Companion*, number April, pages 223–226, 2017.

[4] Nane Kratzke. Smuggling Multi-cloud Support into Cloud-native Applications using Elastic Container Platforms. 2016.

[5] Nane Kratzke and Rene Peinl. ClouNS - a Cloud-Native Application Reference Model for Enterprise Architects. In *2016 IEEE 20th International Enterprise Distributed Object Computing Workshop (EDOCW)*, pages 198–207, Vienna, sep 2016. IEEE.

[6] Nane Kratzke and Peter-Christian Quint. Understanding cloud-native applications after 10 years of cloud computing - A systematic mapping study. *The Journal of Systems and Software*, 126:1–16, 2017.

[7] Karoline Saatkamp, Uwe Breitenbücher, Oliver Kopp, and Frank Leymann. Topology Splitting and Matching for Multi-Cloud Deployments. In *8th Int. Conf. on Cloud Computing and Service Sciences (CLOSER 2017)*, 2017.

[8] Alan Sill. The Design and Architecture of Microservices. *IEEE Cloud Computing*, 3(5):76–80, 2016.