eChallenges e-2011 Conference Proceedings
Paul Cunningham and Miriam Cunningham (Eds)
IIMC International Information Management Corporation, 2011
ISBN: 978-1-905824-27-4

SLAs in Virtualized Cloud Computing Infrastructures with QoS Assurance[1]

Tommaso CUCINOTTA[1], Spyridon GOGOUVITIS[2], Kleopatra KOSTANTELI[2]
[1]*Scuola Superiore Sant'Anna, Pisa, Italy*
Email: cucinotta@sssup.it
[2]*National Technical University of Athens, Athens, Greece*
Email: {kkonst,spyros}@mail.ntua.gr

Abstract: Cloud Computing is gaining momentum as one of the technologies that promises to subvert our own idea of computing. In this paper, the challenging problem is discussed of how to ensure predictable levels of Quality of Service (QoS) to cloud applications across the multiple layers of a typical cloud infrastructure, and how a reasonable Service Level Agreement (SLA) management and enforcement policy might look like. The scope of this paper represents a hands-on experience that was gained by the authors realising the IRMOS real-time cloud-computing infrastructure in the context of the IRMOS European Project[2].

1. Introduction

Cloud Computing is gaining momentum as one of the technologies that promises to subvert our own idea of computing. With an increasing usage of cloud applications and their consequent dependency from connectivity, the nowadays Personal Computer is becoming merely a mobile device acting as a front-end to on-line applications and services. This huge paradigm shift in computing is witnessed for example by big market players who announced the imminent launch of innovative products and Operating Systems (like Chrome notebooks and the accompanying Chrome OS[3]. by Google), which are capable of projecting the user into the network in a few seconds by booting and starting immediately a web browser and (mostly) nothing else. In such a challenging scenario, more and more of the applications that we traditionally used locally on our PC are being hosted on cloud infrastructures and operated remotely through the Internet. This includes not only batch tasks, but also interactive applications which need to operate inherently with good levels of responsiveness. Therefore Quality of Service (QoS) guarantees need to be provided by all the different layers of the Cloud stack.

As we know, provisioning of Cloud Computing applications and services is nowadays following a business model that foresees three major business actors:

- an Infrastructure-as-a-Service (IaaS) provider, responsible for managing physical resources and exposing them in a virtualized fashion to other providers;
- a Platform-as-a-Service (PaaS) provider, responsible for providing a software development infrastructure that enables an easy development of applications over the cloud environment;

[1] The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7 under grant agreement n.214777 IRMOS -- Interactive Realtime Multimedia Applications on Service Oriented Infrastructures.
[2] More information is available at: http://www.irmosproject.eu/.
[3] More information is available at: http://www.google.com/chromeos/.

- a Software-as-a-Service (SaaS) provider, responsible for offering fully operational applications and services into the cloud.

Focusing on the lowest IaaS and highest SaaS levels, it is evident that these business partners have quite different and somewhat conflicting requirements. IaaS providers would like to host their physical resources in a virtualized fashion so as to hide as much as possible from the other providers internal details such as: what resources are exactly available and what their interconnection topology is, what their actual "bare metal" performance is, etc. This is needed in order to guarantee a sufficient degree of flexibility in managing the physical resources. For example, an IaaS provider might be willing to merely rent the physical resources it owns to the higher-level infrastructure. However, at the same time it might want to perform server consolidation and push more Virtual Machines onto the same physical hosts, in order to minimize the costs of running the infrastructure, or simply for maintenance reasons, whenever the contractual constraints to be respected (including QoS constraints) would allow it. Or, as a further case, the IaaS provider might have renewed its physical host equipments, in such a way that the new hardware is capable of running more services, applications and Operating Systems that used to run separately on the old machines. While performing such consolidation actions, it is of paramount importance that the provider respects the QoS constraints as arising from SLAs that regulate the levels of service expected by the customer (paying the resources rental fees) and the end users. On the other hand, the SaaS provider may need a quite detailed knowledge about the hardware running the (virtualised) deployed software components, in order to be able to produce accurate and reliable estimates of the QoS that will be offered to the final users of the applications, given a certain configuration in terms of rented/reserved resources from the IaaS provider. At the same time, any dynamic reconfiguration performed by the IaaS provider internally may act negatively on the performance as estimated by the SaaS provider and experienced by the final users of the virtualised applications.

These contrasting needs become critical in the moment in which precise QoS levels become an important requirement clearly and formally stated in the context of a SLA between the SaaS and the users/customers.

In what follows, these issues are detailed with a particular focus on the experience gained by the authors in the context of the IRMOS European Project, right after an overview of the related work in the research literature.

2. Related Work

The problem of guaranteeing a stable performance to Virtual Machines has been investigated in the literature by a number of authors. For example, Lin and Dinda [6] analysed the impact of using an EDF-based scheduling algorithm [7] for Linux to schedule Virtual Machines (VMs). However, they used a scheduler built into a user-space process (VSched), suffering of high overheads, and they did not use proper algorithms for ensuring the temporal isolation among VMs in presence of I/O operations causing the VMs to block/unblock. These problems are avoided in the IRMOS real-time scheduler [8], which use a variant of the CBS algorithm [9], directly implemented as a kernel-level scheduling policy. Lin and Dinda [10] also proposed that the users of a VM may be given the opportunity to provide feedback on the quality of the experience so that the allocated CPU could be adaptively changed. The main problem of such a technique is that it cannot be automated, and the user needs to be trusted.

Nathuji et al. [11] focused on automatic on-line adaptation of the CPU allocation in order to keep a stable performance of VMs. The framework does not treat a VM as a "black-box", but it needs application-specific metrics in order to run the necessary QoS control loops, going beyond the common IaaS business model. An adaptive technique able to dynamically change the resources reservation was also proposed by Lee et al. [12].

Gupta et al. investigated on the performance isolation of virtual machines [13], focusing on the exploitation of various scheduling policies available in the Xen hypervisor [14]. Furthermore, Dunlap proposed [22] various enhancements to the Xen credit scheduler in order to face with various issues related to the temporal isolation and fairness among the CPU share dedicated to each VM. However, in the IRMOS project the focus has been on the KVM hypervisor[4], in which Linux is used as host OS. In the context of IRMOS, we also showed how to provide isolation of compute-intensive [15] and network-intensive [16] VMs, and we also addressed the modelling issues related to the deployment of an e-Learning application with proper QoS guarantees [4].

Shirazi et al. [17] presented DynBench, an interesting benchmark for infrastructures supporting distributed real time applications. This creates dynamic conditions for the testing of the infrastructures. Germain et al. presented [18] DIANE for Grid-based user level scheduling. However, the focus is on controlling the execution end time of long processing applications, and not on real time interactive ones as done in this paper. Also, the problem of optimum allocation of workflows of virtualised services on a set of physical resources under a stochastic approach has been investigated by us in [4], for soft real-time interactive applications.

In terms of application performance modelling in distributed infrastructures a number of interesting works exists. A code analysing process that allows for the simulation of system performance is described in [19]. It models the application by a parameter-driven Conditional Data Flow Graph (CDFG) and the hardware (HW) architecture by a configurable HW graph. The simulator performs a low-level simulation to catch the detailed HW activities. While promising, it needs the source code to provide the CDFG.

Bekner et al. introduce the Vienna Grid Environment (VGE) [23], a framework for incorporating QoS in Grid applications. It uses a performance model to estimate the response time and a pricing model for determining the price of a job execution. In order to determine whether the client's QoS constraints can be fulfilled, for each QoS parameter a corresponding model has to be in place.

Other works exist that address QoS assurance in Grid environments focusing on performance prediction [20] and control via service selection [21].

3. Approach

3.1 The IaaS View

The specification of the QoS constraints that an IaaS provider should adhere to may be expressed in a variety of ways. It is easy to think of low level parameters that map directly to hardware characteristics, like the processor speed, architecture and capabilities, the configuration in terms of cache and main memory, the locally available persistent storage, etc. This type of specification is quite straightforward to respect in non-virtualized environments, because the provider can simply rent physical resources at fixed fees. However, in a virtualized context, a too precise and low-level requirement like “I need an AMD Opteron 6100 CPU at 2 GHz with this or that memory hierarchy layout” might be too constraining, because the provider might easily provide equivalent computing power under a completely different architecture. Therefore, one of the possible alternatives is to specify the performance requirements on the virtualized resources in terms of well-known benchmark figures. For example, when it comes to scientific applications, the underlying resources might need to respect a minimum throughput in terms of floating-point operations per second (FLOPS), or linear algebra operations per second (e.g., the LINPACK

[4] More information is available at: http://www.linux-kvm.org.

benchmark) or similar. For distributed applications, the problem of deploying multiple Virtual Machines with not only computing but also precise communication requirements makes the problem even more complex.

The software layer managing virtualized resources in the IRMOS architecture is the Intelligent Service-Oriented Networking Infrastructure (ISONI) [1]. It allows for booking virtualized resources by specifying a Technical SLA (T-SLA) which includes: the specification of a graph (a.k.a., Virtual Service Network) of computing and storage nodes, interconnected by a set of logical links, with computing and memory/storage requirements associated to each node and communication requirements to each link. This allows for a great flexibility in the kind of QoS specifications that may be submitted to an ISONI-enabled IaaS provider, including the possibilities sketched out above. Internally, ISONI is capable of respecting the real-time/QoS constraints as dictated at the T-SLA level by proper scheduling deployment policies and low-level resource scheduling techniques [2]. This includes the use of the IRMOS real-time scheduler [3] for providing strong execution guarantees to VMs hosted on the same CPU/core, and state-of-the art networking technologies [1] for providing precise QoS guarantees on the links of a VSN.

3.2 The SaaS View

The SaaS perspective is quite different, in that the interaction with the user and the provisioning of high-level services or even ready-to-use applications calls for a need of expressing QoS requirements in a completely different way, e.g.: for a video streaming application, in terms of sustainable frames per second at a given resolution; for a real-time fractal visualization application, in terms of the maximum delay for updating a fractal image with a given level of detail, as a consequence of the user changing some configuration parameter; etc.. In other words, the SaaS provider needs to cope with QoS requirements stated in the language of the application domain. Here we are faced with a most challenging problem to be tackled by the SaaS provider, the one of understanding what kind of low-level QoS requirements to specify to the IaaS provider(s) for meeting precise QoS constraints dictated at the application level.

3.3 The PaaS View

In order to overcome the above issues the PaaS provider of IRMOS acts as a mediator between the two parties (i.e. the SaaS and IaaS providers) by signing bipartite SLAs with them. The reasoning behind this approach is not only to minimize the complexity of the negotiated contracts, but most importantly to hide unneeded information from the different actors. The contract signed between the SaaS and the PaaS called Application SLA (A-SLA) contains high level application parameters that are understood by the customer. These are automatically translated to low-level hardware parameters that are used in the T-SLA signed between the PaaS and the IaaS.

Clearly, modelling, benchmarking, monitoring and workflow enactment play a fundamental role in this context, because they constitute the enabling factors allowing a PaaS provider to predict the performance that might be experienced by a virtualized application at run-time, despite the unknown factors internal to the IaaS provider(s) that are hidden as much as possible, as discussed above (see [4] for a discussion of how these concepts were applied for the IRMOS application scenarios). These services allow for the deployment of service components and whole applications into the available IaaS provider(s), and for gathering benchmarking data about the high-level QoS metrics exposed by the application (which constitute in turn the language by which the QoS may be precisely specified at the A-SLA level), as resulting from the different available configurations and parameters at the T-SLA level.

In addition to the benchmarking, a key role is played by evaluation and workflow management mechanisms that allow monitoring on-line the performance of virtualized software components, so as to compensate as soon as possible deviations of the application QoS metrics as compared to the values promised at the A-SLA level. This is realized through a distributed monitoring framework that is able to aggregate monitoring information coming from multiple sources and at different levels. These include monitoring information originating from the hardware and event notifications, as reported by the IaaS, as well as high-level QoS parameters reported by the application itself (SaaS side). The monitoring framework seamlessly interoperates with a QoS-aware workflow enactment engine allowing for on-the-fly changes to an application that is deployed as interconnected service components [24].

However, as already discussed, these business entities, including the PaaS provider, often have conflicting interests. With the IRMOS PaaS provider mediating between the SaaS and IaaS providers and being bounded to SLAs on both sides that allow for fault tolerance mechanisms to operate on both levels, there is a call for careful evaluation of the reported monitoring information.

To this direction, the evaluation system of the platform is able to assess on the origin of the application's performance deviation, i.e. whether it constitutes a breach of the application usage terms and if so whether the A-SLA specifies actions to be performed, whether it is an acceptable deviation that can be properly handled or an actual breach of the SLAs [25]. In the last case, further assessment is needed in order to conclude on the specific nature of the SLA breach, (T-SLA, A-SLA or both), i.e., to identify the actual entity or entities that failed to deliver the agreed QoS level.

4. Evaluation

The presented platform has been validated in public demonstrations, such as the European Commission’s ICT 2010 event[5], through three application scenarios namely Interactive eLearning, Virtual and Augmented Reality (VAR) and Film Post-production. Each of the three scenarios highlighted different aspects of the framework. The eLearning application stressed the automatic creation of A-SLAs through modelling and benchmarking [4]. The VAR scenario focused on the adherence of the platform to the QoS requirements and SLA re-negotiation, while the Film Post-production application showcased the reaction of the platform to SLA violations and actions needed to return to needed performance levels [24].

5. Conclusions and Future Work

The main areas that the IRMOS platform advances the state of the art in SLAs are: (i) requirements can be expressed in the language of the application domain, (ii) the user’s needs are dynamically translated to infrastructure requirements in fine grained SLAs, (iii) evaluation and mitigations mechanisms are able to quickly detect SLA violations and take necessary actions at run-time. Thus, it is evident, that the proposed framework adds to the already known benefits of cloud computing the possibility to execute interactive and resource-demanding applications with guaranteed QoS, that existing commercial offerings lack. Therefore service providers will be able to accommodate emerging future Internet applications that involve a broad class of interactive and collaborative tools and environments, including concurrent design and visualization in the engineering sector, media production in the creative industries, and multi-user virtual environments in education and gaming. Many of these applications tend to use dedicated hardware in order to achieve the desired Quality of Service (QoS), greatly increasing the overall cost for maintaining the needed resources. This can be a major hindrance to small businesses and

[5] ICT 2010. http://ec.europa.eu/information society/events/ict/2010/

startup companies that want to make innovative solutions available easily. Adopting a QoS-enabled Cloud solution alleviates this problem by providing the option of pay per use without the need to own expensive equipment.

Still, open research issues to be further investigated in this context include:

- how to formally specify QoS requirements at the SLA level in such a way that the offered and received QoS be verifiable by the interested parties (mainly the customer and the provider, but also the end user)? For example, what is a proper observation window and the exact conditions to observe, for claiming an SLA violation ?
- how to solve disputes between the parties signing a T-SLA or A-SLA, for example in the cases in which the customer claims that the promised QoS level he/she is paying for was not delivered ? By what ways can a provider certify its provided QoS levels, and the customer on the other hand certify its received ones?
- considering the complexity of delivering precise QoS levels at the A-SLA level, and the will of the provider to only offer guarantees on a probabilistic basis [5], what is a reasonable business model and SLA model that allows for a temporary graceful degradation of the delivered QoS with an associated pay-back penalty from the provider to the customer, or discount in cases of a pay-per-use model?

These and further challenging issues are to be investigated in further research.

References

[1] T. Voith, M. Kessler, K. Oberle, D. Lamp, A. Cuevas, P. Mandic, A. Reifert. ISONI Whitepaper v2.0. September 2008. Available on-line at: http://irmosproject.eu/Files/IRMOS_WP6_7_ISONI_White_Paper_ALUD_USTUTT_v2_0.pdf

[2] T. Cucinotta, G. Anastasi, F. Checconi, D. Faggioli, K. Kostanteli, A. Cuevas, D. Lamp, S. Berger, M. Stein, T. Voith, L. Fuerst, D. Golbourn, M. Muggeridge. IRMOS Deliverable D6.4.2 - Final Version of Realtime Architecture of Execution Environment. http://www.irmosproject.eu/Files/IRMOS_WP6_D6_4_2_SSSA_v1_0%5B1%5D.pdf.

[3] F. Checconi, T. Cucinotta, D. Faggioli, G. Lipari. Hierarchical Multiprocessor CPU Reservations for the Linux Kernel. In Proceedings of the 5th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT 2009). Dublin, Ireland. June 2009.

[4] T. Cucinotta, F. Checconi, G. Kousiouris, D. Kyriazis, T.Varvarigou, A. Mazzetti, Z. Zlatev, J. Papay, M. Boniface, S. Berger, D. Lamp, T. Voith, M. Stein. Virtualised e-Learning with Real-Time Guarantees on the IRMOS Platform. In Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2010). Perth, Australia. Dec. 2010.

[5] K. Konstanteli, T. Cucinotta, T. Varvarigou. Optimum Allocation of Distributed Service Workflows with Probabilistic Real-Time Guarantees. Springer Service Oriented Computing and Applications, Vol. 4, No. 4. Dec. 2010.

[6] B. Lin, P. Dinda, Vsched: Mixing batch and interactive virtual machines using periodic real-time scheduling, Proceedings of the IEEE/ACM Conference on Supercomputing, pp. 8, November 2005.

[7] C.L. Liu, J. W. Layland, Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, J. ACM, vol. 20,No. 1, January 1973, pp. 46—61, ACM, New York, NY, USA.

[8] F. Checconi, T. Cucinotta, D. Faggioli, G. Lipari, Hierarchical Multiprocessor CPU Reservations for the Linux Kernel, Proceedings of the 5th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications (OSPERT), June 2009, Dublin, Ireland.

[9] L. Abeni, G. Buttazzo, Integrating Multimedia Applications in Hard Real-Time Systems, Proceedings of the IEEE Real-Time Systems Symposium, 1998, Madrid, Spain.

[10] B. Lin, P. Dinda, Towards Scheduling Virtual Machines Based On Direct User Input, Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing, Washington,DC, November 2006, IEEE Computer Society.

[11] R. Nathuji, A. Kansal, A. Ghaffarkhah, Q-Clouds: Managing Performance Interference Effects for QoS-Aware Clouds, Proceedings of the 5th European conference on Computer systems (EuroSys), April 2010, Paris, France.

[12] J. W. Lee, K. Asanovic, METERG: Measurement-Based End-to-End Performance Estimation Technique in QoS-Capable Multiprocessors, Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 135-147, 2006.

[13] D. Gupta, L. Cherkasova, R. Gardner, A. Vahdat, Enforcing performance isolation across virtual machines in Xen, Proceedings of the ACM/IFIP/USENIX 2006 International Conference on

Middleware, Middleware '06, pp. 342—362, Melbourne, Australia, 2006, Springer-Verlag New York, Inc., New York, NY, USA.

[14] L. Cherkasova, D. Gupta, A. Vahdat, Comparison of the three CPU schedulers in Xen, SIGMETRICS Perform. Eval. Rev., Vol. 35, No. 2, pp. 42—51, September 2007, ACM, New York, NY, USA

[15] T. Cucinotta and G. Anastasi and L. Abeni, Respecting temporal constraints in virtualised services, Proceedings of the 2nd IEEE International Workshop on Real-Time Service-Oriented Architecture and Applications (RTSOAA), July 2009, Seattle, Washington.

[16] T. Cucinotta and D. Giani and D. Faggioli and F. Checconi, Providing Performance Guarantees to Virtual Machines using Real-Time Scheduling, in Proceedings of the 5th Workshop on Virtualization and High-Performance Cloud Computing (VHPC), Ischia (Naples), Italy, August 2010.

[17] B. Shirazi, L. Welch, B. Ravindran, C. Cavanaugh, B. Yanamula, R. Brucks, E. Huh, DynBench: A Dynamic Benchmark Suite for Distributed Real-Time Systems, Proceedings of IPDPS Workshop on Embedded HPC Systems and Applications, 1999, S. Juan, Puerto Rico.

[18] C. Germain-Renaud, C. Loomis, J. Moscicki, R. Texier, Scheduling for Responsive Grids, Journal of Grid Computing, Springer Netherlands, pp. 15-27, Vol. 6, No. 1, 2008.

[19] Z. He, C. Peng, A. Mok, A Performance Estimation Tool for Video Applications, roceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium, 2006, 267—276, IEEE Computer Society, Washington, DC, USA.

[20] G. Kousiouris and D. Kyriazis and K. Konstanteli and S. Gogouvitis and G. Katsaros and T. Varvarigou, A Service-Oriented Framework for GNU Octave-Based Performance Prediction, Proceedings of the 2010 IEEE International Conference on Services Computing (SCC), pp. 114-121, Miami, Florida, August 2010.

[21] D. Kyriazis, K. Tserpes, A. Menychtas, I. Sarantidis, T. Varvarigou, Service selection and workflow mapping for Grids: an approach exploiting quality-of-service information, Concurr. Comput. : Pract. Exper., Vol. 21, No. 6, April 2009, pp. 739—766, John Wiley and Sons Ltd., Chichester, UK.

[22] G. Dunlap, Scheduler development update, Xen Summit Asia 2009, Shanghai, November 2009.

[23] S. Benkner and G. Engelbrecht, A Generic QoS Infrastructure for Grid Web Services, Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICI), 2006

[24] S. Gogouvitis, K. Konstanteli, S. Waldschmidt, G. Kousiouris, G. Katsaros, A. Menychtas, D. Kyriazis, T. Varvarigou, Workflow management for soft real-time interactive applications in virtualized environments, Future Generation Computer Systems, 2011, DOI: 10.1016/j.future.2011.05.017.

[25] S. Gogouvitis, K. Konstanteli, D. Kyriazis, T. Varvarigou, An Architectural Approach for Event-Based Execution Management in Service Oriented Infrastructures, 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2010.