Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 97 (2016) 14 – 23

Procedia Computer Science

CLOUD FORWARD: From Distributed to Complete Computing, CF2016, 18-20 October 2016, Madrid, Spain

Towards the Realization of Multi-dimensional Elasticity for Distributed Cloud Systems

Hong-Linh Truong[a,*], Schahram Dustdar[a], Frank Leymann[b]

[a] *Distributed Systems Group, TU Wien, Austria*
[b] *Institute of Architecture of Application Systems, University of Stuttgart, Germany*

Abstract

As multiple types of distributed, heterogeneous cloud computing environments have proliferated, cloud software can leverage diverse types of infrastructural, platform and data resources with different cost and quality models. This introduces a multi-dimensional elasticity perspective for cloud software that would greatly meet changing demands from the user. However, we argue that current techniques are not enough for dealing with multi-dimensional elasticity in distributed cloud environments. We present our approach to the realization of multi-dimensional elasticity by introducing novel concepts and a roadmap to achieve them.

Peer-review under responsibility of organizing committee of the international conference on cloud forward: From Distributed to Complete Computing

Keywords: distributed clouds; elasticity; cloud services; cloud optimization; monitoring;

1. Introduction

For many application domains, user demands for computing systems are never fixed: even with the same computing infrastructures and software, at different times, our demands can be different due to our constraints on performance, cost, and data quality. For example, such demands are known in an interactive data analysis in the cloud[1] and data analytics of equipment operations in smart cities[2]. Such demands are inherent in distributed, heterogeneous cloud environments, including big, centralized clouds and micro clouds at the edge of the network[3,4,5], where computation, data and network capabilities from these clouds are increasingly virtualized, provisioned and (re)configured as (pay-per-use) utilities in on-demand manners. To deliver tasks in such flexible ways we cannot model resources, costs, and quality in advance (e.g., by means of autonomic knowledge model[6]) due to changing requirements from applications and changing capabilities and (high) availability of distributed cloud resources. That is, depending on the requirements of resources, costs and quality in particular contexts, such as taking more data into the analysis to produce a higher accurate result, we control and (re)configure software systems to meet the demands. Eventually, these systems

[*] Corresponding author
E-mail address: truong@dsg.tuwien.ac.at, dustdar@dsg.tuwien.ac.at, Frank.Leymann@iaas.uni-stuttgart.de

Peer-review under responsibility of organizing committee of the international conference on cloud forward: From Distributed to Complete Computing
doi:10.1016/j.procs.2016.08.276

can return to their previous (normal) behaviors, if such demands are no longer needed, or can move to another behavior, which cannot be modeled at design time, nevertheless, can serve the best trade-offs of resources, costs, and quality under the considered context. The elastic behaviors of such systems must be treated from a multi-dimensional perspective, instead of just resources or costs, when they rely on resources from multiple types of cloud systems.

Researchers realize the paramount importance of being able to formalize, monitor and analyze such multi-dimensional elasticity behaviors to fully exploit elasticity features in cloud computing[7,8]. As such dynamic demands are hard to be captured in advance, in our opinion, existing models and techniques, such as autonomic computing and scalability techniques[6,9,10,11], are not adequate. New methods for elasticity are needed to provide extensible, rigorous ways to enable the development of elasticity adjustment mechanisms to control resources, costs, and quality to quickly deliver the right expectation to the current experienced demands. Such multi-dimensional elasticity adjustment, control and delivery on the basis of resources, costs and quality are fundamental and crucial and have not been addressed so far.

Toward a "complete computing", this paper aims at highlighting a novel approach on tackling multi-dimensional elasticity behavior that enables the realization of elastic distributed cloud software in multi-cloud environments. The rest of this paper is organized as follows: Section 2 presents multi-dimensional elasticity concepts. Section 3 examines some current effort on support multi-dimensional elasticity in distributed clouds. Section 4 presents how to realize these concepts. We conclude the paper and outline our future work in Section 5.

2. Multi-dimensional elasticity in distributed cloud computing systems

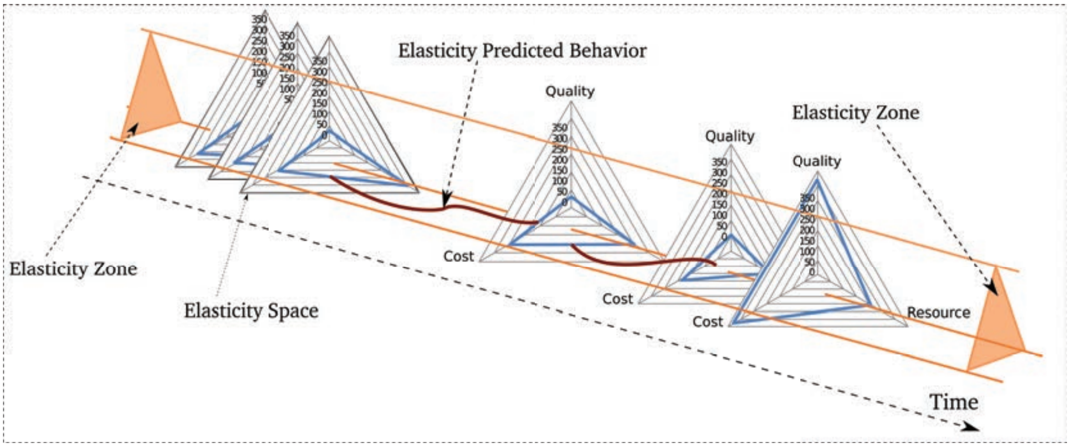

Fig. 1. Elasticity Space, Elasticity Zone, and Elasticity Predicted Behavior

Elasticity as a multi-dimensional perspective could be divided into `Resources`, `Costs`, and `Quality` dimensions[12]. Each dimension can be further divided into sub dimensions. For example, `Quality` can cover performance (response time, availability, etc.) and quality of data (e.g., accuracy and completeness). `Resources` could be virtual machines in data centers, lightweight IoT gateways, or data from sensors or data marketplaces.

Which are new concepts for multi-dimensional elasticity?: Consider a multi-cloud software system[1] whose elasticity is being investigated. Through the time of the system execution, shown in Figure 1, we propose the following important concepts that must be supported for multi-dimensional elasticity realization:

- *Elasticity Zone*: is the required elasticity of a cloud software system that is described by specifying acceptable values for resources, costs, and quality. In distributed clouds where resources are mainly virtual machines/computing resources whose functionality is not really changing, it is highly expected that Elasticity Zones would be based on costs and quality, as resources can be derived from costs and quality. However, when

[1] In this paper, we use "cloud software systems" in a loosely meaning: it can represent cloud applications, SaaS, PaaS, IaaS or a combination of these types.

applications require other types of resources whose functionality is changing, such as continuous streaming data resources, Elasticity Zones could specify a combination of resources (e.g., sources of data) and costs and quality applied for these resources. In Figure 1, we illustrate the Elasticity Zone as a three illustrative dimensions: Resources , Costs, and Quality. This concept is novel as it enables us to specify changing demands based on a multiple dimensions of event-based constraints.

- *Elasticity Space*: consists of the concrete values of elastic properties that a cloud software system displays over time. Elasticity Space is evolved over the time, e.g., as a consequence of internal system operations or external forces. Elasticity Space is novel as it allows us to dynamically capture elasticity behaviors based on specific constraints/demands that enable various types of operations to analyze, predict, and control the elasticity of software systems.
- *Elasticity Prediction Function*: predicts the behavior in the future, based on the elasticity behavior in the past. It also signifies when we need to manipulate the elasticity behavior. As demands are changing and multi-dimensional, Elasticity Prediction Function provides novel concepts for evaluating elasticity behavior in order to enable proactive adjustment of resources, costs, and quality to meet the expected Elasticity Zone.
- *Elasticity Adjustment Function*: alters the system's elastic capabilities; another word to change the current Elasticity Space. As an example, an Elasticity Space changes over time and could leave its associated Elasticity Zone. Thus, the system would not operate within the specified zone anymore. Elasticity Adjustment Function can compute the necessary adjustments to the system in order to keep the Elasticity Space in the Elasticity Zone. Furthermore, the Elasticity Adjustment Function may react to changes in the underlying computing environment, for example, the costs of resources could change. The concept of Elasticity Adjustment Function is powerful and fundamental for elasticity because it presents a formal, general model to describe any kind of control features as pluggable and extensible modules for different underlying cloud systems.

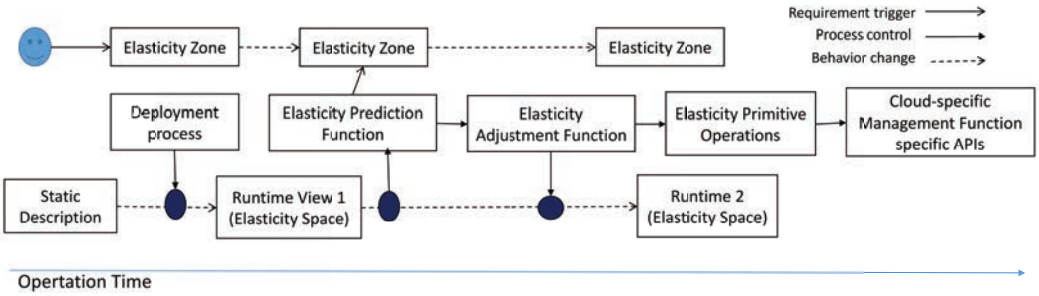
Requirement trigger
Process control
Behavior change
Elasticity Zone
Elasticity Zone
Elasticity Zone
Deployment process
Elasticity Prediction Function
Elasticity Adjustment Function
Elasticity Primitive Operations
Cloud-specific Management Function specific APIs
Static Description
Runtime View 1 (Elasticity Space)
Runtime 2 (Elasticity Space)
Opertation Time

Fig. 2. Lifecycle of cloud systems and proposed multi-dimensional elasticity concepts

Why are these concepts important?: To explain the importance of these concepts, let us (re)examine the lifecycle of the cloud software systems. Shown in Figure 2, it is easy to see a generic lifecycle of a cloud software system: (i) from a *static structure*, e.g. described in CloudFormation, Heat Orchestration Template (HOT), or TOSCA, we deploy the system using certain deployment processes, and (ii) at runtime, we then monitor and analyze the system – sometime we predict the behavior of the system using *Elasticity Prediction Function* – then we adjust the system through elasticity control rules/algorithms which can be captured by *Elasticity Adjustment Function*. In parallel, all actions we perform are due to the change of requirements (reflected in *Elasticity Zone*) and such a change of requirements might be triggered by developers/providers/users or automatic software (e.g., prediction suggests to change zone). While we can see these basic steps in many cloud middleware, we lack formal models, methods and tools to capture and engineer these concepts to enable elasticity management, coordination and interoperability in multi-cloud environments. For examples, Adjustment Functions will need to rely on low-level *Management Functions* from specific cloud systems in order to manipulate resources, costs, and quality (e.g., on-the-fly using software-defined mechanisms). This would be very complex given the sheer number of resources and the heterogeneity of distributed cloud systems. Introducing *"Elasticity Primitive Operations"* as basic actions of Adjustment Functions would simplify a lot of the development and engineering of multi-cloud elasticity.

Table 1. Multi-dimensional elasticity support in some EU projects. When we do not have enough information for analyzing a feature, we let the evaluation of the feature blank

Project	Multi-Cloud Elasticity	Edge and Cloud Elasticity	Elasticity Zone	Elasticity Space	Elasticity Prediction	Elasticity Adjustment
CELAR[13]	**partially**. It does not have a coordination among elasticity in different clouds	**partially**. It enables coordinated elasticity between IoT resources and cloud services[14]	**partially**. It support only static zones through constraints without time-dependent function, although zones can be changed manually during runtime	**partially**. It partially builds elasticity spaces by interfacing to several cloud monitoring tools	**no**	**yes**. Its elasticity strategies and elasticity primitive operations are mapped to underlying operations, such as scale in/out
HARNESS[15]	**no**	**no**	**partially**. It uses static thresholds for SLO defined from typical performance metrics	**no**	**partially**. It supports prediction for web applications using existing workload and statistical models[16] and for job-level objectives with jobs using reconfigurable accelerators[17]	**yes**. It supports scale out/back of VM/machine resources
MODAClouds[18]	**yes**. It supports cloud bursting and centralized auto-scaling controller[19]	**no**	**partially**. It supports thresholds defined based on typical performance metrics.		**partially**. It *estimates* cost and performance at the design time[20]	**yes**. It uses Models@runtime techniques
PaaSage[21]	**yes**. soCloud[22] supports a centralized coordinator for elasticity within individual clouds	**no**	**partially**. It supports zones through constraints of typical performance metrics		**no**	**yes**. SRL[23] supports elasticity adjustment, mainly for VM resources.

3. Current Effort in Multi-dimensional Elasticity in Distributed Clouds

In this section, we analyze the current effort from different EU research projects on support multi-dimensional elasticity based on our concepts defined in Section 2. Table 2 outlines some state-of-the-art *multi-dimensional elasticity* support in several EU projects. The analysis is based on the publications available from these projects. Although several of these project support multi-cloud deployments, little support is for *multi-dimensional elasticity* across different clouds, e.g., in the sense of coordination-aware elasticity[2]. Clearly, most projects support elasticity in single clouds; then they extend single cloud elasticity for distributed clouds. Thus, features for elasticity adjustment are mostly supported for scaling in/out actions for VM/machine resources and elasticity zones are simply pre-defined constraints for thresholds of traditional metrics, like response time and resource cost. Projects, like MODAClouds and PaaSage, focus on multi-cloud applications, thus they introduce basic multi-cloud elasticity using centralized coordination models. Important concepts in our view, such as Elasticity Zone, Elasticity Space and Elasticity Prediction, have not really been investigated for elasticity in distributed clouds, let alone for distributed edge and cloud systems.

4. Realizing Multi-dimensional Elasticity

4.1. Elasticity space and zone

4.1.1. Elasticity Zones

Limitations of current approaches: Although constraints on elasticity can be specified through rules, such as in[24], a formal model of Elasticity Zones has been never proposed. The current way of using rules/policies fails to capture the dynamics of Elasticity Zones w.r.t the time associated with and the properties specified within elasticity requirements. The time in which a rule would be applied is either not specified (reactive) or static (e.g., with a specific time slot in Amazon's scheduled auto-scaling[2]). Another issue is that these rules have not covered resources in multi-cloud environments, e.g. data and computing resources from the edge[4,5,25]. Furthermore, we tend to think that Elasticity Zones, as constraints for elasticity, should be defined by the user (developers, operators and providers). However, Elasticity Zones are not just defined by the end user or the provider (by means of rules) but they can be generated automatically by software, which, for example, generate elasticity zones based on input parameters and historical elasticity information.

Approach to multi-dimensional elasticity realization: First, the characteristics of elasticity properties including the value ranges need to be studied. Currently, the main elasticity properties are the number of resources and the prices to be paid; not to mention that elasticity properties based on CPU and memory are too low-level. However, they are not enough, as we need to consider properties about data resources as well as, e.g., sensors and actuators, which are popular in distributed micro clouds[25,5]. Once we define elasticity properties in terms of attributes specified via their mathematical domains and ranges, we can model the formal model for Elasticity Zones by an n-dimensional manifold of which a dimension of an appropriate combination of elasticity properties is event-dependent. Although Elasticity Zones are described through requirements, zones are not static but are associated with events which determine the zones. Examples of such events are a human in the control loop, a prediction function, or simply a pre-defined time trigger. Thus, while the elasticity properties in an Elasticity Zone might not be changed, the values of these properties might be changed.

4.1.2. Elasticity Space

Limitations of current approaches: One can see that Elasticity Space is no more than a set of multi-dimensional time series datapoints, each point representing a set of metrics. Thus, it can be captured by existing monitoring systems[26,27]. However, the concept of Elasticity Space has an important characteristic: due to changes in the requirement, the number of dimensions (or metrics) is not fixed at runtime. Furthermore, the frequency of metric measurement is also not fixed. For example, given a CPU usage as a metric in the space, we could measure it every second or every 5 seconds, depending specific situations. A consequence is that cloud monitoring systems should not monitor a fixed

[2] `http://docs.aws.amazon.com/autoscaling/latest/userguide/schedule_time.html`

set of metrics, which may either not support enough monitoring data for elasticity analysis and control, or may introduce too much monitoring overhead for producing unneeded metrics. Able to support adaptive instrumentation and monitoring of elasticity properties requires a major change in how to engineer cloud monitoring, especially for micro clouds deployed at the edge[3].

Approach to multi-dimensional elasticity realization: An appropriate formal representation of Elasticity Spaces will correlate concrete elasticity property values and support changes of property values over time. First, we focus on *algorithmic models for Elasticity Space function* for determining Elasticity Space based on their defined Elasticity Zones. We need to provide generic functions for determining and analyzing Elasticity Space, including detecting when a software starts being elastic and ends, and why (under which forces and requirements). We then develop a comprehensive set of Elasticity Space detection function implementations for different common software patterns (e.g., cloud-based patterns). Second, given an Elasticity Space, we need to also develop *operators on Elasticity Spaces*. For example, Elasticity Spaces characterizing different software components can be expected to be merged by a particular operator to create a new Elasticity Space to represent a composition of these components. Such operators will be useful for implementing adaptive elasticity monitoring through monitoring data integration and analysis.

4.2. Elasticity analysis and prediction

Limitations of current approaches: We can categorize elasticity analysis in the cloud into: (i) frameworks which monitor cloud systems, (ii) monitoring frameworks which are elastic, and (iii) monitoring frameworks for elastic properties. However, many of them are not much different from traditional performance analysis in typical distributed systems preceding cloud computing - albeit supporting new metrics and running in the cloud. For example, a scalable framework for data collection and aggregation is introduced in[28]. The cost of Amazon EC2 spot instances is analyzed in[29], and[30] discusses cost-effective strategies for using such instances. Sharma et al.[31] use prediction to support cost-aware resource provisioning. The work in[32] improves existing monitoring systems using custom metric aggregation scripts and service model information. A mechanism for adapting cloud allocation is presented in[11], using an aggregator that monitors the workload at each service tier. In[27] cloud resource usage is monitored. The works in[33] and[34] present comprehensive monitoring systems collecting both virtual infrastructure and service level information. Monitoring performance and data delivery are also the focus of[35], tailored for a specific virtualization framework. An elastic monitoring framework for cloud infrastructures is presented in[26].

Predicting elasticity works, such as[36,37], are still very much focusing on a single dimension. Such analysis is not enough. We need to focus on elasticity relationships. Works like in[38] are just an initial step because they focus on building the platform, while definitions and algorithms for understanding complex elasticity relationships among components are still open.

Approach to multi-dimensional elasticity realization: First, one important class of algorithms is *for elasticity dependency analysis* which can help us to understand novel metrics characterizing the elasticity dependencies among different components of an elastic system. These functions address several important questions, such as which part of the software system can be elasticized and under which context? We have learned from the Amdahl law that certain parts of an application cannot be parallelized and it does not make sense to parallelize these parts if we would like to achieve a higher speedup for the application. Is a similar point valid for elasticity? Which parts of an cloud system should be in the focus of the elasticity control, and why? Answering these question also helps us to determine where we could focus the elasticity controls to meet elasticity requirements. Hence, the dependencies among different parts of cloud systems w.r.t. elasticity capabilities must be detected.

Second, we focus on *algorithms for elasticity prediction function*. Formally, a prediction function epf takes two parameters: ez as an Elasticity Zone and es as an Elasticity Space. Then the function $epf(ez, es)$ produces a predicted Elasticity Space pes_o, $eps(ez, es) \rightarrow pes_o$ which illustrates the predicted elasticity behavior. Since prediction is a complex matter, we focus on developing a comprehensive set of elasticity prediction functions for common software patterns[39] and common elasticity views (e.g., cost-centric, quality-centric, single cloud site, cross clouds, etc.). The rationale is that these patterns and views are building blocks of complex distributed cloud software; having such functions for them would make substantial impact on programming elasticity in cloud software.

4.3. Elasticity Patterns

Limitations of current approaches: There exist performance and scalability best practices and patterns in cloud[40,41] but not really elasticity patterns, although we see cloud patterns discussing elastic components[39]. For elasticity adjustment functions to change the elasticity behavior, underlying platforms on which an elastic system is running has to supply management functions which, e.g., can adjust the resources a software uses, the performance of a database, and the throughput of underlying network function. We have seen such functions in different types of clouds and cloud network systems, but there is a lack of unified way to capture and represent them. Furthermore, management functions for distributed micro clouds at the edge have not been studied together with centralized clouds to provide a uniform view on cross distributed cloud elasticity. Therefore, it will be very hard for programming elasticity features across multiple clouds.

Approach to multi-dimensional elasticity realization: To overcome specific set of management functions for specific platforms, we need to work on a *common model for Elasticity Primitive Operations* (see Figure 2) that will be offered via API. We need well-defined APIs for elasticity primitive operations. The tricky part is to have a common view on multi-dimensional elasticity w.r.t. resources, cost, and quality so that such primitive operations will be supported across cloud environments, including clouds at the edge, using common notations. We can collect existing solutions and patterns into a knowledge base that can be queried to identify common software patterns associated with elasticity behaviors learned from, e.g., machine learning techniques[42]. In parallel, developers and providers can develop different elasticity adjustment functions for different patterns in their systems. Several benchmarks and profiling configurations would be then executed to collect elasticity behaviors, and machine learning is used to understand elasticity. In the knowledge base a set of Elasticity Primitive Operations will specify elasticity capabilities for individual components, topologies of components, or the whole system. Both user-defined and machine learning mechanisms are supported for establishing elasticity capabilities.

Another important issue is to *deduce Elasticity Patterns*. Based on the information collected, existing elasticity solutions and patterns are analyzed to deduct new elasticity patterns from the document solutions and to extend the existing elasticity patterns. This also deals with composite elasticity patterns which are built from multiple of these fundamental patterns in a coordinated fashion.

4.4. Elasticity Adjustment Function

Limitations of current approaches: One of the most active research tracks in elasticity is for elasticity control or adjustment. Yang et al.[9] support just-in-time scalability of resources based on profiles. Kazhamiakin et al.[43] consider KPI dependencies when adapting the service based applications. PRESS[44] and CloudScale[45] are examples of application resource elasticity frameworks which use prediction for reducing the number of over and under estimation errors. Malkowski et al.[46] support multi-level modeling and elastic control of resources for workflows in the cloud. Guinea et al.[47] develop a system for multi-level monitoring and adaptation of service-based systems. Different research works have focused on elasticity control of storage resources and quality, e.g., for deciding how many database nodes are needed[48]. Wang et al.[49] propose an algorithm for software resource allocation considering the loads, analyzing the influence of software resources management on the applications performance. Yu et al.[10] propose an approach for resource management of elastic cloud workflows. Kranas et al.[50] introduce elasticity as a service cross-cutting different cloud stack layers (SaaS, PaaS, IaaS). An intensive survey of existing elasticity control techniques by Lorido-Botran et al.[51] shows that a majority of tools support reactive auto-scaling of resources based on basic metrics, like CPU and memory.

While we can use such specific models for adjusting specific aspects in multi-dimensional elasticity, there is no theoretical foundation on how elasticity adjustment functions should look like by combining these models for multi-dimensional elasticity realization. Moreover, adjusting resources, costs and quality in micro clouds is not in the focus and is not coordinated with adjusting centralized cloud resources.

Approach to multi-dimensional elasticity realization: The Elasticity Adjustment Function controls elastic software to ensure that their Elasticity Space should move within the expected Elasticity Zone. First, this will have to deal with different scales: individual components, topologies of components, and the entire software systems. Second, such adjustment functions can be developed for single clouds or multiple clouds. We develop such functions based on the following steps:

Step 1: Elasticity Pattern Selection: Elasticity behaviors will be used by the Elasticity Adjustment Function to determine the right elasticity control actions (and configurations). Thus, we search the right patterns that should be used to control the elastic systems.

Step 2: Select primitive operations: The cause and effect of primitive operations must be determined in order to select the right actions for the adjustment. In the knowledge base, we must parameterize primitive operations to estimate their effect, as it is likely that effect information in the knowledge base is not enough.

Step 3: Selecting and generating Elasticity Adjustment Function: We envision that each Elasticity Adjustment Function must be defined in particular views (e.g., topology or deployment sites) of a cloud system. Hence, we model an Elasticity Adjustment Function as a workflow of Elasticity Primitive Operations with concrete parameters suitable for the given context. When information about the topology and patterns are clear, we might pickup an existing adjustment function to apply, e.g., based on the evaluation of runtime behaviors[42]. However, different techniques and AI planning methods would be useful for the generation or parameterization of a new or existing Elasticity Adjustment Function.

Our initial effort is to investigate how to reuse some preliminary AI works for certain patterns, like rollback of basic operations for virtual machines and disk volumes[52] and case-based automatic adaptation of workflows[53]. However, we do not believe that automatic generation will work on all cases, e.g., for interactive elasticity adjustment. Therefore, humans are needed to augment the generation process or solving the conflicts- through the so-called human-in-the-loop management in elasticity operation management[54].

5. Conclusions and Future Work

To exploit elasticity capabilities of multiple distributed cloud resources spanning through centralized data centers and edge devices, we need to focus on novel concepts supporting multi-dimensional elasticity. We explain why more formal concepts and rigorous development are needed to support multi-dimensional elasticity. From a detailed analysis of the state of the art, we show our approach to the realization of multi-dimensional elasticity by integrating various concepts into the lifecylce of cloud software systems. Currently, we are extending our iCOMOT prototype[3] with these concepts. iCOMOT includes several tools for configuring, controlling, and monitoring cloud services and software components at the edge (e.g., sensors and IoT gateways). We are testing our concepts atop distributed, heterogeneous cloud computing environments.

Acknowledgments

We thank Johanna Barzen. Georgiana Copil, Christoph Fehling and Daniel Moldovan for their fruitful discussion. Especially we also thank Daniel Moldovan for creating Figure 1.

References

1. Madden, S.. Interactive data analytics: the new frontier. http://acmsocc.github.io/2015/keynotes/socc15-keynote.pdf; 2015. SOCC 15 Keynote.
2. Truong, H.L., Dustdar, S.. Programming elasticity in the cloud. *IEEE Computer* 2015;**48**(3):87–90. URL: `http://dx.doi.org/10.1109/MC.2015.84`. doi:`10.1109/MC.2015.84`.
3. Valancius, V., Laoutaris, N., Massoulié, L., Diot, C., Rodriguez, P.. Greening the internet with nano data centers. In: *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*; CoNEXT '09. New York, NY, USA: ACM. ISBN 978-1-60558-636-6; 2009, p. 37–48. URL: `http://doi.acm.org/10.1145/1658939.1658944`. doi:`10.1145/1658939.1658944`.
4. Satyanarayanan, M.. Cloudlets: At the leading edge of cloud-mobile convergence. In: *Proceedings of the 9th International ACM Sigsoft Conference on Quality of Software Architectures*; QoSA '13. New York, NY, USA: ACM. ISBN 978-1-4503-2126-6; 2013, p. 1–2. URL: `http://doi.acm.org/10.1145/2465478.2465494`. doi:`10.1145/2465478.2465494`.
5. Garcia Lopez, P., Montresor, A., Epema, D., Datta, A., Higashino, T., Iamnitchi, A., et al. Edge-centric computing: Vision and challenges. *SIGCOMM Comput Commun Rev* 2015;**45**(5):37–42. URL: `http://doi.acm.org/10.1145/2831347.2831354`. doi:`10.1145/2831347.2831354`.

[3] `http://tuwiendsg.github.io/iCOMOT/`

6. Huebscher, M.C., McCann, J.A.. A survey of autonomic computing&mdash;degrees, models, and applications. *ACM Comput Surv* 2008; **40**(3):7:1–7:28. URL: `http://doi.acm.org/10.1145/1380584.1380585`. doi:`10.1145/1380584.1380585`.
7. Galante, G., Bona, L.C.E.d.. A survey on cloud computing elasticity. In: *Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing*; UCC '12. Washington, DC, USA: IEEE Computer Society. ISBN 978-0-7695-4862-3; 2012, p. 263–270. URL: `http://dx.doi.org/10.1109/UCC.2012.30`. doi:`10.1109/UCC.2012.30`.
8. Barker, A., Varghese, B., Ward, J.S., Sommerville, I.. Academic cloud computing research: Five pitfalls and five opportunities. In: Kozuch, M.A., Yu, M., editors. *6th USENIX Workshop on Hot Topics in Cloud Computing, HotCloud '14, Philadelphia, PA, USA, June 17-18, 2014*. USENIX Association; 2014, URL: `https://www.usenix.org/conference/hotcloud14/workshop-program/presentation/barker`.
9. Yang, J., Qiu, J., Li, Y.. A profile-based approach to just-in-time scalability for cloud applications. In: *IEEE International Conference on Cloud Computing, CLOUD 2009, Bangalore, India, 21-25 September, 2009*. IEEE. ISBN 978-1-4244-5199-9; 2009, p. 9–16. URL: `http://dx.doi.org/10.1109/CLOUD.2009.87`. doi:`10.1109/CLOUD.2009.87`.
10. Yu, L., Thain, D.. Resource management for elastic cloud workflows. In: *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*. 2012, p. 775–780. doi:`10.1109/CCGrid.2012.107`.
11. Singh, R., Sharma, U., Cecchet, E., Shenoy, P.. Autonomic mix-aware provisioning for non-stationary data center workloads. In: *Proceedings of the 7th International Conference on Autonomic Computing*; ICAC '10. New York, NY, USA: ACM. ISBN 978-1-4503-0074-2; 2010, p. 21–30. URL: `http://doi.acm.org/10.1145/1809049.1809053`. doi:`10.1145/1809049.1809053`.
12. Dustdar, S., Guo, Y., Satzger, B., Truong, H.L.. Principles of elastic processes. *IEEE Internet Computing* 2011;**15**(5):66–71. URL: `http://doi.ieeecomputersociety.org/10.1109/MIC.2011.121`. doi:`10.1109/MIC.2011.121`.
13. Celar - cloud elasticity provisioning. `http://www.celarcloud.eu/`; 2016. Last access: 23 July 2016.
14. Copil, G., Moldovan, D., Truong, H.L., Dustdar, S.. On controlling cloud services elasticity in heterogeneous clouds. In: *Proceedings of the 7th IEEE/ACM International Conference on Utility and Cloud Computing, UCC 2014, London, United Kingdom, December 8-11, 2014*. IEEE Computer Society. ISBN 978-1-4799-7881-6; 2014, p. 573–578. URL: `http://dx.doi.org/10.1109/UCC.2014.88`. doi:`10.1109/UCC.2014.88`.
15. Harness. `http://www.harness-project.eu`; 2016. Last access: 23 July 2016.
16. Fernandez, H., Pierre, G., Kielmann, T.. Autoscaling web applications in heterogeneous cloud infrastructures. In: *2014 IEEE International Conference on Cloud Engineering, Boston, MA, USA, March 11-14, 2014*. IEEE. ISBN 978-1-4799-3766-0; 2014, p. 195–204. URL: `http://dx.doi.org/10.1109/IC2E.2014.25`. doi:`10.1109/IC2E.2014.25`.
17. Grigoras, P., Tottenham, M., Niu, X., Coutinho, J.G.F., Luk, W.. Elastic management of reconfigurable accelerators. In: *2014 IEEE International Symposium on Parallel and Distributed Processing with Applications*. 2014, p. 174–181. doi:`10.1109/ISPA.2014.31`.
18. Modaclouds – model-driven approach for design and execution of applications on multiple clouds. `http://www.modaclouds.eu/`; 2016. Last access: 23 July 2016.
19. Modaclouds: Runtime environment final release. `http://www.modaclouds.eu/wp-content/uploads/2012/09/MODAClouds_D6.5.3_RunTimeEnvironmentFinalRelease.pdf`; 2012.
20. Franceschelli, D., Ardagna, D., Ciavotta, M., Di Nitto, E.. Space4cloud: A tool for system performance and costevaluation of cloud systems. In: *Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds*; MultiCloud '13. New York, NY, USA: ACM. ISBN 978-1-4503-2050-4; 2013, p. 27–34. URL: `http://doi.acm.org/10.1145/2462326.2462333`. doi:`10.1145/2462326.2462333`.
21. Paasage. `http://www.paasage.eu/`; 2016. Last access: 23 July 2016.
22. Paraiso, F., Merle, P., Seinturier, L.. socloud: a service-oriented component-based paas for managing portability, provisioning, elasticity, and high availability across multiple clouds. *Computing* 2016;**98**(5):539–565. URL: `http://dx.doi.org/10.1007/s00607-014-0421-x`. doi:`10.1007/s00607-014-0421-x`.
23. Kritikos, K., Domaschka, J., Rossini, A.. SRL: A scalability rule language for multi-cloud environments. In:[55]; 2014, p. 1–9. URL: `http://dx.doi.org/10.1109/CloudCom.2014.170`. doi:`10.1109/CloudCom.2014.170`.
24. Jamshidi, P., Ahmad, A., Pahl, C.. Autonomic resource provisioning for cloud-based software. In: *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*; SEAMS 2014. New York, NY, USA: ACM. ISBN 978-1-4503-2864-7; 2014, p. 95–104. URL: `http://doi.acm.org/10.1145/2593929.2593940`. doi:`10.1145/2593929.2593940`.
25. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.. Fog computing and its role in the internet of things. In: *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*; MCC '12. New York, NY, USA: ACM. ISBN 978-1-4503-1519-7; 2012, p. 13–16. URL: `http://doi.acm.org/10.1145/2342509.2342513`. doi:`10.1145/2342509.2342513`.
26. König, B., Calero, J.M.A., Kirschnick, J.. Elastic monitoring framework for cloud infrastructures. *IET Communications* 2012;**6**(10):1306–1315. URL: `http://dx.doi.org/10.1049/iet-com.2011.0200`. doi:`10.1049/iet-com.2011.0200`.
27. Dhingra, M., Lakshmi, J., Nandy, S.K.. Resource usage monitoring in clouds. In: *Proceedings of the 2012 ACM/IEEE 13th International Conference on Grid Computing*; GRID '12. Washington, DC, USA: IEEE Computer Society. ISBN 978-0-7695-4815-9; 2012, p. 184–191. URL: `http://dx.doi.org/10.1109/Grid.2012.10`. doi:`10.1109/Grid.2012.10`.
28. Wang, C., Schwan, K., Talwar, V., Eisenhauer, G., Hu, L., Wolf, M.. A flexible architecture integrating monitoring and analytics for managing large-scale data centers. In: *Proceedings of the 8th ACM International Conference on Autonomic Computing*; ICAC '11. New York, NY, USA: ACM. ISBN 978-1-4503-0607-2; 2011, p. 141–150. URL: `http://doi.acm.org/10.1145/1998582.1998605`. doi:`10.1145/1998582.1998605`.
29. Ben-Yehuda, O.A., Ben-Yehuda, M., Schuster, A., Tsafrir, D.. Deconstructing amazon ec2 spot instance pricing. In: *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*. 2011, p. 304–311. doi:`10.1109/CloudCom.2011.48`.
30. Yi, S., Kondo, D., Andrzejak, A.. Reducing costs of spot instances via checkpointing in the amazon elastic compute cloud. In: *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*. 2010, p. 236–243. doi:`10.1109/CLOUD.2010.35`.
31. Sharma, U., Shenoy, P., Sahu, S., Shaikh, A.. A cost-aware elasticity provisioning system for the cloud. In: *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*. 2011, p. 559–570. doi:`10.1109/ICDCS.2011.59`.

32. Shao, J., Wei, H., Wang, Q., Mei, H.. A runtime model based monitoring approach for cloud. In: *IEEE International Conference on Cloud Computing, CLOUD 2010, Miami, FL, USA, 5-10 July, 2010*. IEEE. ISBN 978-1-4244-8207-8; 2010, p. 313–320. URL: http://dx.doi.org/10.1109/CLOUD.2010.31. doi:10.1109/CLOUD.2010.31.
33. Katsaros, G., Kousiouris, G., Gogouvitis, S.V., Kyriazis, D., Menychtas, A., Varvarigou, T.. A self-adaptive hierarchical monitoring mechanism for clouds. *Journal of Systems and Software* 2012;**85**(5):1029 – 1041. URL: http://www.sciencedirect.com/science/article/pii/S0164121211002998. doi:http://dx.doi.org/10.1016/j.jss.2011.11.1043.
34. Clayman, S., Galis, A., Chapman, C., Toffetti, G., Rodero-Merino, L., Vaquero, L.M., et al. Monitoring Service Clouds in the Future Internet. In: *Towards the Future Internet - Emerging Trends from European Research*. IOS Press; 2010, .
35. Kutare, M., Eisenhauer, G., Wang, C., Schwan, K., Talwar, V., Wolf, M.. Monalytics: Online monitoring and analytics for managing large scale data centers. In: *Proceedings of the 7th International Conference on Autonomic Computing*; ICAC '10. New York, NY, USA: ACM. ISBN 978-1-4503-0074-2; 2010, p. 141–150. URL: http://doi.acm.org/10.1145/1809049.1809073. doi:10.1145/1809049.1809073.
36. Loff, J., Garcia, J.. Vadara: Predictive elasticity for cloud applications. In: *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*. 2014, p. 541–546. doi:10.1109/CloudCom.2014.161.
37. Roy, N., Dubey, A., Gokhale, A.S.. Efficient autoscaling in the cloud using predictive models for workload forecasting. In: Liu, L., Parashar, M., editors. *IEEE International Conference on Cloud Computing, CLOUD 2011, Washington, DC, USA, 4-9 July, 2011*. IEEE. ISBN 978-1-4577-0836-7; 2011, p. 500–507. URL: http://dx.doi.org/10.1109/CLOUD.2011.42. doi:10.1109/CLOUD.2011.42.
38. Moldovan, D., Copil, G., Truong, H.L., Dustdar, S.. On analyzing elasticity relationships of cloud services. In: [55]; 2014, p. 447–454. URL: http://dx.doi.org/10.1109/CloudCom.2014.93. doi:10.1109/CloudCom.2014.93.
39. Fehling, C., Leymann, F., Retter, R., Schupeck, W., Arbitter, P.. *Cloud Computing Patterns - Fundamentals to Design, Build, and Manage Cloud Applications*. Springer; 2014. ISBN 978-3-7091-1567-1. URL: http://dx.doi.org/10.1007/978-3-7091-1568-8. doi:10.1007/978-3-7091-1568-8.
40. Gesvindr, D., Buhnova, B.. Performance challenges, current bad practices, and hints in paas cloud application design. *SIGMETRICS Perform Eval Rev* 2016;**43**(4):3–12. URL: http://doi.acm.org/10.1145/2897356.2897358. doi:10.1145/2897356.2897358.
41. Cukier, D.. Devops patterns to scale web applications using cloud services. In: *Proceedings of the 2013 Companion Publication for Conference on Systems, Programming, Languages and Applications: Software for Humanity*; SPLASH '13. New York, NY, USA: ACM. ISBN 978-1-4503-1995-9; 2013, p. 143–152. URL: http://doi.acm.org/10.1145/2508075.2508432. doi:10.1145/2508075.2508432.
42. Copil, G., Truong, H.L., Moldovan, D., Dustdar, S., Trihinas, D., Pallis, G., et al. Evaluating cloud service elasticity behavior. *Int J Cooperative Inf Syst* 2015;**24**(3). URL: http://dx.doi.org/10.1142/S0218843015410026. doi:10.1142/S0218843015410026.
43. Kazhamiakin, R., Wetzstein, B., Karastoyanova, D., Pistore, M., Leymann, F.. Adaptation of service-based applications based on process quality factor analysis. In: *Proceedings of the 2009 International Conference on Service-oriented Computing*; ICSOC/ServiceWave'09. Berlin, Heidelberg: Springer-Verlag. ISBN 3-642-16131-6, 978-3-642-16131-5; 2009, p. 395–404. URL: http://dl.acm.org/citation.cfm?id=1926618.1926660.
44. Gong, Z., Gu, X., Wilkes, J.. Press: Predictive elastic resource scaling for cloud systems. In: *Network and Service Management (CNSM), 2010 International Conference on*. 2010, p. 9–16. doi:10.1109/CNSM.2010.5691343.
45. Shen, Z., Subbiah, S., Gu, X., Wilkes, J.. Cloudscale: Elastic resource scaling for multi-tenant cloud systems. In: *Proceedings of the 2Nd ACM Symposium on Cloud Computing*; SOCC '11. New York, NY, USA: ACM. ISBN 978-1-4503-0976-9; 2011, p. 5:1–5:14. URL: http://doi.acm.org/10.1145/2038916.2038921. doi:10.1145/2038916.2038921.
46. Malkowski, S.J., Hedwig, M., Li, J., Pu, C., Neumann, D.. Automated control for elastic n-tier workloads based on empirical modeling. In: *Proceedings of the 8th ACM International Conference on Autonomic Computing*; ICAC '11. New York, NY, USA: ACM. ISBN 978-1-4503-0607-2; 2011, p. 131–140. URL: http://doi.acm.org/10.1145/1998582.1998604. doi:10.1145/1998582.1998604.
47. Guinea, S., Kecskemeti, G., Marconi, A., Wetzstein, B.. Multi-layered monitoring and adaptation. In: *Proceedings of the 9th International Conference on Service-Oriented Computing*; ICSOC'11. Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-642-25534-2; 2011, p. 359–373.
48. Tsoumakos, D., Konstantinou, I., Boumpouka, C., Sioutas, S., Koziris, N.. Automated, elastic resource provisioning for nosql clusters using tiramola. In: *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*. 2013, p. 34–41. doi:10.1109/CCGrid.2013.45.
49. Wang, Q., Malkowski, S., Kanemasa, Y., Jayasinghe, D., Xiong, P., Pu, C., et al. The impact of soft resource allocation on n-tier application scalability. In: *Parallel Distributed Processing Symposium (IPDPS), 2011 IEEE International*. 2011, p. 1034–1045. doi:10.1109/IPDPS.2011.99.
50. Kranas, P., Anagnostopoulos, V., Menychtas, A., Varvarigou, T.. Elaas: An innovative elasticity as a service framework for dynamic management across the cloud stack layers. In: *Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on*. 2012, p. 1042–1049. doi:10.1109/CISIS.2012.117.
51. Lorido-Botran, T., Miguel-Alonso, J., Lozano, J.A.. A review of auto-scaling techniques for elastic applications in cloud environments. *J Grid Comput* 2014;**12**(4):559–592. URL: http://dx.doi.org/10.1007/s10723-014-9314-7. doi:10.1007/s10723-014-9314-7.
52. Weber, I., Wada, H., Fekete, A., Liu, A., Bass, L.. Automatic undo for cloud management via ai planning. In: *Proceedings of the Eighth USENIX Conference on Hot Topics in System Dependability*; HotDep'12. Berkeley, CA, USA: USENIX Association; 2012, p. 10–10. URL: http://dl.acm.org/citation.cfm?id=2387858.2387868.
53. Minor, M., Bergmann, R., Görg, S., Walter, K.. *Towards Case-Based Adaptation of Workflows*. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-642-14274-1; 2010, p. 421–435.
54. Copil, G., Truong, H.L., Dustdar, S.. Supporting cloud service operation management for elasticity. In: Barros, A., Grigori, D., Narendra, N.C., Dam, H.K., editors. *Service-Oriented Computing - 13th International Conference, ICSOC 2015, Goa, India, November 16-19, 2015, Proceedings*; vol. 9435 of *Lecture Notes in Computer Science*. Springer. ISBN 978-3-662-48615-3; 2015, p. 123–138.
55. *IEEE 6th International Conference on Cloud Computing Technology and Science, CloudCom 2014, Singapore, December 15-18, 2014*. IEEE Computer Society; 2014. ISBN 978-1-4799-4093-6. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7031670.