WEB SERVICE CPU OVERUTILIZATION IN THE CLOUD

Marjan Gusev, Goran Velkoski, Sasko Ristov and Monika Simjanoska

University Sts Cyril and Methodius, Faculty of Information Sciences and Computer Engineering
Skopje, Macedonia
marjan.gushev@finki.ukim.mk, velkoski.goran@gmail.com, sashko.ristov@finki.ukim.mk, m.simjanoska@gmail.com,

Abstract

The goal in this paper is to compare the CPU utilization when one would like to migrate its applications to the cloud. This will allow the user to specify the real capacity of the server in order to match the foreseen memory and computational demands. Each overutilization of the CPU will significantly impact the performance. In this paper we analyze two different types of web services, defined on memory and computation demands. The first web service is memory only demanding and the second is both memory demanding and computation intensive. The experiments are realized on two environments using the same hardware, the first is on-site environment, and the second is cloud environment, where on top of the first we include private OpenStack cloud solution. The comparison of results will show what is the impact of cloud deployment, and also how the load of web services impacts the CPU utilization.

Keywords - Performance, CPU utilization, Cloud deployment impact, Workload impact

1 INTRODUCTION

Cloud computing is a reality and most of the companies are thinking to migrate their applications onto cloud. The research on how to migrate applications onto cloud and how much the performances are affected by moving onto cloud is a hot topic lately.

Performance analysis of services and comparison of what happens when moving onto cloud are analyzed in [5]. Our analysis showed that most of the authors measured response times and speed for various algorithms in the cloud and on-premise [1, 2, 10, 7]. The authors in [6] present performance tools for an open-source application framework and findings of a detailed assessment of web services performance. Web service data forwarding has significant performance advantage over normal web service framework for workflows with large data transfer [9]. Some of the authors have shown speedup comparisons or published how much the performance is degraded when the web services are hosted on premise or in the cloud [5]. Most of these efforts showed the impact of cloud deployment, i.e. what happens when virtualization and multi-tenancy are deployed and performance interference among multiple VMs running on the same hardware platform with the focus on network I/O processing [8].

The hypothesis we would like to confirm is that the CPU utilization increases with same growth factor both by increasing the workload of the web service and the number of web services. In addition we would like to find out if there is correlation between performance degradation and the CPU utilization.

However, the CPU utilization was not analyzed in analyses published in recent papers. Our goal was to analyze the CPU utilization and how it affects the impact of cloud deployment. In this process we have also analyzed that most applications have capacity problems, initiated by memory demands or intensive computations. Therefore we have selected two representatives for web services, one that is only memory bound and the other that is both memory and computation bound. We have set the experiments with different loads, trying to simulate capacity constraints, the first increases directly the memory and computation demands of the web service, and the other increases the web server directly, by specifying more executions of the same web service. The experiments will present what is the effect of these input parameters.

The rest of the paper is organized as follows. Section 2 presents the testing methodology including descriptions on test cases. Results are presented in Section 3 with appropriate explanations. Comparison and further discussion is presented in Section 4. Conclusions and future work are presented in Section 5.

2 TESTING METHODOLOGY

Testing methodology is presented in this section with detailed explanation of testing environment, test scenarios, test data and test plan.

2.1 Testing Environment

Experiments are realized following two different environments:

(1) On-premise client server environment; and

(2) Cloud computing environment.

The first testing environment is presented in Fig. 1. The server is running on hardware layer consisting of Intel(R) Xeon(R) CPU X5647 @ 2.93GHz with 4 cores and 8GB RAM. The next level is the operating system which is 64-bit Ubuntu Server 12.04 LTS. The application layer is presenting the web server Apache Tomcat 6.0. Two different test scenarios are realized by different web services as Concat and Sort JAX-RPC web services.

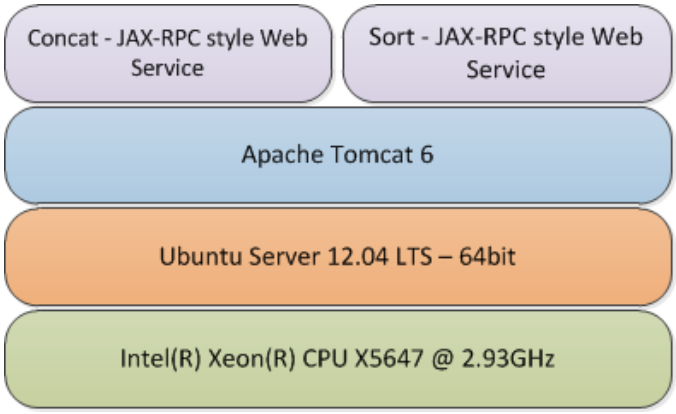

Fig. 1. On-premise testing environment

The second environment is depicted in Fig. 2. Although it is cloud based, it still uses the same hardware and operating system as in the first environment. OpenStack cloud solution software is deployed in dual node with single Controller and Compute Node both with equal hardware infrastructure over the operating system. KVM hypervisor is being used to instantiate virtual machines (VMs) in the cloud. The VM used for testing is setup using 4 CPUs, exactly the same as in the traditional client-server scenario. Software used in VM is by all means the same in order to guarantee equal testing environment.

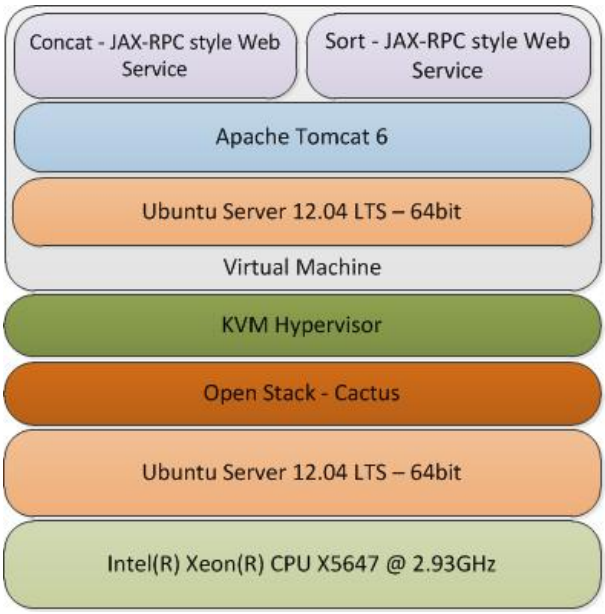

Fig. 2. Cloud testing environment

2.2 Test Data

The main goal of the experiment is to measure the CPU utilization for different scenarios. This measurement will help making decisions about achieved performances. The following tools are used to measure the CPU utilization.

Fig. 3 and Fig. 4 describe the setup of experiments for both testing environments. XML messages are exchanged in both environments by using SoapUI [3].

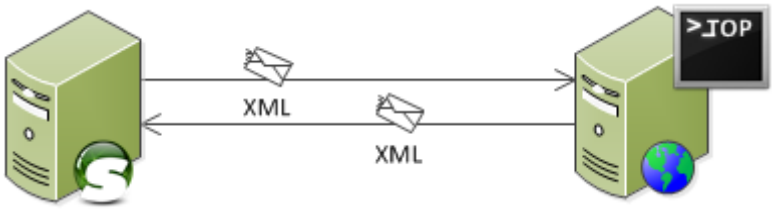

Fig. 3. Experiment setup - On-premise

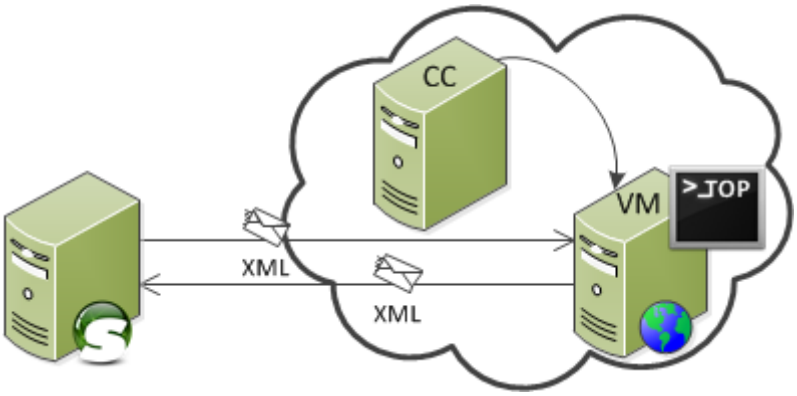

Fig. 4. Experiment setup - Cloud

We use the same client in both environments. The client hardware is workstation with 4GB DDR2 RAM and Intel(R) Core(TM)2 Quad CPU Q9400 @ 2.66GHz. The client software is SoapUI installed on Linux Ubuntu 12.04 operating system. SoapUI is Open Source Functional Testing tool for Web Service Testing and we used it for varying web server load implying. Linux based program top [4] is being used for server side CPU utilization measurement or Cloud VM CPU utilization measurement depending on the testing scenario. In order to minimize network latency the client PC is on the same VLAN as the web servers [11].

Top utility is running on server side and records CPU utilization. Screen updates are done every three seconds; consequently for each test case twenty CPU utilization numbers are collected. In the end the average of these values is calculated.

2.3 Test scenarios

In this section we describe two test cases to be realized in the experiment. The goal of test cases is to test web services with different computational characteristics: memory or computational demanding. For this purpose we constructed Concat and Sort web services. Both web services accept two string parameters. The Concat web service returns the concatenation of both strings, while the Sort web service realizes the concatenation and returns sorted parameter concatenation.

The analysis shows that Concat web service is memory only demanding web service. It's memory complexity is *O(M)* where *M* is the parameter size.

Sort web service is both memory and computation demanding web service with memory complexity of *O(M)* and time complexity of *O(M * Log M)*. It is realized with the merge sort algorithm.

The scenario behind these two test cases is to measure the performance and CPU utilization while changing the load. The load change is achieved by different parameter sizes *M* of the messages and different number of messages *N*.

Parameter size *M* used in our experiments varies in the following range: 0KB, 1KB, 2KB, 4KB, 5KB, 6KB, 7KB, 8KB, and 9KB.

Additionally, for each of these experiments a varying number of requests per second *N* is being sent in order to examine web server CPU utilization based on workload mode. The values for *N* are found in the following range: 12, 100, 500, 750, 1000, 1250, 1500, 1750 and 2000 requests per second.

2.4 Test Plan

The test plan consists of execution of two series of test cases. The first series consist of test cases that examine the impact of message size variation on web server CPU utilization and the second series consist of test cases that invoke analysis of the increasing number of concurrent messages impact on CPU utilization. Both of these test case series are performed on (1) web services hosted on-premise; and (2) web services hosted in the cloud.

Each test case runs for 60 seconds sending *N* messages, each with parameter size of *M*. The variance is 0,5. We don't use burst or overload mode but regular web server load mode.

3 RESULTS AND ANALYSIS

This section describes CPU utilization testing results and the impact of cloud virtualization layer. The analysis emphasizes the impact of different message sizes and number of concurrent messages for both test cases defining memory demanding and computation intensive web services. The CPU utilization is being normalized using (1), where *n* is the number of cores in the host used for testing and *U(n)* is the utilization recorded when web services are hosted on *n*-core host. By this normalization we scale each CPU utilization from 0 to 1 to enable easy comparison.

$$NU = \frac{U(n)}{n} \qquad (1)$$

3.1 Web Service CPU Utilization Hosted On-premise

The purpose of this experiment is to measure the CPU utilization and the impact of web services hosted on-premise with different: (1) message size for constant number of concurrent messages; and (2) number of concurrent messages for constant message size.

Fig. 5 depicts Concat web service CPU utilization while hosted on-premise. We can conclude that both input parameters *M* and *N*, i.e. message size and requests per second impact the CPU utilization. The message size *M* impacts the CPU utilization proportionally and gradually increasing with small factor. The effect of the number of requests per second *N* is manifested with significant factor increase of CPU utilization.

CPU utilization of Sort web service hosted on-premise is presented on Fig. 6. Again, we can conclude that both message size and requests per second impact the CPU utilization. Sort web service differs from Concat web service by the fact that CPU utilization strongly depends on the message size *M*, and the number of concurrent messages *N*.

3.2 Web Service CPU Utilization Hosted in the Cloud

This experiment aims to measure the CPU utilization impact of web services hosted in the Cloud.

The results obtained for Concat web service CPU utilization while hosted in the Cloud are presented in Fig. 7. As shown in the figure, both message size and requests per second impact the CPU utilization. The impact of the message size parameter *M* is proportional, i.e. increase of *M* results with a small increase of the CPU utilization approximately proportionally. Nevertheless, input parameter *N* impacts significantly the CPU utilization, meaning that small increase of the number of messages significantly increases the CPU utilization.

CPU utilization of Sort web service hosted in the cloud is presented on Fig. 8. As in previous cases the conclusion is similar, i.e. both message size and requests per second impact the CPU utilization. However, in the case of the Sort web service, the CPU utilization depends on both message size *M*, and number of concurrent messages *N* unlike the Concat web service, where the impact of the number of messages sent is greater than the message size. This is due to the fact that the Sort web service is both memory demanding and computation intensive web service.

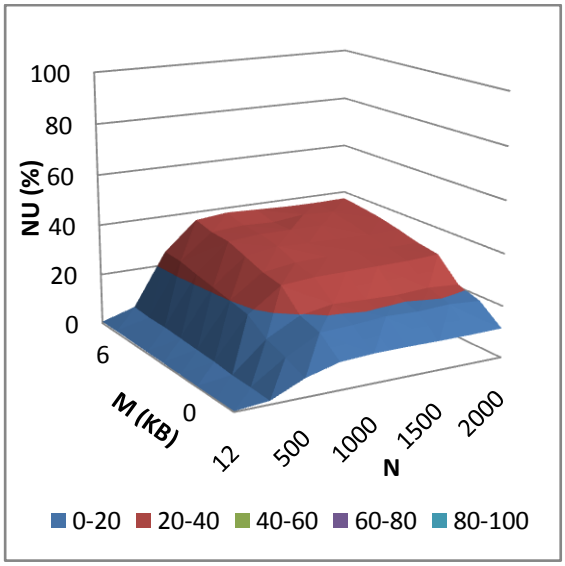

Fig. 5. Concat web service normalized CPU utilization while hosted On-premise

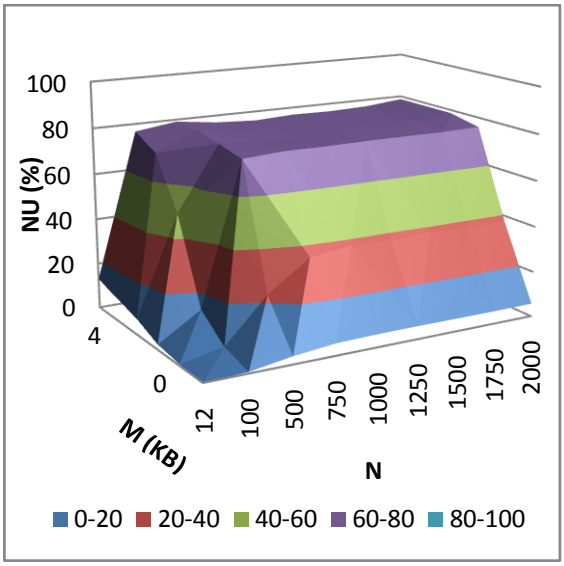

Fig. 6. Sort web service normalized CPU utilization while hosted On-premise

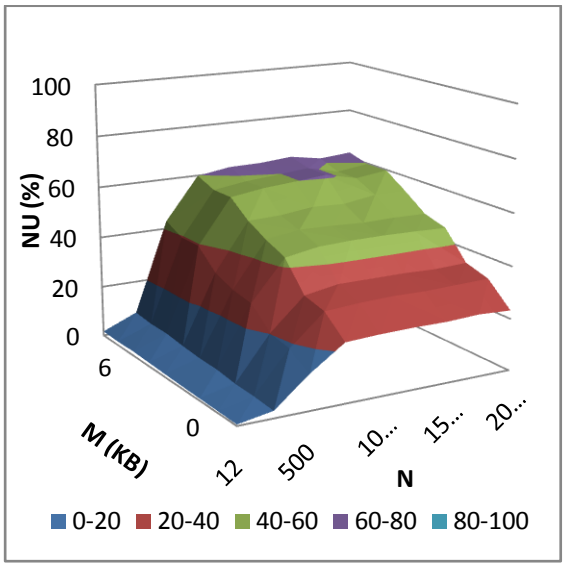

Fig. 7. Concat web service normalized CPU utilization while hosted in the Cloud

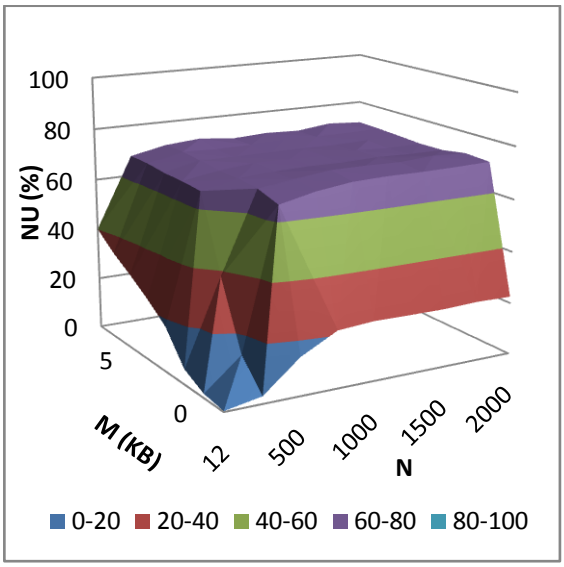

Fig. 8. Sort web service normalized CPU utilization while hosted in the Cloud

4 RESULTS DISCUSSION

In this section we will discuss the results presented in Section 3 with two goals:

- To compare the impact of both Concat and Sort web services; and
- To compare the results obtained in both hosted on-premise and cloud environments.

4.1 Memory Dependent Web Service vs. Memory and computation dependent Web Service

Table 1 contains maximum and minimum values expressed as percentages of web services CPU utilization hosted on-premise or in the cloud.

Based on these results we can conclude that the minimum CPU utilization recorded for web services hosted on-premise is the same for both Concat and Sort web services. Minimum recorded CPU utilization is for very small messages, *M*, and small number of concurrent web service requests. This can be justified with the fact that *M* is equal or close to 0 so the time overhead for computations in the Sort web service equals to 0. However, the maximum CPU utilization ranges from 38.05% to 79.12%

CPU utilization. This difference is due to the time overhead needed for computations, i.e. for sorting the input strings.

The minimum CPU utilization is insignificantly different when hosted in the Cloud, i.e. the impact of cloud deployment is insignificant compared to other costs mainly dependent on communication and web service execution. However, the maximum CPU utilization rises when hosted in the cloud for Concat web service with increases of both parameters *M* and *N*. The maximum CPU utilization for Concat web service is 63.18% and for Sort web service is 72.46%. This is enforced by Sort web service CPU utilization overhead for its computation intensive sorting algorithm.

Table 1. Maximum and minimum CPU utilization for web services hosted on-premise or in the cloud

Hosted on premise				Hosted in the Cloud			
Concat WS		Sort WS		Concat WS		Sort WS	
Min	Max	Min	Max	Min	Max	Min	Max
0.26%	38.05%	0.13%	79.12%	0.54%	63.18%	0.33%	72.46%

4.2 On-premise vs. Cloud hosted web services

In the previous subsection we analyzed different web services hosted in the same environment, i.e. on-premise and further on in the cloud. This section is dedicated to relative comparison between same web services hosted in different environments. The measure of relative utilization is defined in (2), where $U_{on\text{-}premise}$ is the relative CPU utilization for web services when hosted on premise and U_{cloud} is the relative CPU utilization for web services when hosted in the Cloud.

$$RU = \frac{U_{on-premise}}{U_{cloud}} \quad (2)$$

Fig. 9 depicts the relative CPU utilization for Concat web service hosted on premise and in the Cloud. We can conclude that web services CPU utilization when hosted in the cloud is always greater than when hosted on-premise. For small messages *M*, and small number of requests per second *N*, the relative speedup is minimal due to the small CPU utilization for these test cases. Increases of message size *M* and number of messages *N* impact the relative CPU utilization.

The experiments showed that the minimum relative CPU utilization is 0.37 and the maximum CPU utilization is 0.69, whereas the average relative CPU utilization is 0.58. These values are close to what authors found out in [5] for response time i.e. that the Cloud decreases the performance to 71.10% of on-premise, when using memory dependent web services.

Fig. 10 depicts the relative CPU utilization for Sort web service hosted on premise and in the Cloud. As in the previous case, we can also conclude that web services CPU utilization when hosted in the cloud is always greater than when hosted on-premise.

The relative speedup for small messages *M*, and small number of requests per second *N*, is minimal due to the small CPU utilization for these test cases. The impact when *M* and *N* reach higher values is expressed with relative CPU utilization steep growth.

Minimum relative CPU utilization is 0.23 and maximum CPU utilization is 1.15 whereas the average relative CPU utilization is 0.78. As in previous cases, these values are close to what authors published in [5] for response time i.e. that for memory dependent and computational intensive web services the Cloud decreases the performance to 73.86% of on-premise.

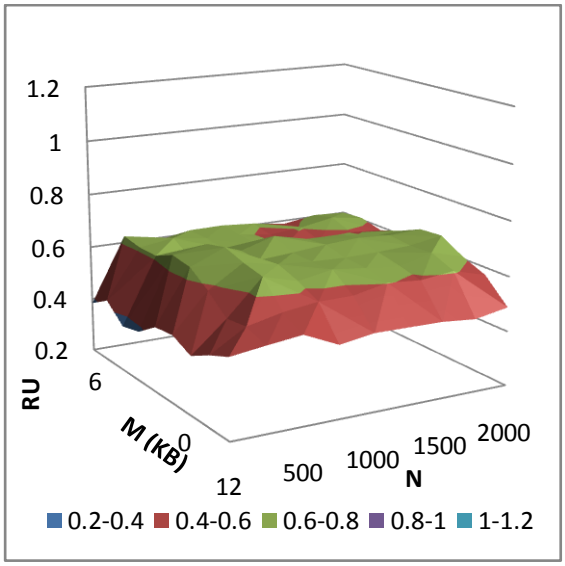

Fig. 9. Relative utilization for Concat web service

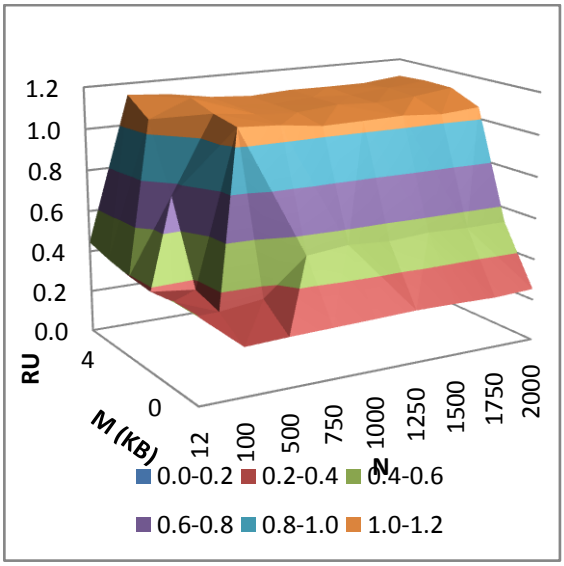

Fig. 10. Relative utilization for Sort web service

5 CONCLUSION AND FUTURE WORK

This paper analyzes two different types of web services, i.e. memory demanding, and both memory demanding and computation intensive web services. A series of experiments are realized on two environments using the same hardware environment, the first is on-site environment, and the second is cloud environment, where on top of the first we include private OpenStack cloud solution using the same operating system and web server. CPU utilization is measured during the experiments.

Input parameter number of concurrent messages directly impacts the CPU utilization for memory demanding web service, i.e. small increase of the number of messages will significantly increase the CPU utilization. However, both input parameters, i.e. number of messages and message size impact the CPU utilization for both memory demanding and computation intensive web service. These conclusions are valid while web services are hosted in both on-premise and cloud environments.

The results show that web service CPU utilization is always greater while hosted in the cloud compared to on-premise, for both web services. The drawbacks are more emphasized for memory demanding and computation intensive web service while it is loaded with greater message size.

In this paper we analyzed two different web services while hosted in the cloud and on-premise on the same hardware infrastructure. However, the real cloud environments is heterogeneous using thousands CPU cores orchestrated in virtual machine instances allocated with different computational resources. Our future research will be focused on normalized CPU utilization in heterogeneous multi-tenant cloud environment and on determining which resource allocation will provide minimal utilization of CPU cores for same web server load.

References

[1] S. Ristov, M. Gusev, M. Kostoska, K. Kjiroski, “Virtualized environments in cloud can have superlinear speedup,” in *Proceedings of the Fifth Balkan Conference in Informatics (BCI '12). ACM, New York, NY, USA, 8-13.*

[2] M. Gusev and S. Ristov, “Matrix multiplication performance analysis in virtualized shared memory multiprocessor," *in MIPRO, 2012 Proceedings of the 35th International Convention*, IEEE Conference Publications, May 2012, pp. 264-269.

[3] SoapUI: Functional testing tool (Dec. 2012), http://www.soapui.org/

[4] Ubuntu manuals. Top (Dec. 2012) http://manpages.ubuntu.com/manpages/lucid/man1/top.1.html

[5] S. Ristov, G. Velkoski, M. Gusev, M. Kostoska, and K. Kjirovski, “Compute and Memory Intensive Web Service Performance in the Cloud”, in *ICT Innovations 2012*, Advances in Intelligent and Soft Computing, (ed. S. Markovski and M. Gusev), Springer Verlag, Berlin Heidelberg, 2013, volume AISC 257, pp.215-224

[6] M. L. Pardal, J. P. Pardal, J. A. Marques, “Improving web services performance, one step at a time”. *In CLOSER* 2012, pages 542–551.

[7] Huang Liu; Xudong Liu; Jianxin Li; Yongwang Zhao; Zhuqing Li; , "Building high-speed roads: Improving performance of SOAP processing for cloud services," Service Oriented System Engineering (SOSE), 2011 IEEE 6th International Symposium on , vol., no., pp.72-78, 12-14 Dec. 2011

[8] Pu, X.; Liu, L.; Mei, Y.; Sivathanu, S.; Koh, Y.; Pu, C.; Cao, Y.; Liu, L.; , "Who is Your Neighbor: Net I/O Performance Interference in Virtualized Clouds," Accepted in *Services Computing, IEEE Transactions on*

[9] Donglai Zhang; Coddington, P.; Wendelborn, A.; , "Data Transfer Performance of Web Service Workflows in the Cloud Environment," *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on* , vol., no., pp.338-344, 5-8 Dec. 2011

[10] Minzhi Yan; Hailong Sun; Xu Wang; Xudong Liu; , "Building a TaaS Platform for Web Service Load Testing," Cluster Computing (CLUSTER), 2012 IEEE International Conference on , pp.576-579, 24-28 Sept. 2012

[11] Juric, M. B., Rozman, I., Brumen, B., Colnaric, M., and Hericko, M. (2006). Comparison of performance of web services, ws-security, rmi, and rmi-ssl. J. Syst. Softw., 79(5):689–700.