SeaClouds: An Open Reference Architecture for Multi-Cloud Governance[*]

Antonio Brogi[2], Jose Carrasco[1], Javier Cubo[1], Francesco D'Andria[3], Elisabetta Di Nitto[4], Michele Guerriero[4], Diego Pérez[4], Ernesto Pimentel[1], and Jacopo Soldani[2]

[1] Universidad de Málaga, Dept. Lenguajes y Ciencias de la Computación, Spain
`{josec,cubo,ernesto}@lcc.uma.es`
[2] University of Pisa, Dept. Computer Science, Italy
`{brogi,soldani}@di.unipi.it`
[3] ATOS Research and Innovation, Spain
`francesco.dandria@atos.net`
[4] Politecnico di Milano, Dip. di Elettronica, Informazione e Bioingegneria, Italy
`{elisabetta.dinitto,michele.guerriero,diego.perez}@polimi.it`

Abstract. We present the open reference architecture of the SeaClouds solution. It aims at enabling a seamless adaptive multi-cloud management of complex applications by supporting the distribution, monitoring and reconfiguration of app modules over heterogeneous cloud providers.

1 Motivation and Objectives of SeaClouds

Cloud computing is a model for enabling convenient and on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. Many private and public clouds have emerged during the last years, offering a range of different services at SaaS, PaaS levels aimed at matching different user requirements. Current cloud technologies suffer from a lack of standardization, with different providers offering similar resources in a different manner, which results in the vendor lock-in problem. This problem affects all stages of the cloud applications' lifecycle, ranging from their design to their operation. Application developers must know the features of the services to be used, and have a deep knowledge of the providers' API. To reduce the need of using deep knowledge, we can find solutions based on the use of standards, such as OASIS CAMP or OASIS TOSCA , DMTF CIMI , unified APIs, such as jClouds [5], or solutions like Docker[6]. These solutions are indeed very different, for example, whereas jClouds provides a cloud agnostic API library to provision and configure secure communications with cloud virtual machines, container-based solutions like Docker allow describing and deploying applications and their dependencies through containers on machines with the corresponding engine. Furthermore, different vendors (e.g., Dell, BMC, Abiquo) are currently commercialising tools for

---

[5] `https://jclouds.apache.org`
[6] `https://www.docker.com`

the provisioning, management and automation of cloud applications. A promising perspective, opened by the availability of different cloud providers is the possibility of distributing cloud applications over multiple heterogeneous clouds.

To mitigate this heterogeneity and get a vendor-agnostic solution, independent tools and frameworks have emerged as the result of integrating, under a single interface, the services of multiple public and private providers [5], [6], [3], [4]. These solutions offer a portable and interoperable environment where developers can describe their systems and select the resources that better fit their requirements. However, in all these attempts, platforms allow operating simultaneously with a single level of service to deploy applications, i.e., all the components of an application are deployed either at the IaaS level or all at the PaaS level (see, e.g., [1], [2], [7]). From this, with the goal of unifying cloud services, we propose a software architecture supporting the integration of IaaS and PaaS levels under a single interface. Then, this will allow developers to deploy their applications combining services offered by providers at any of these levels. In such a way, our proposal goes a step further in the software architecture and development of common APIs by unifying IaaS and PaaS services of different providers under the same interface, and using the TOSCA standard for the agnostic specification of applications' components and interdependencies. With such a proposal, we do not only reduce the need of vendor-specific knowledge to develop our applications for designing, deploying and operating them. In fact, such a homogenized API greatly improves portability and interoperability as well. This solution has been developed in the scope of the SeaClouds EU-funded research project[7]. This project aims to develop a new open source framework which performs Seamless Adaptive Multi-Cloud management of service-based applications.

2 The SeaClouds Platform: A Reference Architecture

SeaClouds is a software platform based on an open reference architecture to make more efficient the design, development, planning and management of complex business apps distributed on multi-cloud environments. It orchestrates services, platforms and infrastructures to ensure they meet the needs of cloud apps.

2.1 SeaClouds Functionalities and Open Reference Architecture

SeaClouds is a multi-cloud app management system based on standards and following the DevOps approach. The basic capabilities delivered to the developer via an innovative GUI are listed in the following.

– **Discovery and Matchmaking**. It allows querying available cloud offerings (IaaS/PaaS) determining suitable ones based on app requirements.
– **Cloud Service Optimizer**. It optimizes the deployment topology of an application across multiple clouds to address non-functional requirements.
– **Application Management**. It supports efficient deployment and multi-cloud governance of a complex application on various cloud offerings (IaaS and PaaS) leveraging cloud harmonized APIs and platform-specific adapters.

---

[7] http://www.seaclouds-project.eu

- **Monitoring and SLA enforcement**. It provides monitoring and independent metrics to allow operators to monitor the health and performance of applications, hosted across multiple clouds.
- **Repairing**. It scales horizontally and vertically cloud resources to maximize the performance of each module of an application.
- **Application migration**. It provides a seamless migration of the app modules between dissimilar (but compatible) clouds, allowing application portability.

Figure 1 presents the reference architecture and the design of the SeaClouds platform. The platform features a **GUI** used by two main stakeholders: **Designers and Deployment Managers**, and it considers **Cloud Providers** offering cloud resources. From SeaClouds platform functionalities standpoint, we can identify five major components in the architecture, plus a RESTful harmonized and unified **SeaClouds API** layer used for the deployment, management and monitoring of simple cloud-based applications through different and heterogeneous cloud providers, and exploiting a **Dashboard**.

- **SeaClouds Discoverer**. It is in charge of discovering available capabilities and add-ons offered by available cloud providers.
- **SeaClouds Planner**. It is in charge of generating an orchestration plan considering the application topology and requirements.
- **SeaClouds Deployer**. It is in charge of executing deployment plans generated by the Planner, and supports the integration of both IaaS and PaaS.
- **SeaClouds Monitor**. It is in charge of monitoring that the Quality of Services (QoS) properties of the application are not violated by the clouds in which they were deployed; and of determining together with the Deployer, the reconfiguration strategies to trigger the repairing or migration actions.
- **SeaClouds SLA Service**. It is in charge of mapping the low level information gathered from the Monitor into business level information, Quality of Business (QoB), about the fulfillment of the SLA defined.

A distinguishing aspect of the SeaClouds architecture is that it builds on top of two OASIS standards initiatives: TOSCA (at design time) to specify the topologies and generate the plans, and TOSCA and also CAMP is supported (at runtime) to manage the building, running, administration, monitoring and patching of applications in the cloud. Also, an important advance of SeaClouds as regards other software architecture solutions is the fact of unifying IaaS and PaaS of multiple vendors. Specifically, we propose to use a provider-agnostic TOSCA-based model of the topology of applications and their required resources, indistinctly using IaaS and PaaS services, which can be used for their deployment using (an extended version of) Apache Brooklyn.[8]

2.2 SeaClouds Implementation and Evaluation

The SeaClouds project has provided a solution which can be downloaded from the github repository[9]. The consortium identified Apache Brooklyn as the tool to deploy SeaClouds. To ensure a good level of quality assurance, a free Continuos

---

[8] Apache Brooklyn: `https://brooklyn.apache.org/`.
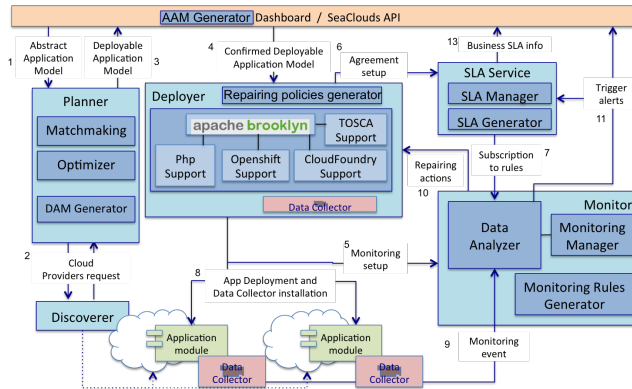[9] `https://github.com/SeaCloudsEU/SeaCloudsPlatform`

Fig. 1. Architecture of the SeaClouds Platform.

Integration (CI) and Continuos Distribution was set up, travis-ci.org. SeaClouds platform is built using Java language, distributing the artefacts generated from the source code, like jar file, war file etc, to a well-know public managed maven repository hosted by Sonatype (free for opensource projects).

The SeaClouds solution has been evaluated in several examples, with the main focus on two real use cases: i) Atos Software application and ii) Nuro Gaming application[10], both consisting of several components (servers, database) and distributed in heterogeneous cloud providers (IaaS and PaaS).

3 Conclusions and Future Issues

We have presented the SeaClouds platform, which provides an open source framework to address the problem of deploying, managing and reconfiguring complex applications over multiple clouds. The SeaClouds solution has addressed the main functionalities presented in previous section. As future work, the consortium has agreed to create the SeaClouds Alliance in order to continue working on some aspects, such as the improvement of the unification of providers supported by the deployment, and the reconfiguration covering replanning actions and data synchronization in database. Also, SeaClouds is an open source project, so it is open to receive more contributions and extensions from the community.

References

1. Yehia Elkhatib. Defining cross-cloud systems. *ArXiv e-prints*, 2016.
2. Eman Hossny et al. A case study for deploying applications on heterogeneous PaaS platforms. In *CloudCom-Asia*, 2013.
3. Fotis et al Gonidis. A development framework enabling the design of service-based cloud applications. In *Advances in Service-Oriented and Cloud Computing*. 2014.
4. Claus Pahl. Containerization and the PaaS cloud. *IEEE CLOUD*, 2015.
5. Linh Manh et al Pham. Roboconf: a hybrid cloud orchestrator to deploy complex applications. In *IEEE CLOUD*, 2015.
6. Mohamed et al Sellami. PaaS-independent provisioning and management of applications in the cloud. In *IEEE CLOUD*, 2013.
7. Dimitris Zeginis and Francesco D'Andria et al. *Scalable Computing: Practice and Experience*, 2013.

---

[10] Deliverables 6.3.3 and D6.4.3 `http://www.seaclouds-project.eu/deliverables`