# Deployment Experiment of an Elastic Cloud Application

Technical University of Vienna,
Advanced Internet Computing,
(Jan 2013)

Soodeh Farokhi*
1228800

Gajo Gajic
0828150

Martin Kalany
0825673

Jia Wei
0035204

## ABSTRACT

Scalability is said to be one of the major advantages brought by the cloud paradigm and, more specifically, the one that makes it different to an âĂIJadvanced outsourcingâĂİ solution [11].

## 1. INTRODUCTION

goal: providing scalable software platforms in clouds

### 1.1 CloudScale Features

1. CloudScale, on the other hand, allows application developers to retain full control over their application. CloudScale applications are not bound to any specific cloud provider, are easy to migrate, work well in the context of private or hybrid clouds, and support a wider variety of applications, while still providing an abstraction comparable to commercial PaaS solutions [8]..

2. The main disadvantage of all of those systems is that they imply a significant loss of control for the developer and also they typically require the usage of a given public cloud (typically provided by the same vendor), imply the usage of propri- etary APIs, and restrict the types of applications that are supported [8].

3. CloudScale middleware for transparently scaling applications using an IaaS cloud.

4. The CloudScale approach provides a middle ground between common IaaS offerings, which provide great control over the application, but do so at the costs of high deployment effort, and PaaS offerings, which are easy to use, but provide little control. [9]

## 2. STATE OF THE ART

*In alphabetical order



There are essentially two schools of thought of how one should build any distributed system. On the one hand,many approaches aim at hiding the complexity of distribution behind convenient abstractions, such as remote procedure call systems. On the other hand, some claim that such abstractions always have to be leaky, and, hence, should be avoided altogether. we follow the former school of thought. Essentially, CloudScale provides an abstraction that makes elastic applications running on top of an IaaS cloud seem like regular, non-distributed Java applications.

The possible mechanisms for scalability of applications in Clouds has been depicted in Fig.1, and CloudScale approach falls into "container replication" category in the platform layer (the container being CloudScale in this case) as well as Aneka and AppScale.

### 2.1 Container-level Scalability

This category is like CloudScale approach (description of this kind of approach is described in [11].

Aneka [12]: For high loads, and to avoid overprovisioning of resources, it would be useful to be able to federate clouds so components can be run in external/public clouds if needed. Aneka is able to deploy containers and run users applications in several IaaS providers [11].

AppScale [3]: Platforms will require container replicas to be deployed or released dynamically to handle load variations. AppScale can scale the VMs used to host containers depending on actual application demand, automatically configuring the load balancers [11].

We can consider two main categories for the WR:

### 2.2 Related Research (scientific approaches)

(1) paper [10] (MDA): Enough about standardization, let's build cloud applications

(2) paper [6]:(cite CloudScale) A topology-aware adaptive deployment framework for elastic applications

(2-1) **Ubuntu Juju** [1]: Basic services are described as predefined charms

(2-2) **RIM. Carina Environment Manager** [2]: It automate and speed up the deployment of services onto the OpenNebula private clouds

[1]https://juju.ubuntu.com/
[2]https://github.com/blackberry/OpenNebula-Carina

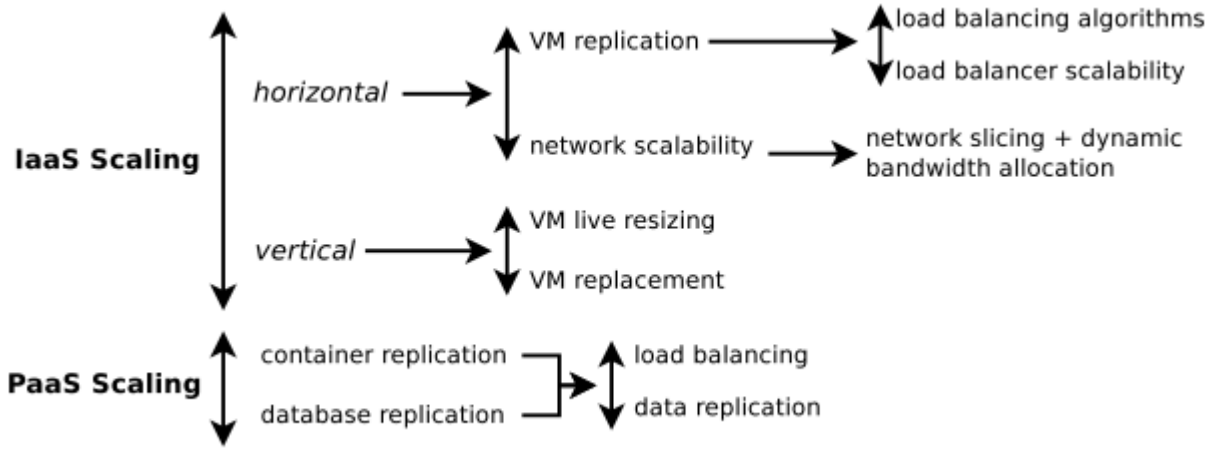


**Figure 1: Available Mechanisms for Holistic Application Scalability [11]**

(2-3) CA 3TerasâĂć [3]: 3Teras AppLogic automates complex application deployment.

(2-4)**Aeolus** /citedi2012towards: AeolusâĂŹs component model specifies compositions of services to automate deployments, planning of day-to-day activities such as software upgrade planning, service deployment, elastic scaling

(3)Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (IaaS) cloud applications [1]

(4)Component-based scalability for cloud applications [5]

(5)Look up!: your future is in the cloud [7]

(6)Orleans: cloud computing for everyone [2]

(7)Lightweight Resource Scaling for Cloud Applications [4]

**Aneka** is a .NET-based platform with a focus on enabling hybrid cloud applications. Unlike CloudScale, Aneka is more akin to traditional Grid computing middleware, providing a relatively low-level abstraction based on the message passing interface (MPI). In general, Aneka seems suitable for building scientific computing applications, but less so for enterprise applications [8]

**AppScale**, on the other hand, specifi- cally (and exclusively) targets Online Transaction Processing (OLTP) style enterprise applications [8].

another platform with a focus on data processing has been proposed by the BOOM research project [4]. BOOM builds on a custom, declarative programming model and is clearly geared towards data analytics [8]..

The main contribution of CloudScale over all of these models is that Cloud- Scale has amore general claim, and is able to handle a wide variety of elastic application types, including data-intense, processing-intense and OLTP style web applications [8]..

The European commission funded project **Contrail** [2] is currently building another PaaS for hosting relatively flexible elastic applications, called ConPaaS [28] (Contrail PaaS). ConPaaS supports web and high-performance applications built in Java or PHP, and can be used in combination with Amazon EC2 or an OpenNebula private cloud. However, ConPaaS seems to operate on a lower level of abstraction than CloudScale. This PaaS environment consists of elastically hostedWeb services, such as web or database servers, and mostly integrates existing stand-alone components [8].

## 2.3 Related tools and commercial approaches

A service particularly related to our work is **CloudBees RUN@Cloud** [4], which provides continuous integration and an elastic platform for hosting Enterprise Java Beans (EJB) applications [8].

Other PaaS of- ferings that support the Java programming language include the well-known GAE, AmazonâĂŹs Elastic Beanstalk services, or MicrosoftâĂŹs Azure [8].

Outside of the Java world, **Heroku** is gaining traction as a provider of PaaS for dynamic scripting languages, such as Ruby or Python[5] [8].

AppScale can be also considered as a commercial product and it allows users to build their own GAE compliant PaaS on top of any private or public IaaS service. We consider this system a significant step forward in comparison to other vendors [8].

Many core features and terms of CloudScale, as introduced in Section 3.1, are akin to well-known Java remote procedure call technology, e.g., Java RMI or EJB. However, note that the main contribution of CloudScale is not in enabling remoting (as RMI and EJB do), but rather in enabling elasticity. Hence, the actual conceptual overlap between CloudScale and these frameworks is not particularly large. In this paper, we more focus on the first category by investigating the scientific state of the art in elastic cloud applications, while just introduce some existing tools and technologies for the second category [8].

## 2.4 Other aspects

1. efficiently deploying applications in the cloud

2. distributed deployment and hosting of (Java) applications: most replicating and clustering of **Enterprise Java Beans** (EJB) implementations lack the dynamism of Cloud-

[3]http://www.ca.com/us/cloud-platform.aspx

[4]http://www.cloudbees.com/

[5]https://www.heroku.com/

Scale, and do not provide resource management strategies or transparent code distribution [9].

3. Specialized programming models: **Aneka** is a platform for deployment of .NET-based applications, employing a specialized programming model. The BOOM initiative at UC Berkeley aims at simplifying declarative programming for the cloud. Whereas BOOM is mostly suited for data analytics, Cloud- Scale is targeted at general-purpose programming and deployment of arbitrary Java applications [9].

4. PaaS and SaaS: From a technical viewpoint, these providers do little more than allowing scalable remote access to existing software developing platforms, enriched with rich user interface and Web 2.0 experience; The main difference between CloudScale and those commercial SaaS and PaaS solutions is that, when using CloudScale, developers retain full control over their application. That is, even though CloudScale hides some scalability-related issues from developers, they are still free to customize the way CloudScale works to their own needs, either by implementing custom-scaling policies, adapting the CloudScale framework itself, or managing some COs in the application manually [9].

5. Java-based remoting: e.g., **Java RMI, Enterprise Java Beans** (EJB) or CORBA. These frameworks, while technically similar, provide only limited support for automated scaling. How- ever, the ideas behind the CloudScale middleware could be implemented on top of these technologies, enhancing, e.g., the CORBA remoting model with automated load balancing in the cloud.

# 3. CONCLUSION

# 4. REFERENCES


[1] P. C. Brebner. Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (iaas) cloud applications. In *Proceedings of the third joint WOSP/SIPEW international conference on Performance Engineering*, pages 263–266. ACM, 2012.

[2] S. Bykov, A. Geller, G. Kliot, J. R. Larus, R. Pandya, and J. Thelin. Orleans: cloud computing for everyone. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 16. ACM, 2011.

[3] N. Chohan, C. Bunch, S. Pang, C. Krintz, N. Mostafa, S. Soman, and R. Wolski. Appscale design and implementation. 2009.

[4] R. Han, L. Guo, M. M. Ghanem, and Y. Guo. Lightweight resource scaling for cloud applications. In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, pages 644–651. IEEE, 2012.

[5] S. Kächele and F. J. Hauck. Component-based scalability for cloud applications. In *Proceedings of the 3rd International Workshop on Cloud Data and Platforms*, pages 19–24. ACM, 2013.

[6] M. Keller, M. Peuster, C. Robbert, and H. Karl. A topology-aware adaptive deployment framework for elastic applications. In *Intelligence in Next Generation Networks (ICIN), 2013 17th International Conference on*, pages 61–69. IEEE, 2013.

[7] J. R. Larus. Look up!: your future is in the cloud. In *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation*, pages 1–2. ACM, 2013.

[8] P. Leitner, Z. Rostyslav, W. Hummer, C. Inzinger, P. Leitner, W. Hummer, and C. Inzinger. CloudScale : Efficiently Implementing Elastic Applications for Infrastructure-as-a-Service Clouds. Technical report, 2013.

[9] P. Leitner, B. Satzger, W. Hummer, C. Inzinger, and S. Dustdar. Cloudscale: a novel middleware for building transparently scaling cloud applications. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 434–440. ACM, 2012.

[10] J. Miranda, J. Guillén, J. M. Murillo, and C. Canal. Enough about standardization, let's build cloud applications. In *Proceedings of the WICSA/ECSA 2012 Companion Volume*, pages 74–77. ACM, 2012.

[11] L. M. Vaquero, L. Rodero-Merino, and R. Buyya. Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, 41(1):45–52, 2011.

[12] C. Vecchiola, X. Chu, and R. Buyya. Aneka: a software platform for .net-based cloud computing. *High Speed and Large Scale Scientific Computing*, pages 267–295, 2009.