A Multi-Cloud Framework for Measuring and Describing Performance Aspects of Cloud Services Across Different Application Types

G. Kousiouris[1], G. Giammatteo[2], A. Evangelinou[1], N. Galante[2], E. Kevani[1], C. Stampoltas[1], A. Menychtas[1], A. Kopaneli[1], K. Ramasamy Balraj[2], D. Kyriazis[1], T. Varvarigou[1], P. Stuer[3] and L. Orue-Echevarria Arrieta[4]

[1]*Dept. Of Electrical and Computer Engineering, NTUA, 9 Heroon Polytechnioy Str, 15773, Athens, Greece*
[2]*Research and Development Laboratory, Engineering Ingegneria Informatica S.p.A., V. R. Morandi, 32 00148 Roma, Italy*
[3]*Spikes Research dept., Spikes, Mechelsesteenweg 64, B-2018 Antwerpen, Belgium*
[4]*TECNALIA, ICT-European Software Institute Division, Parque Tecnológico Ed #202, E-48170 Zamudio, Spain*
gkousiou@mail.ntua.gr, gabriele.giammatteo@eng.it, aevang@mail.ntua.gr, nunzioandrea.galante@eng.it, eltonkevani@hotmail.com, stampoltaschris@gmail.com,{ameny, alikikop}@mail.ntua.gr, kanchanna.ramasamybalraj@eng.it, dimos@mail.ntua.gr, dora@telecom.ntua.gr, peter.stuer@spikes.be, leire.orue-echevarria@tecnalia.com

Keywords: Benchmarking, Cloud Services, Multi-Cloud, Performance.

Abstract: Cloud services have emerged as an innovative IT provisioning model in the recent years. However, after their usage severe considerations have emerged with regard to their varying performance due to multitenancy and resource sharing issues. These issues make it very difficult to provide any kind of performance estimation during application design or deployment time. The aim of this paper is to present a mechanism and process for measuring the performance of various Cloud services and describing this information in machine understandable format. The framework is responsible for organizing the execution and can support multiple Cloud providers. Furthermore we present approaches for measuring service performance with the usage of specialized metrics for ranking the services according to a weighted combination of cost, performance and workload.

1 INTRODUCTION

Performance of Cloud environments has started to gain significant attention in the recent years (Hauck, 2010). After promises for infinite resources and on-demand scalability, the issues of Cloud environments instability with regard to performance issues of the allocated resources have begun to arise (Kousiouris et al., 2012). Thus, for a successful Cloud migration process, the issue of provider performance should be taken very seriously, in order to save money but also guarantee a (as much as possible) stability in the migrated application. As identified in our previous work (ARTIST Consortium D7.2, 2013) a significant gap in existing research is the lack of such descriptions in current metamodels regarding Cloud infrastructures. However, different providers may have their own metrics and strategies for guaranteeing Cloud QoS. Thus a more abstracted and common way should be found for identifying performance aspects of Cloud environments.

The main performance aspects of Cloud computing as analysed by the ARTIST approach can be summarized as:

a) Heterogeneous and unknown hardware resources: the computing resources offered by the Cloud providers are unknown to the external users. Available information may be limited to number of cores for example, memory sizes or disk quotes. However this level of information is far from sufficient in order to characterize the provider's hardware capabilities that may depend also on architecture, interconnection, RAM speeds etc. According to a study on Amazon platform conducted by Aalto University (Zhonghong Ou et al., 2012), the variation between the fast instances and slow instances can reach 40%. In some

applications, the variation can even approach up to 60%.

b) Different configurations: even in the existence of the same hardware however, the way this resource is configured plays a significant role in its performance. The same applies for software configurations (e.g. a DB instance over a virtual cluster) or variations in the software development.

c) Multi-tenancy and obscure, black box management by providers: Cloud infrastructures deal with multiple different users that may start their virtual resources on the same physical host at any given time. However, the effect of concurrently running VMs for example (Kousiouris et al., 2011) significantly degrades the actual application performance. This is even more affected by the usage patterns of these resources by their virtual owners or their clients. Furthermore, consolidation decisions made by providers and that are unknown to the users may group virtual resources on the same physical node at any given time, without informing the owner.

d) VM interference effects. In (Koh et al., 2007) an interesting research investigates the performance interference for a number of applications in experimental virtual environments that were selected for classifying their behaviour using different metrics. The result from the research shows that combined performance varies substantially with different combinations of applications. Applications that rarely interfere with each other achieve performance to the standalone performance. However, some combinations interfere with each other in an adverse way. Furthermore, virtualization is a technology used in all Cloud data centers to ensure high utilization of hardware resources and better manageability of VMs. Despite the advantages provided by virtualization, they do not provide effective performance isolation.

All these aspects plus the fact that Cloud providers are separate entities and no information is available on their internal structure and operation, makes it necessary to **macroscopically** examine a provider's behaviour with regard to the offered resources and on a series of metrics. This process should be performed through benchmarking, by using the suitable tools and tests. One of the key aspects is that due to this dynamicity in resource management, the benchmarking process must be iterated over time, so that we can ensure as much as possible that different hardware, different management decisions (like e.g. update/reconfiguration/improvement of the infrastructure) are demonstrated in the refreshed metric values, but also observe key characteristics such as performance variation, standard deviation etc. Finally, the acquired information should be represented in a machine understandable way, in order to be used in decision making systems.

The aim of this paper is to provide such mechanisms to address the aforementioned issues. A benchmarking framework designed in the context of the FP7 ARTIST project is presented in order to measure the ability of various Cloud offerings to a wide range of applications, from graphics and databases to web serving and streaming. The framework has defined also a number of templates in order to store this information in a machine understandable fashion, so that it may be used by service selection mechanisms. What is more. we define a metric, namely *Service Efficiency* (SE), in order to rank different services based on a combination of performance, cost and workload factors.

The paper is structured as follows. In Chapter 2, an analysis of existing work is performed. In Chapter 3 the description of the ARTIST tools for mitigating these issues is presented, while in Chapter 4 a case study on AWS EC2 resources is presented. Finally, conclusions and future work are contained in Chapter 5.

2 RELATED WORK

Related work around this paper ranges in the fields of performance frameworks, available benchmark services and description frameworks and is based in the according analysis performed in the context of the ARTIST project (ARTIST Consortium D7.2, 2013). With regard to the former, the most relevant to our work is (Garg, 2012). In this paper, a very interesting and multi-level Cloud service comparison framework is presented, including aspects such as agility, availability, accountability, performance, security and cost. Also an analytical hierarchical process is described in order to achieve the optimal tradeoff between the parameters. While more advanced in the area of the combined metric investigation, this work does not seem to include also the mechanism to launch and perform the measurements. Skymark (Iosup et al., 2012) is a framework designed to analyze the performance of IaaS environments. The framework consists of 2 components – Grenchmark and C-Meter. Grenchmark is responsible for workload generation and submission while C-Meter consists of a job scheduler and submits the job to a Cloud manager

that manages various IaaS Clouds in a pluggable architecture. Skymark focuses on the low level performance parameters of Cloud services like CPU, Memory etc. and not on elementary application types.

CloudCmp (Li, 2010) provides a methodology and has a goal very similar to our approach to estimate the performance and costs of a Cloud deployed legacy application. A potential Cloud customer can use the results to compare different providers and decide whether it should migrate to the Cloud and which Cloud provider is best suited for their applications. CloudCmp identifies a set of performance metrics relevant to application performance and cost, develop a benchmarking task for each metric, run the tasks on different providers and compare. However CloudCmp does not seem to define a common framework for all the benchmark tasks.

With regard to benchmarking services, the most prominent are CloudHarmony.com and CloudSleuth.com. The former utilizes a vast number of benchmarks against various Cloud services, offering their results through an API. However, there are two aspects that can be improved with relation to this approach. Initially it is the fact that too many benchmarks are included in the list. We believe that a more limited scope should be pursued in order to increase the focus of the measurements. Furthermore, the measurement process is not repeated on a regular basis, in order to investigate aspects such as deviation. For CloudSleuth, the focus is solely on web-based applications and their response time/availability. Their approach is very worthwhile, by deploying an elementary web application across different providers and monitoring it constantly, however it is limited to that application type.

With regard to description frameworks, a number of interesting approaches exist. According to the REMICS project (REMICS Consortium Deliverable D4.1 v2.0, 2012) PIM4Cloud, which is focused in both private and public Clouds, has been defined to provide support to model the applications and also to describe the system deployment on the Cloud environment. PIM4Cloud is implemented as a profile for UML and a meta-model which is capable to describe most of the features of a system that will be deployed in a Cloud environment. It is organized in four packages (Cloud Provider domain, Cloud Application domain, Physical Infrastructure domain and Resource domain).

FleRD (Schaffrath et al., 2012) is a flexible resource description language for inter-provider communication in virtual networks architectures. It appears enhanced with regard to realism and generality (ability to describe real world topologies), extensibility, grouping and aggregation. FleRD is mainly focused around networking elements, however its concepts of modeling more information for QoS of networks has influenced our approach.

EDML (Charlton, 2009) defines a XML syntax for declaring internal and external general parsed entities. VXDL(Koslovski et al, 2008) is an XML-based language that describes Virtual Private eXecution Infrastructure (ViPXi) which is a time-limited organized aggregation of heterogeneous computing and communication resources. VXDL can describe resources, networks' topology that are virtual but are also, to some extent, adapted to physical ones and finally to represent timeline.

The implementation of DADL (Mirkovic et al., 2010) is based on the prediction that future businesses will use allocated resources from different Clouds such as public or private to run a single application. DADL was developed as an extension of SmartFrog (framework for deploying and managing distributed software systems based on java) and it is used to specify application architecture and Cloud resources that are necessary for an application to run. There are elements to describe QoS features such as CPU speed, number of cores etc.

The main issue with the aforementioned approaches, which most of them support description of QoS terms, is the fact that in many cases the standard ways (CPU cores, frequency etc.) of describing capabilities are not sufficient to demonstrate the actual performance of virtualized resources. Thus a new approach based on benchmark scores should be pursued that would indicate the direct capability of a resource service to solve a particular computational problem. The descriptions defined in this paper are centered around this test-based approach.

3 BENCHMARKING APPROACH IN ARTIST

The benchmarking approach followed in the context of ARTIST has the following targets:

- Identify a set of common application types and the respective benchmarks to measure the performance of Cloud services
- Create a framework that is able to automatically install, execute and retrieve the benchmark

results, with the ability to support multiple providers
- Investigate aspects of Cloud service performance with regard to variation and ability across a wide range of potential applications
- Define a machine understandable way of representing this information and improved metrics that will characterize more efficiently the services.

The use case diagram for the benchmarking suite appears in Figure 1. We envisage that the capabilities of the toolkit will be exploited by an entity ("Benchmarks Provider") that will be responsible for performing and obtaining the tests, similar to the role of CloudHarmony.com. This entity will utilize the various aspects of the toolkit in order to create provider models that have concrete results and metrics per service offering, that are stored on the ARTIST repository, so that an external entity ("Model User") may be able to retrieve and consult them. More details on each part of the process are presented in the following paragraphs.

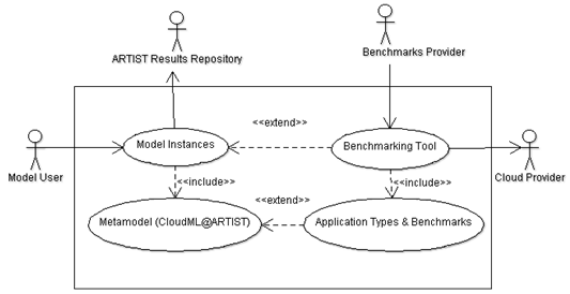

Figure 1: Use case diagram for the ARTIST Benchmarking process and tools.

3.1 Application Benchmark Types

The main target of the application benchmark types is to highlight a set of common and popular application tests that can be used in order to benchmark provider offerings. Thus each offering may have a performance vector indicating its ability to solve specific problems or cater for a specific type of computation. The set of application benchmarks used in ARTIST appears in Table 1.

Table 1: Benchmark Tests used in the ARTIST platform.

Benchmark Test	Application Type
YCSB	Databases
Dwarfs	Generic Applications
Cloudsuite	Common web aps like streaming, web serving etc.
Filebench	File system and storage

For the future these tests may be extended with other specialized tests like DaCapo benchmarking suite (Blackburn et al., 2006) for measuring JVM related aspects.

3.2 Models of Necessary Information

In order to exploit the information from the benchmark execution, a suitable machine understandable format should be in place in order to store results and utilize them in other processes like service selection. For achieving this, suitable model templates have been designed. These templates include all the relevant information needed, such as measurement aspects (number of measurements, statistical information like standard deviation etc.), test configurations and workload profiles. Initially these templates are defined as an XML schema and in later stages they will be incorporated into a suitable metamodel developed in the context of the ARTIST project (CloudML@ARTIST). Examples of types defined in the schema are portrayed in Figure 2.

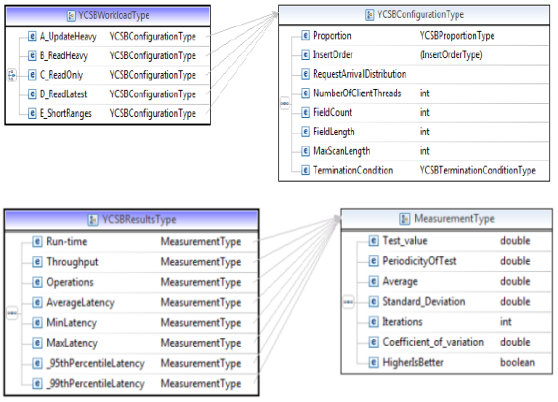

Figure 2: Examples of templates for describing benchmarking information in a machine understandable fashion. Each test is represented by an according template.

3.3 Benchmarking Suite Architecture

The Benchmarking Suite Architecture appears in Figure 4. The user through the GUI (Figure 3) may set the conditions of the test, selecting the relevant benchmark, workload conditions, target provider and service offering. This information is passed to the Benchmarking Controller which is responsible for raising the virtual resources on the target provider and executing the tests. The former is based on the incorporation of Apache LibCloud project, in order to support multiple provider frameworks. The latter needs to install first the tests, through the utilization

of an external Linux-like repository that contains test executables. Once the tests are installed (through a standard repo-based installation), the workload setup scripts are transferred to the target machines and the execution begins. Results are transferred back and processed in order to be included in the model descriptions, following the template definition highlighted in Section 3.2.

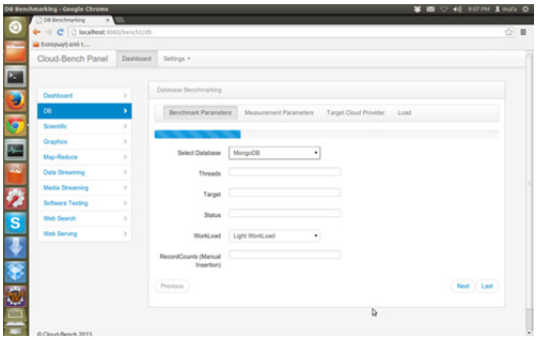

Figure 3: GUI for selecting and configuring tests on provider offerings.

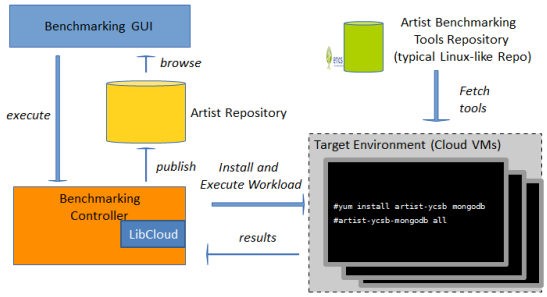

Figure 4: Overall System Architecture.

At the moment the two parts (GUI and Controller) are not integrated, however the benchmarking controller which is the main backend component can also be used standalone to perform measurements on the various application types. Installation instructions can be found in (ARTIST Consortium D7.2.1, 2013).

3.4 Service Efficiency Metric Description

In order to better express the performance ability of a service offering, we considered the usage of a metric that would fulfill the following requirements:

- Include workload aspects of a specific test
- Include cost aspects of the selected offering
- Include performance aspects for a given workload
- Give the ability to have varying rankings based on user interests
- Intuitively higher values would be better

Following these points, we considered that positive factors (e.g. workload aspects) should be used in the numerator and negative on the denominator (cost and Key Performance Indicators that follow a "higher is worse" approach). Furthermore, normalization should be used in order to have a more generic rating. Usage of sum would be preferable over product, since the former enables us to include weight factors. Thus such a metric can answer a potential question of the sort: "what is the best service to run my web streaming application, when I am interested more on a cheap service?".

The resulting formula of Service Efficiency is the following (Equation 1):

$$SE = \frac{\sum_i s_i l_i}{\sum_j s_j w_j f_j} \quad (1)$$

Where s scaling factor
l: workload metric
f: KPI or cost metric
w: weight factor

The resulting metric can be compared between different offerings, potentially of different providers **but on the same workload basis.** However the incorporation of workload is necessary since it affects the performance and thus ranking of the services. Usage of different workloads may display a different optimal selection, based on the anticipated workload conditions for the application that needs to be deployed.

4 METRIC CASE STUDY ON AMAZON EC2

In order to experiment with the metrics definitions and initially investigate differences in VM performance, we utilized the service described in (Kousiouris et al, 2013). This is a service implementation offering time series prediction, by utilizing a backend Matlab implementation of prediction models. The instance on which the service was running was changed in order to test multiple types of Amazon EC2 VM instances (micro, small, c1.medium, m1.medium).

The number of concurrent clients was set to 1, 5 and 10, with 10 representing the maximum workload (it is a computationally heavy service and it was observed that after 10 clients the delay made it

unusable). Each client launches constantly requests against the service in a serial manner, so that at all times there are that many pending requests equal to the client number. The number of calls made by a client thread is regulated according to the number of clients workload in order to accumulate 500 values for each combination 500. From these the average value and standard deviation were extracted for each case (Table 2).

Table 2: Average Measurements and Cost for combinations of EC2 instances and workload.

Instance-Clients	Avg Delay (msec)	Std Dev (msec)	Cost/Hour
micro-1	2.95E+04	1.39E+04	0.02
micro-5	1.27E+05	1.17E+05	0.02
micro-10	3.37E+05	3.38E+04	0.02
small-1	1.11E+04	558.5749	0.06
small-5	4.75E+04	3.10E+03	0.06
small-10	1.00E+05	8.92E+03	0.06
m1.medium-1	6.64E+03	106.5448	0.12
m1.medium-5	2.20E+04	2.27E+03	0.12
m1.medium-10	4.36E+04	1.27E+03	0.12
c1.medium-1	5.92E+03	61.5125	0.145
c1.medium-5	1.06E+04	954.8628	0.145
c1.medium-10	2.23E+04	1.41E+03	0.145

Then the metric SE described in Section 3.4 was applied with the following form (results appear in Figure 5):

$$SE = \frac{\#Clients}{w_1 * \text{delay} + w_2 * Cost}$$

Different weights were given to the performance and cost aspects (50-50, 90-10,10-90) and different normalization intervals were considered (1-2,1-10) in order to check the metric's sensitivity. We avoided using a normalization interval including 0 since it may lead to infinite values for some cases. One should compare between same color bars, indicating similar workloads. From the graphs it is evident how the ranking of a potential VM rating can change based on the aspect that we are most interested in. For example, for a weighted decision (50-50) with high workload, the best choice would be EC2 small instance, while for a performance-biased selection (90-10) one should focus on c1.medium instances. For cost-biased selection, one should focus on micro instances. While this may seem the obvious choice (cheaper VM for the cost-biased and more expensive for the performance-biased), this is only evident in the extreme cases. If we need an intermediate trade-off then the choice is not that obvious and the defined metric could help in this selection process.

In the future the measurements will be based on the elementary application types and benchmarks that are described in Section 3.1, in order to obtain more generic and reusable (not case specific) ratings.

Another variation that was pursued was the incorporation of the standard deviation measurements as a KPI, in order to include the interest to rank services based on their stability. Thus the metric equation was altered as follows:

$$SE = \frac{\#Clients}{w_1 * \text{delay} + \text{w}\,2 * \text{deviation} + w_3 * Cost}$$

The contents of Table 2 were then used for the combinations of equal interest (0.33 weight for all

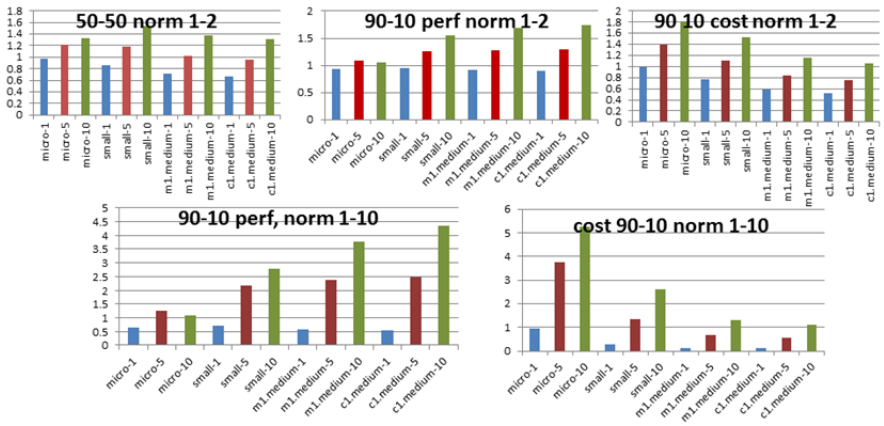

Figure 5: Application of the metric on an example service response times across different type of Amazon EC2 VM instances and variable workload (higher score is better). Comparison should be made between columns with similar color (identical workload conditions).

parameters) and performance biased interest including deviation (0.45 for performance, 0.45 for deviation and 0.1 for cost) and were compared to the standard 50-50 graph for performance and cost in Figure 6. The deviation affects the ranking (for example in the m1.medium.10 case for high workload). This would be especially helpful in cases where we have applications that are very sensitive to deviation phenomena like multimedia streaming applications.

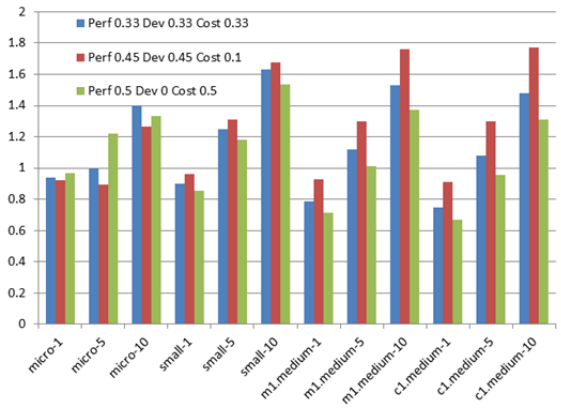

Figure 6: Inclusion of deviation as a KPI in the formula and comparison for various combinations in the normalization interval 1-2.

5 CONCLUSIONS

The emergence of Cloud computing has led to a plethora of various offerings by multiple providers, covering different needs of the consumer. However significant questions emerge for the stability of these pay-per-use resources, mainly with regard to performance, in this highly dynamic management environment. In this paper a multi-Cloud measurement framework has been presented, that has the aim of investigating these performance issues of the virtualized offerings. The framework utilizes a variety of benchmark tests in order to cover a wide range of application types and it mainly focuses on investigating aspects such as performance variations.

A combined metric (Service Efficiency) is also proposed in order to combine workload, performance and cost aspects in a single rating for comparing Cloud offerings across different providers. A case study on the Amazon EC2 Cloud has indicated the application of such a metric to characterize the offerings based on this combination.

For the future, we intend to complete the integration of the framework (currently missing the integration between GUI and Benchmark Suite Controller) and investigate the addition of tests with extended focus (e.g. JVM aspects, availability measurement aspects etc.). However the Benchmarking Controller for the execution of the tests can be used also as standalone, following the instructions in (ARTIST Consortium D7.2.1, 2013). Another interesting aspect would also be the incorporation of other non-functional aspects such as availability in the main SE metric and a comparison across different provider offerings.

ACKNOWLEDGEMENTS

The research leading to these results is partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 317859, in the context of the ARTIST Project.

REFERENCES

ARTIST Consortium (2013), Deliverable D7.2 v1.0-PaaS/IaaS Metamodelling Requirements and SOTA, Available at: http://www.artist-project.eu/sites/default/files/D7.2%20PaaS%20IaaS%20metamodeling%20requirements%20and%20SOTA_M4_31012013.pdf.

REMICS Consortium (2012), Deliverable D4.1 v2.0 - PIM4Cloud, Available at: http://www.remics.eu/system/files/REMICS_D4.1_V2.0_LowResolution.pdf.

Kousiouris, George; Vafiadis, George; Varvarigou, Theodora, "Enabling Proactive Data Management in Virtualized Hadoop Clusters Based on Predicted Data Activity Patterns," P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), *2013 Eighth International Conference on*, vol., no., pp.1,8, 28-30 Oct. 2013.

doi: 10.1109/3PGCIC.2013.8.

George Kousiouris, Dimosthenis Kyriazis, Andreas Menychtas and Theodora Varvarigou, "Legacy Applications on the Cloud: Challenges and enablers focusing on application performance analysis and providers characteristics", in *Proceedings of the 2012 2nd IEEE International Conference on Cloud Computing and Intelligence Systems (IEEE CCIS 2012)*, Oct. 30th ~ Nov. 1st,Hangzhou, China.

Hauck, M., Huber, M., Klems, M., Kounev, S., Muller-Quade, J., Pretschner, A., Reussner, R., and Tai, S. Challenges and opportunities of Cloud computing. Karlsruhe Reports in Informatics 19, Karlsruhe Institute of Technology - Faculty of Informatics, 2010.

Zhonghong Ou, Hao Zhuang, Jukka K. Nurminen, Antti Ylä-Jääski, and Pan Hui. 2012. Exploiting hardware heterogeneity within the same instance type of Amazon EC2. In *Proceedings of the 4th USENIX*

conference on Hot Topics in Cloud Ccomputing (HotCloud'12). USENIX Association, Berkeley, CA, USA, 4-4.

George Kousiouris, Tommaso Cucinotta, Theodora Varvarigou, "The Effects of Scheduling, Workload Type and Consolidation Scenarios on Virtual Machine Performance and their Prediction through Optimized Artificial Neural Networks , *The Journal of Systems and Software (2011)*,Volume 84, Issue 8, August 2011, pp. 1270-1291, Elsevier, doi:10.1016/ j.jss.2011.04.013.

Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu. An analysis of performance interference effects in virtual environments. In *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 200–209, April 2007.

Schaffrath, Gregor and Schmid, Stefan and Vaishnavi, Ishan and Khan, Ashiq AND Feldmann, Anja, 2012. A Resource Description Language with Vagueness Support for Multi-Provider Cloud Networks. Munich, Germany: *International Conference on Computer Communication Networks (ICCCN '12)*.

Charlton, S. (2009). Model driven design and operations for the Cloud. In OOPSLA09, *14th conference companion on Object Oriented Programming Systems Languages and Applications*, pages 17–26.

Koslovski, G., Primet, P.V.-B., Charao, A. S.: VXDL: Virtual Resources and Interconnection Networks Description Language. In: GridNets 2008 (October 2008).

Mirkovic, J., Faber, T., Hsieh, P., Malayandisamu, G., Malavia, R.: DADL: Distributed Application Description Language. USC/ISI Technical Report ISI-TR-664 (2010).

Stephen M. Blackburn, Robin Garner, Chris Hoffmann, Asjad M. Khang, Kathryn S. McKinley, Rotem Bentzur, Amer Diwan, Daniel Feinberg, Daniel Frampton, Samuel Z. Guyer, Martin Hirzel, Antony Hosking, Maria Jump, Han Lee, J. Eliot B. Moss, Aashish Phansalkar, Darko Stefanović, Thomas VanDrunen, Daniel von Dincklage, and Ben Wiedermann. 2006. The DaCapo benchmarks: java benchmarking development and analysis. SIGPLAN Not. 41, 10 (October 2006), 169-190. DOI=10.1145/1167515.1167488 http://doi.acm.org/10.1145/1167515.1167488.

Saurabh Kumar Garg, Steve Versteeg, Rajkumar Buyya, A framework for ranking of Cloud computing services, Future Generation Computer Systems, Volume 29, Issue 4, June 2013, Pages 1012-1023, ISSN 0167-739X, http://dx.doi.org/10.1016/j.future.2012.06.006.

A. Iosup, R. Prodan, and D. Epema, "Iaas Cloud Benchmarking: Approaches, Challenges, and Experience," *Proc. of the Int'l Conf. on High Performance Networking and Computing (SC)*, MTAGS 2012 . IEEE/ACM, pp. 1–8.

Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. 2010. CloudCmp: comparing public Cloud providers. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement (IMC '10)*. ACM, New York, NY, USA, 1-14. DOI=10.1145/1879141.1879143 http://doi.acm.org/10.1145/1879141.1879143.

ARTIST Consortium (2013), Deliverable D7.2.1 v1.0-Cloud services modelling and performance analysis framework, Available at: http://www.artist-project.eu/sites/default/files/D7.2.1%20Cloud%20services%20modeling%20and%20performance%20analysis%20framework_M12_30092013.pdf.