PacificClouds: A Flexible MicroServices based Architecture for Interoperability in Multi-Cloud Environments

Juliana Oliveira de Carvalho[1,2,3], Fernando Trinta[1] and Dario Vieira[2]
[1]*Federal University of Ceará (UFC), Fortaleza, Brazil*
[2]*EFREI-Paris, Paris, France*
[3]*Federal University of Piauí (UFPI), Picos, Brazil*

Keywords: Multi-cloud, Microservices, Interoperability.

Abstract: Cloud Computing has become a popular IT service delivery model in recent years. While the cloud brings several benefits, there are still some challenges that need to be overcome to apply the cloud model in certain scenarios. One such problem is the so-called vendor lock-in since different cloud providers offer peculiar and often incompatible services, which results in the automatic migration impossibility of the application between cloud providers. This issue becomes even more problematic when thinking of future applications composed of services or components hosted by different cloud providers in a multi-cloud environment. Dealing with vendor lock-in in multiple clouds requires addressing two important challenges: interoperability and portability. Some solutions have been proposed to deal with both problems, but most of them fail to provide flexibility. Therefore, we propose PacificClouds, a novel architecture based on microservices for addressing interoperability in a multi-cloud environment. PacificClouds differs from previous works by providing greater flexibility due to the microservices architectural pattern. In this article, we also propose a definition of microservices and a comparative analysis of the works related to PacificClouds. Finally, we show the main challenges of PacificClouds, and we point out the future directions.

1 INTRODUCTION

Cloud computing represents the consolidation of the utility computing idea, i.e., users access the cloud as a service and utilize resources such as hardware, software and storage through a contract with a provider and pay for consumption according to established policies. Furthermore, cloud computing resources seems like an inexhaustible source of virtualized resources for the user; in consequence, the cloud allows the systems elastic growth (Vaquero et al., 2009).

Facing the cloud computing's success, cloud providers proliferate rapidly; each one them wants to offer infrastructure, platform, and services to satisfy the needs of its users in such a manner that they do not want to use other providers. The infrastructure and services of a provider differ somewhat from others. Therefore, providers become considerably specific. Thus, the problem known as vendor lock-in arises as a consequence the user applications are dependent on a single cloud provider technology. According to (Opara-Martins et al., 2015), vendor lock-in is one of the barriers to the adoption of cloud computing. The research further points out that organizations' desire to adopt the cloud for their benefit is primarily related to capacity, scalability, and speed, but they consider urgent the vendor lock-in treatment.

One method for treating vendor lock-in is the use of multiple clouds, although a small number of enterprises adopt this approach, their popularity is increasing. One reason for the low adoption of multiple clouds is cloud providers interest lack to promote interoperability and portability (Grozev and Buyya, 2014). According to this context, (Opara-Martins et al., 2015) observes the need for dealing with interoperability and portability in order to mitigate the problem of vendor lock-in. Section 2 describes multiple clouds, interoperability and portability.

In addition, (Opara-Martins et al., 2015) still notes that no approach exists that meets the needs of enterprises. Some of the reasons there is no proposed solution widely adopted to mitigate vendor lock-in are: most solutions are inflexible; software architects must adopt specific technologies for the application development with a steep learning curve, (Petcu, 2013).

In consideration of these limitations, the applica-

Carvalho, J., Trinta, F. and Vieira, D.
PacificClouds: A Flexible MicroServices based Architecture for Interoperability in Multi-Cloud Environments.
In *Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)*, pages 448-455
ISBN: 978-989-758-295-0

tions must be designed and developed as native cloud applications. For that matter, the applications are built with the focus on integration with the cloud model, to obtain full cloud advantages; it, also ensures other features labeled as IDEAL (**I**solated state, **D**istribution, **E**lasticity, **A**utomated management, **L**oose coupling). In this manner, the native cloud application can facilitate the application deployment in multiple clouds, hence help treat interoperability and portability (Fehling et al., 2014).

Microservices can aid in obtaining the native cloud application's characteristics; therefore, they focus on aspects as componentization of small and lightweight services, agile and DevOps practices, infrastructure automation with continuous delivery features, decentralized data management, and decentralized governance among services. The microservices promise more agility, more delivery speed, and more scalability compared with traditional monolithic applications, resulting in less overall cost (Newman, 2015), (RV, 2016). In Section 3, we describe, present challenges and propose a definition for microservices.

In this work, we propose a novel architecture based on microservices to address interoperability for a multi-cloud environment, called PacificClouds, in order to mitigate vendor lock-in and aid to obtain full cloud advantages. PacificClouds promotes flexibility for user's decisions related to requirements and application architecture, and for choosing the clouds to execute each microservice of the application. Therefore, PacificClouds differentiate from the other works as it possesses features related to the use of the microservice and native cloud application, and mainly due to the flexibility. In Section 4, we present in details PacificClouds.

PacificClouds is flexible as it allows the use of several independent cloud providers. The providers can use different technology backgrounds and let software architects utilize several techniques in developing the applications, update the software and the user and application requirements at runtime. The proposed architecture is lightweight, since each module is responsible only for one business logic. It possesses decentralized governance of the application because each part of the application is independent of one another, i.e., the application modules possess loosely coupled. The applications are native cloud applications based on microservices. Thus, the applications can be deployed/redeployed/updated at runtime, analyzing the application requirements and the user's SLA. Therefore, software architects do not need to be concerned with the cloud selections, application deployment and what technologies are used in application development. In addition, they can request, in runtime, a new SLA by analyzing the monitoring metrics.

According to the context aforementioned, the contributions of this article are given below:

- The proposition and description of PacificClouds, a novel architecture, which provides greater ease in the development of native cloud application, and more flexibility in the deployment and execution of an application in multiple clouds environment because of microservice use.
- A more comprehensive definition of microservice.
- A comparative analysis of the works related to PacificClouds. The related work are described in Section 5 and a discuss is presented in Section 6.

Last, in Section 7, we show this work's primary contributions, describe PacificClouds' main developing challenges and discuss future directions.

2 MULTIPLE CLOUDS

Multiple clouds enable applications to take advantage of the best features of different components provided by several cloud providers. Since there is still no standardized taxonomy for the subject, different terms are used in the literature and they often have the same meaning or they are a branch of an existing one.

In this work, we adopt the definitions used by (Petcu, 2014b), who classifies different multiple cloud application scenarios as delivery models and presents three main proposals for these models. The first, called multi-cloud, is a delivery model that does not include a prior agreement among the cloud providers, a third party assumes the role of intermediary. In a cloud federation, the second delivery model exists as an established collaboration arrangement among cloud providers to share their resources. The third delivery model, called inter-cloud, can be on both a cloud federation and multi-cloud, but the scalable and opportunistic services must be dynamic, and inter-cloud must possess the cloud broker, which is an intermediary actor in the relationship between the provider and the consumer.

The use of multiple clouds brings several advantages, and through them, we can achieve the full benefits of cloud properties such as elasticity and pay-as-you-go (Mezgár and Rauschecker, 2014), (Silva et al., 2013). However, the multiple clouds bring several challenges, as well, e.g., interoperability and portability related to mitigating vendor lock-in. We consider portability the ability to allow customers to migrate data and systems from one cloud to another and interoperability capacity to allow customers to use services across multiple clouds (Rezaei et al., 2014).

3 MICROSERVICES

In the literature, microservices have several definitions; in this section, we present four of them and propose a definition for microservices. In addition, we present the main challenges of microservices.

According to (Newman, 2015), microservices are small, autonomous services that work together. In the second definition, given by (Pautasso et al., 2017), microservices are an architectural style as an approach to developing a single application as a suite of small services, each running in its process and communicating with lightweight mechanisms, often an Http resource. In the definition provided by (RV, 2016), microservices are an architectural style or an approach to building IT systems as a set of building capabilities that are autonomous self-contained, and loosely coupled. Last, (Dragoni and Lluch-lafuente, 2016) proposed two definitions related to microservices: one for microservice and another for microservice architecture. Therefore, microservice is a cohesive, independent process interacting via messages, and microservice architecture is a distributed application where all its modules are microservices.

We propose a definition for microservices based on the definitions cited in this work, because we understand that our definition of microservices shows the meaning of microservices in the context of multi-clouds and more specifically in a scenario which the application is distributed over multiple clouds.

Definition (Microservices): are a set of autonomous, independent, self-contained services, in which each service has a single goal, is loosely coupled, and interact to build a distributed application. Although there are several benefits of the microservices, they possess some challenges. The main challenges related to the adoption of microservices in the construction of a system are: (i) The software architects must change the manner of thinking systems design; (ii) the complexities of the distributed systems; (iii) the need for scaling the systems differently and ensuring that microservices are resilient (Dragoni and Lluch-lafuente, 2016).

4 PacificClouds

In order to promote greater adoption of cloud computing, we propose PacificClouds to mitigate vendor lock-in. PacificClouds is a novel architecture that addresses the interoperability of distributed application in multiple clouds via microservices. Thus, PacificClouds intends to provide flexibility for the software architect both in making decisions related to application requirements and the use of the clouds. In addition, PacificClouds can migrate microservice in runtime to better meet application and user needs.

In this section, we describe a scenario that shows some of the PacificClouds contributions. Next, we introduce and explain the PacificClouds architecture.

4.1 Scenario

In this section, we describe a scenario of a web application designed and deployed in to a fashion store named AnaGomesFashion, which is expanding its customers as well as the frontiers of action. AnaGomesFashions is a store that sells beachwear, intimate and fitness fashion. Each fashion segment has several suppliers distributed in Brazil's states. It also offers specialized services to its clients through partnerships with nutritionists, physiotherapists, and personal trainer. AnaGomesFashions' online store allows access to its clients through desktop or mobile phone. The application has three types of users: Client, ProfessionalService, and AnaGomesFashions.

Client use the application to purchase products as well as to get the professional services offered by the store. ProfessionalService users are professionals who partnered with the store to offer services to AnaGomesFashions' clients. The AnaGomesFashions user is responsible for registering products and suppliers, as well as monitoring sales, services and finances.

AnaGomesFashions virtual fashion store offers a variety of services such as: three fashion segments sales service; product fitting service according to the client's body; services provided in partnership with three types of professional; progress monitoring service; follow-up service for the delivery process, by the customer and by the store; sales analysis to help determine which products should be purchased from the store through its suppliers; after-sales service to analyze customer satisfaction regarding the products and services offered by AnaGomesFashions.

4.2 PacificClouds Architecture

According to the PacificClouds goals described in section 1, we adopt the multi-cloud delivery model, for it brings greater flexibility. In relation to portability, we assume the three categories used by (Petcu et al., 2013) and we report the IaaS and PaaS levels for the portability requirements. According to interoperability levels, we have considered the following criteria, (Nogueira et al., 2016): we have adopted both syntactic and semantic level interoperability, associated with the agreement level. Regarding the

adoption level, we assume the communication mechanisms adopted for microservices. Related to the deployment level, we consider both horizontal and vertical interoperability in IaaS and PaaS service models. We assume the asynchronous communication in the interactions patterns between cloud levels. Finally, associated with the end-user level, we adopted user-centric interoperability.

PacificClouds intends to address interoperability in the multi-cloud that focuses on the user's perspective through microservices and native cloud applications. In this manner, it also addresses an important issue related to the adoption of cloud computing, flexibility. Thus, in this subsection, we present and describe the architecture PacificClouds.

In order for architecture to cover all objectives of PacificClouds, aforementioned in this article, it must include the following functionalities: (i) identify the application microservices as well as its requirements; (ii) identify user requirements; (iii) discover and monitor the capabilities of the available clouds; (iv) manage the application deployment and execution; (v) monitor the deployment and execution of each application microservice; (vi) monitor user and application requirements, as well as the capabilities of the clouds involved in running the application; (vii) provide the deployment of each application microservice; (viii) manage the cloud resources of each application microservice; (ix) provide application performance information to the user, as that the user can modify the requirements at runtime. Therefore, PacificClouds architecture consists of three parts: PacificClouds API, Adapter, pacificClouds Core. The Figure 1 illustrates the PacificClouds architecture; after, we describe each PacificClouds architecture component.

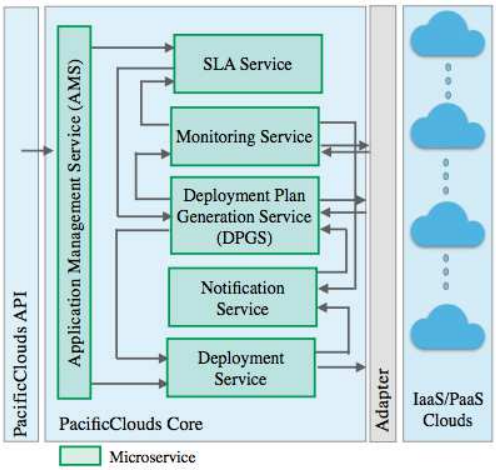

Figure 1: PacificClouds Architecture.

PacificClouds API is the communication interface between the software architect and the PacificClouds Core. The PacificClouds API receives two artifacts from the software architect: the application and users' SLA, which it sends to the PacificClouds Core. The PacificClouds Core provides information, for the software architect via PacificClouds API, both about resources used by an application and about application and user's requirements.

Adapter is the communication interface between the multiple available clouds and the PacificClouds Core. The adapter sends the capabilities and information about the cloud resources used by the application to PacificClouds Core. it also sends the microservices received from the PacificClouds Core to the clouds.

PacificClouds Core is the central part of the architecture. It is responsible for discovering the application microservices and their respective requirements; determining the clouds; deploying the microservices; managing cloud resources used by the application; monitoring information about the resources used by the application at runtime; monitoring all capabilities of the clouds; and verifying the user requirements. Therefore, it is composed of six microservices to perform all its tasks. These microservices themselves communicate through synchronous calls and asynchronous events. In the next subsections, we describe these microservices.

4.2.1 Application Management Service

Application Management Service (AMS) receives an application from the PacificCloud API and it is responsible for registering it in the AppData (via Applications Register Service (ARS)) and sending it to the Microservices Identification Service (MIS). Then, the AMS extracts each application microservice (via MIS), in addition to the requirements of each microservice and the user (via Requirements Discovery Service (RDS)). In addition, the AMS registers the user profile in the USrData (via Users Register Service (URS)), according to Figure 2.

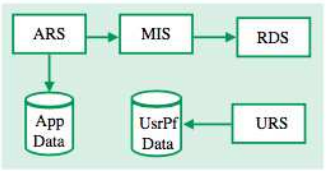

Figure 2: The services of the AMS microservice.

4.2.2 SLA Service

Figure 3 illustrates the SLA Service, which is responsible for mapping the user's and microservice's requirements received from the AMS, as well as the cloud resources data used by the application received from

the Monitoring Service (via Data Mapping Service (DMS)). In addition, the SLA Service develops procedures and methods for evaluating and reporting information on the application performaence to the software architect (via Data Analysis Service (DAS)).

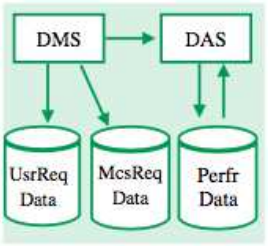

Figure 3: The services of the SLA Service microservice.

4.2.3 Notification Service

Notification Service, shown in Figure 4, is responsible for notifying DPGS microservice (via Deployment Notification Service (DplNS)) about changes in the user requirements or the microservice requirements, or the updating of the application; it also informs DPGS about violation of the criteria by cloud, registers the notification data in DplNData and notifies the software architect for deploying or redeploying of the application (via User Notification Service (UsrNS)).

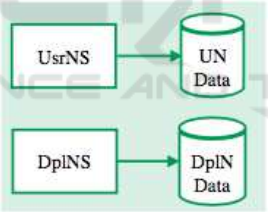

Figure 4: The services of the Notification microservice.

4.2.4 Monitoring Service

Monitoring Service, shown in Figure 5, is responsible for monitoring the cloud resources used by the application at runtime through the adapter. Monitoring Service is composed by three services: (i) Cloud Resources Register Service (CRRS) receives from DPGS the deployment plan data. After, it registers it in MsRsData and it sends it to CRDAS; (ii) Cloud Resources Detection Service (CRDetS) detects all capabilities of the clouds, which are deployed by the microservices. Next, it sends them to CRDAS; (iii) Cloud Resource Data Analysis Service(CRDAS) is responsible for comparing the application performance received from CRDS with the deployment plan data; next, it sends the cloud resources data used by the application to SLA Service and sends a notification to Notification Service if any violation happens.

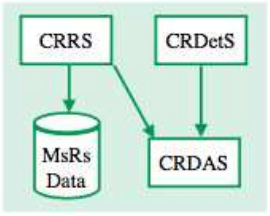

Figure 5: The service of the Monitoring microservice.

4.2.5 Deployment Plan Generation Service

The Figure 6 illustrates Deployment Plan Generation Service (DPGS), which determines the cloud providers for deploying the microservices without violating application and user requirements via Cloud Selection Service (CSS). For this, it receives from the SLA Service all requirements. Afterward, it must generate a microservices deployment plan based on the requirements and the capabilities of the clouds obtained by Cloud Capabilities Discovery Service (CCDS). Also, it analyzes the notifications received from Notification Service and the requirements received from SLA service; next, if necessary, it generates a redeployment plan of some microservices.

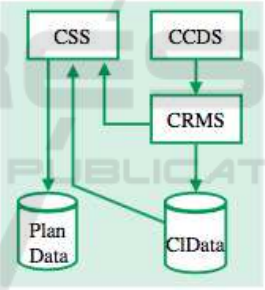

Figure 6: The deployment plan generation microservice.

4.2.6 Deployment Service

Deployment Service, illustrated in Figure 7, receives the microservices from AMS through Application Microservices Reception Service (AMRS), besides registering each microservice in McsData. Next, Deployment Service identifies the cloud for each microservice according to deploy plan via Microservice Association Service (MAS); after, it deploys the microservices via Microservice Deployment Service (MDplS).

We can observe that PacificClouds intends to manage the deployment and execution process of the distributed application in multiple clouds to better meet the needs of the application and consequently improve end-user satisfaction with the application performance. Thus, the PacificClouds will be able to help the software architecture in the process of de-

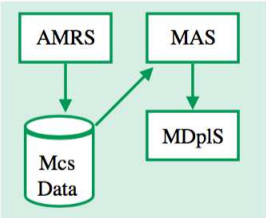

Figure 7: The Deployment Service.

signing and developing the application. For this, PacificClouds must identify each of the services offered by the application, as well as its requirements. In addition, PacificClouds must select the most suitable clouds for each one of the application services, and to deploy each of the application services according to the clouds selection process. The PacificClouds must also monitor application execution by observing the clouds involved in the application deployment to detect any violation of requirements, and monitor changes in application requirements, application services' updates and capabilities of the clouds to identify the need to re-deploy some application services.

5 RELATED WORK

In this section, we describe an overview of the six most relevant works related to PacificClouds in regars to treating the interoperability in multiple clouds, in which each of them proposes a different solution to mitigate vendor lock-in.

Cloud4SOA introduces a broker-based architecture whose primary goal is to address semantic interoperability challenges at the PaaS layer, based on SOA architecture (Dandria et al., 2012). The mOSAIC aims at offering access to heterogeneous resources from multiple clouds. Under a high-level perspective, an API and a PaaS compose the mOSAIC, which PaaS allows us to deploy, configure and manage applications running on IaaS. (Petcu et al., 2013).

MODAClouds offers a set of techniques for development and runtime operation management of multiple clouds applications. It delivers an open-source IDE for the high-level design, cloud service selection, early prototyping, QoS assessments, semi-automatic code generation, and multiple cloud applications automatic deployment (Ardagna et al., 2012). The RASIC focuses on resolving the semantic interoperability issues in IaaS and introducing a user-centric approach for applications that use cloud resources. It facilitates the smooth switching among cloud providers and allows the composition and services integration of different clouds (Loutas et al., 2010).

Automated Setup of Multi-Cloud Environments for Microservices Applications proposes an automated approach for the selection and configuration of cloud providers for microservices based applications. But it does not deal directly with applications deployment and does not consider costs and quality of service for optimizing the selection as these depend heavily on application usage (Sousa et al., 2016). The SeaClouds focuses on deploying and managing complex multi-component applications over heterogeneous clouds. The approach is based on the concept of service orchestration and is designed to fulfill functional and non-functional properties of the application. In addition, the services can be deployed, replicated, and administered using standard harmonized APIs such as CAMP specification and Cloud4SOA (Brogi et al., 2015).

6 DISCUSS

In this section, we present an analysis of the works described in Section 5 and the PacificClouds. For this, we offer a summary of the features of all the solutions described in this article, including PacificClouds, related to promoting interoperability in a multi-cloud environment. We consider 10 aspects described in this article, because through them we can observe the level of flexibility of a given solution.

In Table 1, there is the only aspect that all solutions promote the semantic interoperability. Some solutions use the semantic interoperability in the applications portability, while others use it in the interoperability between application modules distributed over multiple clouds, and it includes solutions that use it to treat vertical interoperability between service models.

One of the characteristics described in Table 1 refers to the service model used by the solutions. MODAClouds and ASMEMA treat interoperability in the three primary service models: IaaS, PaaS, and SaaS. Despite addressing the three levels, MODAClouds only addresses horizontal interoperability for cloud federation. ASMEMA does not address any of the vertical and horizontal interoperability levels for the multi-cloud delivery model. Two other solutions, Cloud4SOA and RASIC, address interoperability in only one of the service models. Cloud4SOA addresses interoperability at the PaaS level; it does not address any vertical and horizontal interoperability levels, and it uses the hybrid delivery model, which is a cloud federation that has a private cloud and at least one public cloud (Petcu, 2014a). RASIC treats interoperability in the IaaS service model for multi-cloud

Table 1: Summary of the characteristics the related work with PacificClouds.

Projects Characteristics	Solutions for Multiple Clouds						
	Cloud4SOA	mOSAIC	MODAClouds	RASIC	ASMEMA	SeaClouds	**PacificClouds**
Service Model	PaaS	IaaS/PaaS	IaaS/PaaS/SaaS	IaaS	IaaS/PaaS/SaaS	IaaS/PaaS	IaaS/PaaS/SaaS
Delivery Model	Hybrid	Hybrid	Cloud Federation	Multi-Cloud	Multi-Cloud	Multi-Cloud	Multi-Cloud
Portability	✔	✔	✔	✘	✘	✔	✔
Vertical Interoperability	✘	✔	✘	✘	✘	✔	✔
Horizontal Interoperability	✘	✘	✔	✔	✘	✔	✔
Application Distributed	✘	✘	✘	✔	✔	✔	✔
Different Technological Background	✘	✔	✔	✔	✔	✘	✔
Semantics	✔	✔	✔	✔	✔	✔	✔
Flexibility	✘	✘	✘	✘	✔	✘	✔
Decentralized Governance	✘	✘	✘	✘	✘	✘	✔

✔ The work has the characteristic ✘ The work does not have the characteristic

but addresses only the horizontal interoperability level. Finally, mOSAIC, SeaClouds, and PacificClouds treat interoperability in the IaaS and PaaS service models. mOSAIC uses the hybrid delivery model; it deals with vertical interoperability as it makes application portability. SeaClouds and PacificClouds address vertical and horizontal interoperability levels for the multi-cloud delivery model as well as promote application portability.

RASIC, ASMEMA, SeaClouds, and PacificClouds address interoperability for distributed applications in multiple clouds geographically dispersed, while the other solutions address only application portability between clouds. In relation to the background technologies in clouds, just Cloud4SOA and SeaClouds do not use clouds that have different background technologies, that is, all clouds involved in the portability or interoperability of applications must have the same background technology for Cloud4SOA and SeaClouds.

ASMEMA and PacificClouds solutions are flexible because they allow deploying each application module in a different cloud, and they enable one module to use different technologies than another module. Also, it enables one module to be updated independently of other modules. The difference between ASMEMA and PacificClouds is that the former does not do decentralized governance by not treating either interoperability or portability. Therefore, PacificClouds is the only proposed solution that promotes the decentralized governance of the applications, to the best of our knowledge, allowing the application modules to have a loosely coupled, in addition to generating less traffic in the network and therefore allowing greater flexibility.

According to the analysis performed in this subsection, we can observe that PacificClouds allows greater flexibility regarding interoperability in a multi-cloud environment. Because of allowing an application to be distributed over several clouds according to the requirements of the application and the user regardless of the technology used, through the use of native cloud application and microservices.

7 CONCLUSION

We show in this work a novel architecture, called PacificClouds, that supports the interoperability across multiple clouds, allowing the software architect to choose the application requirements and architecture with no need to worry about the application deployment in the clouds. The architecture also allows the use of several independent clouds and different technologies backgrounds. Thus, PacificClouds achieves another goal of flexibility, which helps more cloud computing adoption. The PacificClouds architecture is based on microservices to help achieve their goals. In this manner, PacificClouds provides other contributions: a) lightweight, since each microservice has only one function; b) decentralized governance of the application, because each microservice is independent of one another, which allows loosely coupled; c) the software architect can request at runtime a new SLA by analyzing the monitoring metrics; d) facilitates the use of native cloud application in application development.

One of the contributions of PacificClouds is the flexibility; however, to achieve this, some challenges need to be overcome. One of the more significant challenges is to reach the interoperability of clouds that possess distinct background technological; in this manner, software architects are primarily concerned with application development and the user's SLA. PacificClouds must deploy the microservices that compose applications through a deployment plan, and it must monitor the requirements of the application and user in runtime, beyond the capabilities of the clouds.

In addition, PacificClouds must build the decentralized governance of the application so that the solution is flexible. Finally, it must allow the application microservices to use distinct technologies.

REFERENCES

Ardagna, D., Nitto, E. D., Milano, P., Petcu, D., Sheridan, C., Ballagny, C., Andria, F. D., and Matthews, P. (2012). MODAC LOUDS : A Model-Driven Approach for the Design and Execution of Applications on Multiple Clouds. *Modeling in Software ...*, pages 50–56.

Brogi, A., Fazzolari, M., Ibrahim, A., Soldani, J., Wang, P., Informatica, D., Carrasco, J., Cubo, J., Dur, F., Pimentel, E., Mal, U. D., Nitto, E. D., Elettronica, D., Bioingegneria, I., Milano, P., and Andria, F. D. (2015). Adaptive management of applications across multiple clouds : The SeaClouds Approach. 18(1):1–14.

Dandria, F., Bocconi, S., Cruz, J. G., Ahtes, J., and Zeginis, D. (2012). Cloud4SOA: Multi-cloud application management across paas offerings. *Proceedings - 14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2012*, pages 407–414.

Dragoni, N. and Lluch-lafuente, A. (2016). Microservices : yesterday , today , and tomorrow. (June).

Fehling, C., Leymann, F., Retter, R., Schupeck, W., and Arbitter, P. (2014). *Cloud Computing Patterns*.

Grozev, N. and Buyya, R. (2014). Inter-Cloud architectures and application brokering : taxonomy and survey. (December 2012):369–390.

Loutas, N., Peristeras, V., Bouras, T., Kamateri, E., Zeginis, D., and Tarabanis, K. (2010). Towards a Reference Architecture for Semantically Interoperable Clouds. In *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, pages 143–150.

Mezgár, I. and Rauschecker, U. (2014). The challenge of networked enterprises for cloud computing interoperability. *Computers in Industry*, 65(4):657–674.

Newman, S. (2015). *Building Microservices*. O'REILLY.

Nogueira, E., Moreira, A., Lucrédio, D., Garcia, V., and Fortes, R. (2016). Issues on developing interoperable cloud applications: definitions, concepts, approaches, requirements, characteristics and evaluation models. *Journal of Software Engineering Research and Development*, 4(1):7.

Opara-Martins, J., Sahandi, R., and Tian, F. (2015). Critical review of vendor lock-in and its impact on adoption of cloud computing. *International Conference on Information Society, i-Society 2014*, pages 92–97.

Pautasso, C., Zimmermann, O., Amundsen, M., Lewis, J., and Josuttis, N. (2017). Microservices in Practice, Part 1: Reality Check and Service Design. *IEEE Software*, 34(1):91–98.

Petcu, D. (2013). Multi-Cloud: Expectations and Current Approaches. In *Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds*, MultiCloud '13, pages 1–6, New York, NY, USA. ACM.

Petcu, D. (2014a). Consuming Resources and Services from Multiple Clouds. *Journal of Grid Computing*, 12(2):321–345.

Petcu, D. (2014b). Portability in Clouds : Approaches and Research Opportunities, Scalable Computing : Practice and Experience. 15(3):251–270.

Petcu, D., Macariu, G., Panica, S., and Crăciun, C. (2013). Portable cloud applications - From theory to practice. *Future Generation Computer Systems*, 29(6):1417–1430.

Rezaei, R., Chiew, T. K., Lee, S. P., and Shams Aliee, Z. (2014). A semantic interoperability framework for software as a service systems in cloud computing environments. *Expert Systems with Applications*, 41(13):5751–5770.

RV, R. (2016). *Spring Microservices*. Pack Publishing.

Silva, G. C., Rose, L. M., and Calinescu, R. (2013). A Systematic Review of Cloud Lock-In Solutions. *2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, pages 363–368.

Sousa, G., Rudametkin, W., and Duchien, L. (2016). Automated Setup of Multi-Cloud Environments for Microservices-Based Applications. *9th IEEE International Conference on Cloud Computing*.

Vaquero, L. M., Rodero-Merino, L., Caceres, J., and Lindner, M. (2009). A break in the clouds. *ACM SIGCOMM Computer Communication Review*, 39(1):50.