Service Level Agreements-Driven Management of Distributed Applications in Cloud Computing Environments

Alexandru-Florian Antonescu*§, Torsten Braun*
*University of Bern, Communication and Distributed Systems (CDS), Neubrückstrasse 10, 3012 Bern,
§SAP (Switzerland) Inc., Althardstrasse 80, 8105 Regensdorf, Switzerland
antonescu@iam.unibe.ch, braun@iam.unibe.ch

***Abstract*—Cloud Computing enables provisioning and distribution of highly scalable services in a reliable, on-demand and sustainable manner. However, objectives of managing enterprise distributed applications in cloud environments under Service Level Agreement (SLA) constraints lead to challenges for maintaining optimal resource control. Furthermore, conflicting objectives in management of cloud infrastructure and distributed applications might lead to violations of SLAs and inefficient use of hardware and software resources. This dissertation focusses on how SLAs can be used as an input to the cloud management system, increasing the efficiency of allocating resources, as well as that of infrastructure scaling. First, we present an extended SLA semantic model for modelling complex service-dependencies in distributed applications, and for enabling automated cloud-infrastructure management operations. Second, we describe a multi-objective VM allocation algorithm for optimised resource allocation in infrastructure clouds. Third, we describe a method of discovering relations between the performance indicators of services belonging to distributed applications and then using these relations for building scaling rules that a CMS can use for automated management of VMs. Fourth, we introduce two novel VM-scaling algorithms, which optimally scale systems composed of VMs, based on given SLA performance constraints. All presented research works were implemented and tested using enterprise distributed applications.**

I. Introduction

Recent advancements in cloud computing have enabled the proliferation of distributed applications, which require orchestration and control of multiple services running on virtualised containers inside network-connected cloud environments. However, without an efficient mechanism for scaling services in response to changing workload conditions, application performance might suffer, leading to Service Level Agreement (SLA) violations and/or inefficient use of hardware resources. Combining dynamic application requirements with the increased use of virtualised computing resources creates a highly challenging resource management situation for application and network and computing infrastructure owners.

In such complex environments, business entities use SLAs as a means for specifying quantitative and qualitative requirements of services. Given that SLAs have legal and financial implications, proper management of SLAs is critical for modern and future application and infrastructure service providers. Leading cloud service providers of infrastructure or application services (e.g. Google, Amazon) use SLA management for specifying and maintaining the quality of service (QoS) and availability levels to their customers. Dealing with exclusively-owned virtual machine (VM) instances deployed on a shared physical infrastructure presents a bigger challenge for each resource management lifecycle phase, given (1) the multi-objective resource allocation optimization problem of having to maintain SLAs for various customers while minimising resource utilisation targets, and (2) the differentiation in SLA-requirements from different classes of VMs and VM users. Furthermore, the violation of SLAs results in cash penalties for the cloud-infrastructure provider, adding a direct economic dimension to the problem.

We introduce the requirements and the architecture of a SLA-aware cloud management system for controlling the complexity of scaling applications composed of multiple services using mechanisms based on fulfilment of SLAs. For this, we provide the answers to the following research questions:

1) "How do distributed application owners and infrastructure providers benefit from using Service Level Agreements in Cloud Computing environments?"
2) "How to control infrastructure resource allocation in cloud environments while maximising both user QoS and provider Efficiency of Operations?"
3) "How to design an effective system for management (scaling) of SLA-constrained distributed cloud services?"

In order to answer these questions, we considered multiple connected problems from the perspective of both enterprise application owners and cloud infrastructure providers. The efficient management of SLAs is of particular importance for Cloud Computing, where exclusively-owned Virtual Machines (VMs) are allocated resources on hosts in a shared physical infrastructure.

The application owner is interested in finding the optimum amount of computing and network resources to ensure that the performance requirements of all applications are met. After the initial allocation of virtual computing resources, the application owner is also interested in appropriately scaling distributed applications so that application performance guarantees are maintained even under dynamic user-generated workload conditions.

Similarly, the infrastructure providers are interested in optimally allocating the virtual resources on the physical infrastructure resources so that its operational costs are minimised,

978-3-901882-76-0 @2015 IFIP

while maximising the tenant's application performance. In this context, we show how predicting the infrastructure utilisation peaks and dynamically optimising the distribution of virtual resources is a viable approach for optimising the use of available cloud resources, in comparison to simple or reactive (i.e. non-predictive) approaches.

This dissertation aims at developing a converging solution to the above mentioned management situations, by enabling the cloud resource management platforms to dynamically coordinate the various lifecycle actions on both the virtual and the physical cloud resources using semantically enriched SLAs. This work describes the architecture, models and control algorithms of such a platform for dynamic management of both distributed applications and virtual infrastructure computing resources.

First, we present the characteristics of distributed cloud-based applications with a focus on VM-bound services orchestration and application-level requirements representation. We introduce a semantic model for the representing cloud-SLAs. Based on this model we gather the requirements for a management system handling SLAs.

Next, we provide a detailed description of the multi-objective infrastructure resource allocation problem and various approaches to satisfying this problem. We present a system implementing a genetic algorithm for performing virtual resource allocation, while considering the optimization of multiple criteria. We describe the system's functional blocks and the interactions between these components.

Following this, we present the management requirements for handling distributed applications, with regards to runtime dynamic sizing of virtual infrastructures composed of application services running inside VMs. We introduce mathematical formalisms and algorithms (e.g. statistical data processing, machine learning) behind the mechanisms for dynamically scaling of distributed services. We also present a framework for dynamic discovery of distributed service scaling rules using application benchmark mechanisms. We then show how these scaling rules can be combined and included into semantic SLA for controlling services allocation. We then describe the mechanisms for detecting the application utilisation patterns and show how these could be used for optimising the reservation of virtual resources in cloud environments.

We evaluate the benefits of using SLAs for cloud services scaling and infrastructure resources allocation, both by running experiments on a distributed test-bed and by simulations. We describe the experiments with scaling and allocation of VM-bounded distributed services, by using a Distributed Enterprise Information System (dEIS) case study. We present the results of using our service management platform combined with the virtual resources allocation platform and with the semantic SLAs against service management in the absence of these advanced management capabilities. We also present the system's stability results by using a benchmark generated SLA for dEIS scaling management while under SLA performance constraints. We conclude the evaluation by describing a SLA-based large-scale VM scaling simulation based on a model of the dEIS distributed system.

Overall, the key contributions of this dissertation[1] are:

1) a semantic SLA-enabled specification language and architecture for dynamically managing distributed software and computing, storage and network infrastructure services [1]
2) a framework for dynamically allocating VMs to physical resources while considering SLA-constraints and multiple objectives optimisations by using a genetic algorithm combined with data forecasting (exponential smoothing) [2]
3) a method of sizing virtual infrastructures based of SLA-defined constraints [3], and composing scaling rules for distributed services using prediction mechanisms and correlation-derived relationships between SLA monitoring metrics [4]
4) two SLA-based VM scaling algorithms that use reactive and analytic mechanisms combined with data prediction methods and results from applying Little's Law ([5], [6], [7])
5) implementation of the previous mentioned techniques and tools for SLA-aware management of cloud infrastructure and software resources in the Service Middleware Layer (SML) component of the GEYSERS FP7 EU research project ([8], [9], [10])

II. SLA-Driven Management of Cloud-Distributed Applications

Cloud Computing [11] has evolved to become an enabler for delivering access to large-scale distributed applications[12] running inside managed environments composed of network-connected computing systems, under strict performance and quality of service requirements, defined using SLAs.

SLAs [13] are contracts defining the performance and quality of service (QoS) boundaries of distributed applications. A cloud management system (CMS) enforces SLAs by dynamically allocating available computing resources to cloud services. A CMS monitors both the software cloud resources as well as the underlying physical network and computing resources. It then uses this information for deciding the actions to be taken, such as increasing the number of VMs (scaling-out/up), decreasing (scaling-in/down), or migrating software components in order to maintain the conditions defined in the SLAs and for maximising provider-specific metrics (e.g. energy efficiency).

We define the research question as: *"How can a CMS dynamically scale the number of VMs allocated to cloud services, so that the SLA-defined performance constraints are maintained under variable workload conditions such as fluctuating number of users?"*.

We define the SLA Cloud Management Optimization (SLA-CMO) problem ([1] [2] [3]) as: improving the efficiency of using datacenter's computing resources by dynamically allocating and then adapting the number of VMs associated with cloud services, so that SLA-defined performance requirements are met under variable workload conditions. Solving the SLA-CMO problem depends directly on the following aspects: (1) having machine-readable representations of the

[1] http://www.iam.unibe.ch/~antonescu/phd.pdf

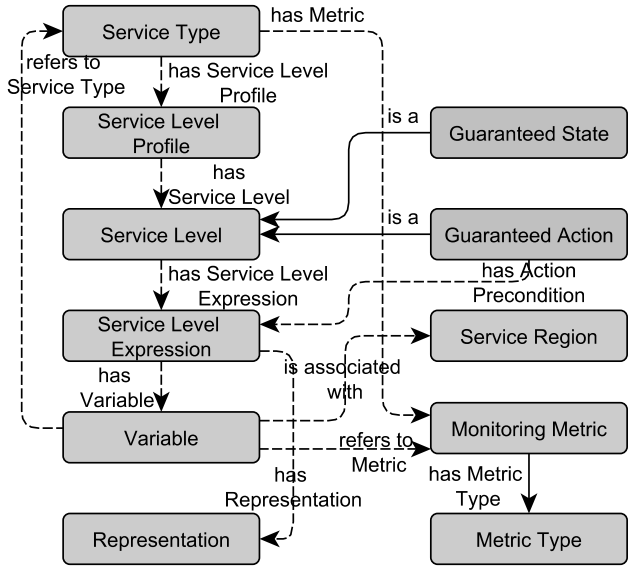

Fig. 1. Extended USDL-SLA Ontology

distributed services, and relations between the services' monitoring metrics and SLA-defined guarantees and conditions; (2) having reliable sources of monitoring information reflecting the state (e.g. number of allocated VMs, system's throughput, requests arrival rate, response time) of the managed distributed systems, which are then used by SLA-aware CMS-components for managing the VMs and cloud resources; (3) being able to optimally choose the amount of cloud infrastructure resources (e.g. computing, network bandwidth, and storage) allocated to cloud services; (4) acting timely on potential condition that might lead to SLA violations by scaling the number of VMs.

In the following subsections we present our proposed solutions to each of the four mentioned SLA-CMO subproblems.

A. Semantic SLA-Enabled Framework for Dynamic Management of Distributed Applications

In order for a CMS to be able to autonomously manage virtual infrastructures composed of services running across multiple VMs, it must be provided with a machine-readable representation of the application services, their requirements in terms of cloud resources, the relations between them, and the associated SLA monitoring metrics. For satisfying these conditions, we describe a semantic model for representing application topologies combined with SLA conditions and guaranteed CMS-actions [1] .

Figure 1 presents the model used for defining the SLAs in relation to the monitoring metric associated with the application services. It was built using RDF [14] by extending the USDL-SLA [15] vocabulary. The model defines SLA guaranteed states and actions (management rules), as well as the conditions required for automatic execution of the actions, which enable the CMS to autonomously take action on certain conditions potentially leading to SLA violations.

Once the semantic SLA model has been created by application topology architects, it will be given as input to the CMS

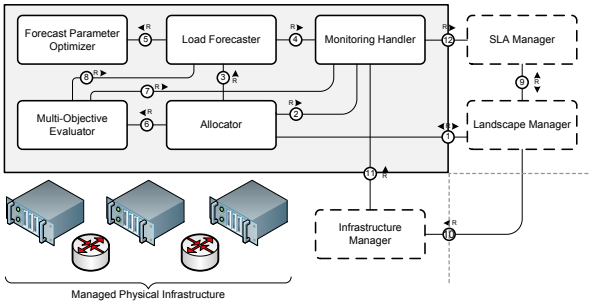

Fig. 2. CMS architecture

[1]. The CMS will perform the dynamic topology orchestration process, by performing five stages. (1) It will analyse the services contained in the SLA, the relations between them and their requirements in terms of cloud resources (computing, network bandwidth and storage), followed by the scheduling of appropriate infrastructure operations (e.g. VM images preparation, VM instantiation, network provisioning). (2) The cloud infrastructure will be prepared by reserving the corresponding amount of resources (e.g. network, storage, computing). (3) Application services will be deployed, followed by activation of probes for monitoring the services' performance. (4) At this stage the distributed application is already running and its services will be monitored, with the data being fed to the CMS for evaluating the SLA guarantees and action triggers. (5) As various SLA actions are triggered, the CMS executes in response the SLA-defined operations for ensuring that the distributed system remains in the SLA-defined performance boundaries. Possible CMS-operations are: scale-out - adding VM instances; scale-in - terminating VM instances, scale up - increasing resources (e.g. CPU, memory) allocated to a VM; scale-down - decreasing the amount of resources allocated to a VM.

B. Multi-Objective SLA-Aware Allocator of VMs in Cloud Environments

It is often the case that at datacenter level the CMS must solve a complex multi-objective optimization resource allocation problem. The physical infrastructure providers must deliver and maximise the SLA-advertised levels of performance and capacity availability, while minimising energy consumption, resource wastage and costs created by violating the agreed SLAs.

In order to solve this problem, we describe a CMS architecture using a VM allocator based on a multi-objective genetic algorithm, and predictive mechanisms [2]. The system allocates VMs to physical hosts while considering the effect (e.g. penalties) of the existing SLAs and using historical monitoring data to forecast the incoming load on both the physical and virtual infrastructure, in order to (1) maximise SLA satisfaction, (2) minimise penalties, and (3) minimise energy consumption at datacenter level.

Figure 2 shows the CMS's control loop responsible for the allocation of VMs and for its dynamic optimization of based on the SLAs and on the forecasted level of resources' utilisation. The main component is the multi-objective VMs allocator.

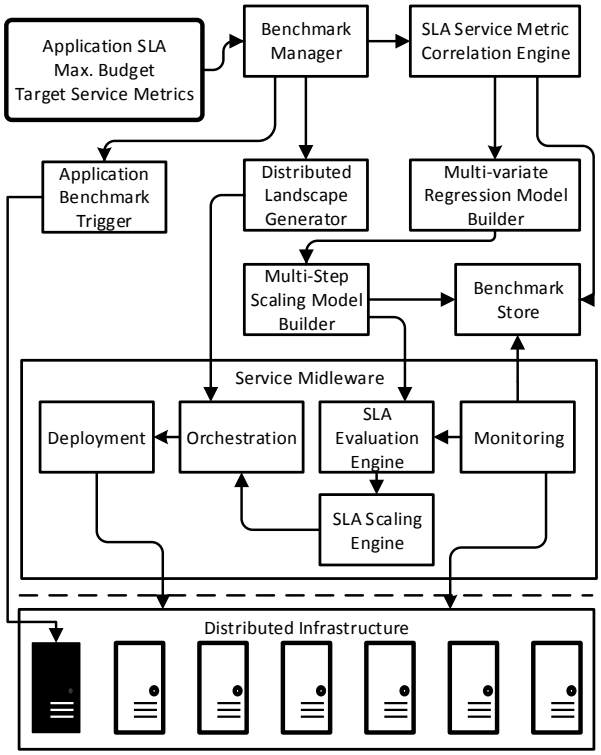

Fig. 3. DynSLAOp System Architecture

Given that the problem on allocating VMs to physical hosts is equivalent to the NP-hard bin-packing problem, we decided on using a genetic algorithm (GA) [16] for solving it. Also, given the fact that the VMs are naturally grouped by their hosts, we selected a group-oriented GA [17] for finding a solution to the VMs allocation problem.

The proposed GA works by encoding VMs as genes, hosts as groups and the entire datacenter as a single chromosome. The genetic operator *mutation* works by randomly deleting a gene (VM), followed by its re-insertion using a heuristic algorithm (e.g. first-fit), ensuring that no VMs are simply eliminated. The genetic *crossover* operator works by selecting from each of the two chromosomes the groups with the highest fitness value. As this might lead of omission of some VMs from the resulting chromosome, they will be reinserted using the same heuristics as in the case of the mutation operator.

Once a population has been generated, it will be ranked, then a elite number of chromosomes will be transferred to the next generation, and finally mutation and crossover will be applied with different probabilities. This process will be repeated until the stop criterion will be reached.

For calculating the fitness of a chromosome we considered four objectives in for the allocation of VMs: maximising the total revenues, minimising the datacenter's energy costs, minimising the costs associated with VM migrations, and minimising SLA penalty costs. These different objectives are combined in an aggregate objective function by assigning weights to each of the four mentioned costs.

As three of the objectives (energy consumption, migration time and CPU oversubscription penalty) depend on the VM's

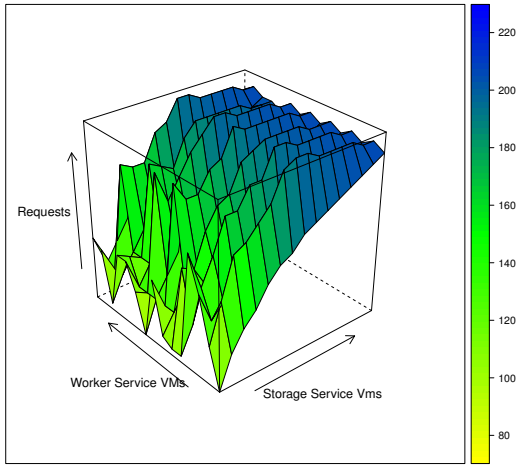

Fig. 4. System's request-processing capacity vs. number of Worker and Storage VMs

CPU utilisation level, we predicted the value of CPU utilisation by using Holt-Winters triple exponential smoothing [18].

C. SLA-Based Virtual Infrastructure Sizing and Dynamic Composition of VM-Scaling Rules

Dynamically changing (scale out/in) the number of VMs allocated to a distributed services is one of the main mechanisms used in cloud environments for adapting the performance of applications to variation in workload conditions. In [3] we proposed using a dynamically generated scaling model for the VMs associated to services of distributed applications, for enabling a CMS to scale VMs as a reaction to variations in the number of application users, so that it can maintain SLA-defined performance guarantees. We answered the following research question: "How to dynamically decide how many services instances are needed in order to handle a larger workload within the same time constraints?".

Figure 3 presents the architecture of the Dynamic SLA Optimiser (DynSLAOp) system, which receives as input a semantic-SLA specification for a distributed application, together with an upper limit for the application's performance. DynSLAOp instantiates the required cloud resources (VMs, network connections) and then it begins benchmarking the distributed application with an increasing number of application users until the SLA-defined performance bound is reached. The monitoring information about service instances is recorded for further processing. After completing the current benchmark, the number of VMs is increased and the benchmark is repeated. This process is performed until the user-defined maximum number of VMs is reached.

Figure 4 shows the evolution of the combined processing capacity (PrC) of the dEIS system when increasing the number of worker (*WK*) and storage (*ST*) VMs. We noticed that the system's total PrC rises faster when increasing the number of *WK* VMs (at a constant number of *ST* VMs), due to the nature of the dEIS application. Also, as we used only one VM for generating the user requests, this limited the maximum number of responses received in the defined time interval - shown as the asymptotic convergence of the number of total requests.

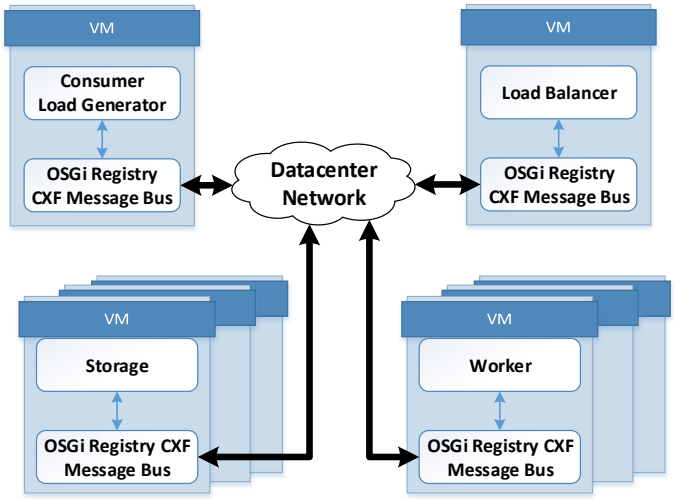

Fig. 5. Distributed Enterprise Information System

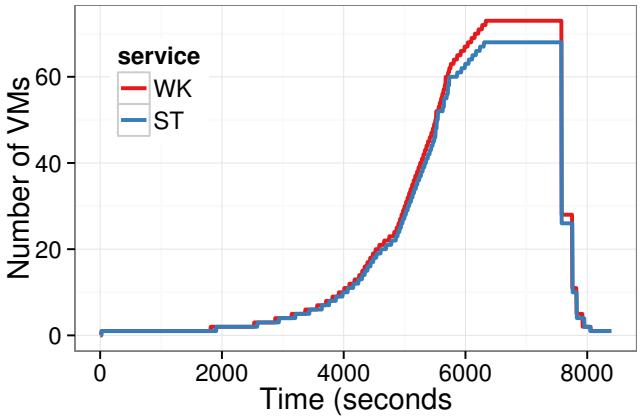

Fig. 6. SLA-based VM-scaling simulation using system's rate of arrival

After all benchmarks have been executed, DynSLAOp creates a scaling "path" (ScP) for supporting a monotonically increasing number of application users. One ScP is a collection of tuples $(n_{S_1}, n_{S_2}, ..., u)$, where n_{Sx} is the number of VM instances allocated to service S_x and u is the number of application users. The monitoring information corresponding to each ScP-tuple is then consolidated into a large dataset, with only the data entries corresponding to the selected number of users (u). The dataset is then analysed for determining correlations between the critical performance metric (given to the system) and the application's monitoring SLA-metrics. The set of correlated parameters is then used for building a multi-variable regression model (MVRM) for estimating the number of VMs (n_{S_x}) of each service (S_x) in ScP [3].

The MVEM is finally converted into a SLA scaling rule. The CMS continuously compares the MVEM's produced value with the actual number of VMs in the system and performs either scale-out or scale-in in order to keep the number of VMs synchronised.

D. Predictive SLA-Aware Scaling of Distributed Services in Cloud Environments

In order to validate our assumptions related to SLA-constrained VM-scaling of distributed cloud applications we set to perform a large-scale simulation of an enterprise application. For this purpose, we first described a method of profiling a distributed application and building a simulation model for it [6]. We did this by first deploying exactly one VM instance of each service of the distributed application, followed by running multiple benchmarks with a constant number of application users. This allowed us to record the CPU, memory, network I/O, storage I/O utilisation at the same time with application level metrics, under constant workload conditions. All the monitoring datasets were then consolidated in a large dataset, where each entry represented a distributed transaction across the enterprise application, tagged with the number of concurrent transactions executed by the system.

In the next step we converted this consolidated dataset into a simulation model by converting the measured execution durations into MIPS (million instructions per second) units. This allowed us to adapt the consolidated model of the enterprise application to CloudSim [19] simulator. Our validation experiments and analytical testing (using Kolmogorov-Smirnov test and the linear dependency coefficient) showed that the simulation model accurately modelled the behaviour of the real enterprise application.

Having an accurate model of a real enterprise distributed application, we tested whether a reactive SLA-based VM-scaling algorithm could be used for managing our distributed enterprise application [5]. We also evaluated the ability of CloudSim to simulate multiple cloud tenants. We have successfully simulated two cloud tenants with and without VM-scaling with a dynamic, slowly changing, workload. This validated the use of CloudSim for multi-tenant, large-scale cloud simulations.

We described two novel SLA-based VM-scaling algorithms, which use the SLA-defined maximum value for the average execution time for determining the maximum processing throughput of each service [7]. The optimum throughput (Th_{max}) is calculated by applying Little's Law [20] to the consolidated dataset described in [6]. The first VM-scaling algorithm works by assuming that the system operates at equilibrium, where the system's throughput (Th) is (almost) equal to the workload's rate of arrival (λ). The VM-scaling algorithm then calculates the number of VMs (vm^*) that the system should have so that under uniform load-balancing, each VM will process at most 80% of the Th_{max}. If wm^* is higher than the current number of running VMs (vm) plus the number of VMs created by not yet started (vm^+), then the CMS will create an additional number of VMs, equal to $vm^* - vm - vm^+$.

The second VM-scaling algorithm from [7] built on the previously described algorithm by using prediction to forecast the workload's rate of arrival (λ), and then it used the predicted-λ for determining the expected number of VMs for a time horizon of a few minutes. Knowing the average VM's instantiation duration allowed the CMS to anticipate the VMs-scale-out operations and to execute the scale-out before the actual scaling conditions were met, ensuring that the distributed system was always in a SLA-compliant state.

III. Results and Evaluation

In order to evaluate our proposed algorithms and techniques, we took a dual approach, in which we ran VM allocation and VM-scaling experiments both on real-word deployments ([3], [4], [10]), and using simulation models ([1], [2], [5], [6]) based on real-world monitoring traces coming from an actual distributed enterprise application.

A. Distributed Enterprise Information System

We briefly introduce the application under test, which we used throughput ([1], [3], [4], [6], [7], [10]). Figure 5 presents the four core services composing the distributed enterprise application (dEIS): one or more Thin Clients (CS), a Load Balancer (LB), one or more Worker services (WK), and one or more Database Storage services (ST). Each service runs in its own VM and communicates asynchronously with the other services using a distributed service messaging bus. The dEIS's components are presented in [1], and dEIS's performance characteristics are described in detail in [6].

Important characteristics of dEIS application are (1) its ability to increase the system's processing capacity with the number of VMs allocated to WK and ST services, as we show in [3] and [7], and (2) its functioning under uniform load-balancing conditions. This ensured that the application was well suited for running horizontal-scaling experiments, and thus supporting the evaluation of our proposed VM-allocation and VM-scaling algorithms.

B. Contributions and Results

The SLA semantic model [1] described in Section II-A was used in the implementation of the Service Middleware Layer (SML) cloud infrastructure management platform ([8], [9], [10], [21]). We evaluated the model's capacity to represent complex service level objectives (SLOs), as well as the relations between cloud services, and the conditions for CMS-automated execution of cloud-infrastructure operations.

The multi-objective VM-allocator algorithm described in Section II-B was evaluated by means of a large-scale simulation, in which we considered the impact of VM migrations and CPU oversubscription on VM allocation operations. Our results [2] have shown that using a multi-objective VM allocator based on a group-oriented genetic algorithm, combined with resource-utilisation prediction mechanisms can lower the costs associated with CPU and network oversubscription, when compared with well-known heuristics-based allocation algorithms.

The methods and algorithms for dynamic composition of service-scaling rules described in Section II-C were evaluated by scaling the services of dEIS system as described in [3]. The VM-scaling is briefly presented in Figure 4 for a single requests-generator VM. These experiments enabled creating an analytic model of the dEIS's performance, which we then used for the large-scale simulation described in Section II-D.

We evaluated VM-scaling algorithms from Section II-D with a real-world workload (described in [7]), which varied the number of users from 1 to 25'000 during approximately four hours. The CMS created approximately 150 VMs (briefly shown in Figure 6), validating the stability and effectiveness of the VM-scaling algorithms, as they were able to maintain the average execution time below the maximum value defined in the SLA.

IV. Conclusions

The main contribution of this dissertation is on the management of scalable distributed enterprise applications. We summarise as follows the researched set of novel techniques and tools for datacenter-infrastructure resource allocation, VM-scaling, and management of cloud-services.

For cloud-deployed multi-tier applications it is often the case that the number of allocated infrastructure resources have to be dynamically changed. This dissertation proposes a novel semantic SLA model for supporting (1) the description of complex, interconnected distributed applications, their monitoring metrics, infrastructure management operation, and SLOs; (2) the automated execution of infrastructure-scaling operations based on SLA-defined triggers.

Convergent management of cloud infrastructure resources (network, computing, storage) requires considering multiple objectives, such as (1) minimising the costs associated with violating SLAs due to oversubscription of network and computing resources, (2) maximising the tenants' quality of service, and (3) minimising energy consumption. This dissertation proposes a novel resource allocation algorithm for virtual machines based on a genetic algorithm, which combines uses prediction mechanisms for optimally allocating VMs, while satisfying multiple objectives concerning the efficiency of the management operations.

Multi-tier distributed applications have complex inter-dependencies between their services, making design of scaling rules a complicated process. This dissertation proposes using benchmarks for exploring the dependencies between the number of allocates services of a distributed application and its maximum processing capacity. We present (1) a method of using correlations for discovering the key parameters that determine the number of required service instances of a distributed application; and (2) an algorithm for composing SLA-based VM-scaling rules based on the benchmark-obtained datasets.

Different cloud applications have varied requirements regarding the value of quality of service metrics, making automated scaling of such systems a complicated operation. This dissertation proposes different VM-scaling algorithms, which are able to incorporate SLA-derived performance requirements for (1) finding the maximum per-VM processing capacity of each distributed service, (2) and then use this for optimally-scaling the number of VMs such that, under varying workload conditions, the level of performance experienced by the application's users remains in the bounds defined by the SLAs.

Acknowledgment

This research has been (partly) performed during 2011-2013 in the GEYSERS EU FP7 research project.

REFERENCES

[1] A.-F. Antonescu, P. Robinson, and T. Braun, "Dynamic topology orchestration for distributed cloud-based applications," in *Proc. 2nd IEEE Symposium on Network Cloud Computing and Applications (NCCA)*, 2012.

[2] ——, "Dynamic SLA management with forecasting using multi-objective optimizations," in *Proc. 13th IFIP/IEEE Symposium on Integrated Network Management (IM)*, May 2013.

[3] A.-F. Antonescu, A.-M. Oprescu *et al.*, "Dynamic optimization of SLA-based services scaling rules," in *Proc. 5th IEEE Internetional Conference on Cloud Computing Technology and Science (CloudCom)*, December 2013.

[4] A.-F. Antonescu and T. Braun, "Improving management of distributed services using correlations and predictions in SLA-driven cloud computing systems," in *Proc. IEEE/IFIP Network Operations and Management Symposium (NOMS)*, May 2014.

[5] ——, "SLA-driven simulation of multi-tenant scalable cloud-distributed enterprise information systems," in *ACM PODC Workshop on Adaptive Resource Management and Scheduling for Cloud Computing (ARMS-CC)*, 2014.

[6] ——, "Modeling and simulation of concurrent workload processing in cloud-distributed enterprise information systems," in *ACM SIGCOMM Workshop on Distributed Cloud Computing (DCC 2014)*, August 2014.

[7] ——, "Simulation of SLA-based VM-scaling algorithms for cloud-distributed applications," *Future Generation Computer Systems (FGCS)*, 2015.

[8] A.-F. Antonescu and P. Robinson, "Towards cross stratum SLA management with the geysers architecture," in *Proc. 10th IEEE Int. Symposium on Parallel and Distributed Processing with Applications (ISPA)*, 2012.

[9] E. Escalona *et al.*, "Geysers: A novel architecture for virtualization and co-provisioning of dynamic optical networks and it services," in *Future Network & Mobile Summit (FutureNetw)*. IEEE, 2011, pp. 1–8.

[10] J. F. Riera, J. Garcia-Espin, A.-F. Antonescu *et al.*, "Virtual infrastructures as a service enabling converged optical networks and data centres," *Optical Switching and Networking (OSN)*, pp. 197–208, 2014.

[11] P. Mell and T. Grance, "The NIST definition of cloud computing," *NIST special publication*, 2011.

[12] D. Woods, *Enterprise Services: Architecture*. O'Reilly Media, Inc., 2003.

[13] P. Wieder, *Service level agreements for cloud computing*. Springer, 2011.

[14] G. Klyne and J. J. Carroll, "Resource description framework (RDF): Concepts and abstract syntax," http://www.w3.org/TR/2004/REC-rdf-concepts-20040210, 2004.

[15] Leidig, T. and C. Momm, "USDL service level agreement," http://www.linked-usdl.org/ns/usdl-sla, April 2012.

[16] T. Bäck, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, USA, 1996.

[17] E. Falkenauer, "A hybrid grouping genetic algorithm for bin packing," *Journal of heuristics*, vol. 2, no. 1, p. 5Äì30, 1996.

[18] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *International Journal of Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.

[19] G. Lab, "Cloud simulator cloudsim," http://code.google.com/p/cloudsim, 2014.

[20] J. D. Little and S. C. Graves, "Little's law," in *Building Intuition: Insights From Basic Operations Management Models and Principles*. Springer, 2008, pp. 81–100.

[21] A.-F. Antonescu, P. Robinson, and M. Thoma, "Service level management convergence for future network enterprise platforms," in *Future Network & Mobile Summit (FutureNetw)*, 2012, pp. 1–9.