Towards A Meta-Model of the Cloud Computing Resource Landscape *

Kleopatra Chatziprimou[1], Kevin Lano[1], Steffen Zschaler[1]
[1]*Department of Informatics, King's College London, UK*
{ *kleopatra.chatziprimou, kevin.lano, steffen.zschaler*}*@kcl.ac.uk*

Keywords: Cloud Computing; Meta-model; Resources

Abstract: As Cloud Computing becomes more predominant, large scale datacenters are subject to an increasing demand for efficiency and flexibility. However, growing infrastructure management complexity and maintenance costs are becoming a hindrance to the advancement of the Cloud vision. In this paper we discuss how existing datacenter resource management approaches fail to provide infrastructure elasticity and suggest a resources provisioning architecture to fill this gap. As a first step towards implementing our targets, we present a meta-model to describe the characteristics of the Cloud landscape, emphasising on a provider's perspective. With this meta-model we intend to introduce new modelling concepts towards facilitating the selection of optimal reconfigurations in a timely fashion.

1 INTRODUCTION

The advent of Cloud Computing has brought datacenter design at evolutionary crossroads. However, massive data growth, challenging economic conditions, and physical limitations of power, heat and space are exerting pressure on the enterprise. Finding architectures that can minimize cost, complexity and associated risks in the Cloud has become a priority (Josyula, 2012).

Inside a Cloud, workload is balanced among hundreds of Virtual Machines (VMs). Each VM hosts one or more applications and VM sizes are decided according to load, while VMs are packed to Physical Machines (PMs) (i.e. actual hardware servers) employing live VM migrations. The mapping of VMs to PMs form the current configuration. During operation, the Cloud runtime execution context continuously evolves, as workload demand fluctuates according to unstable user-preferences, demand spikes (i.e., unexpected increases in workload volume) and/or data spikes (i.e., change in the distribution of particular objects' popularity) can be noticed. In addition, system attacks or unexpected failures might occur. The volatility of the runtime context poses the need for reconfigurations in order to preserve the system's functionality and QoS targets.

To highlight the criticality of the aforementioned, consider the case of a news company such as Guardian, which offers public facing internet-scale services to millions of users. In the case of an unexpected "snap" article (e.g., terrorist attack) the company's infrastructure will have to temporarily serve a massive amount of user queries, without compromising the expected QoS level. Questions as the following arise: How many new resources need to be added to the system to meet the SLAs? How will the reconfigurations affect the operational expenditures (OPEX)? Answering these questions involves time consuming, complex procedures that cannot be performed in a timely fashion by human operators. Will the system manage to provide up-to-date response to the crisis?

Existing resource provisioning solutions seem to focus on technology silos and not platforms as a whole. Hence, they cannot sufficiently handle mission critical workloads, withstand variable demand or widely control operational costs. Towards filling this gap, we propose an interoperable infrastructure management layer to optimize the resources (re)configurations complexity within a reasonable cost and time-range during datacenter operation. To this end, we suggest the combination of Model Driven Engineering (MDE) tools with dynamic multi-objective optimization and policy-driven design. Expected outcomes of this combination include dynamic selection of the Cloud optimization objectives according to the runtime context, as well as automation of the transitions between problematic Cloud states and corresponding reconfiguration strategies.

*This work has been funded by the European Commission under FP7 ITN RELATE.

As a first step towards implementing our solution, we present a meta-model that specifies the core concepts of the Cloud space. First, the proposed meta-model aligns the notion of an enterprise Cloud with its expected revenue. Second, it comprises information regarding both physical and virtual Cloud resources as well as the mapping of the running services between them. Finally, it models the Cloud points of variability i.e., the aspects of the Cloud architecture that can be reconfigured at run-time. The meta-model serves as a basis to support on-line performance analysis, automated extraction of valid reconfigurations and run-time validations towards achieving optimal adaptations.

The rest of the paper is organised as follows: In Section 2 we discuss the literature in the field of resources management in dynamic datacenters and Clouds, Section 3 introduces our envisioned approach, Section 4 presents the meta-model, Section 5 demonstrates a use case and finally Section 6 concludes the paper.

2 RELATED WORK

In the current literature, systems are continuously monitored and accordingly adapt their resource configurations, to comply with volatile user requirements and preserve end-to-end QoS. Particularly, research in the resource provisioning field span the following directions: (a) Operational Costs Awareness, (b) Network/Communication Overheads Awareness and (c) Applications Affinities Awareness.

Operational Costs Awareness A plethora of solutions has been proposed to alleviate the costs of operating and maintaining a datacenter. Bin packing heuristics have been widely explored to densely pack overloaded VMs to underutilized PMs (Khanna et al., 2006). The problem of minimization of eager VM migrations is considered crucial to achieve stability inside the datacenter (Ferreto et al., 2011). (Kusic et al., 2008) approached dynamic consolidation by building estimates for future workloads incorporating Limited Lookahead Control, and feed them to a utility maximization problem. Other utility based approaches are presented in (David Breitgand, 2011; Chuen et al., 2011). Costs optimization is attempted through Constraint Programming, deduction to Generalized Assignment Problem (GAP) or Genetic Algorithms. Reinforcement Learning (RL) is used in (Tesauro et al., 2006) to estimate rewards of different reconfiguration actions. Feedback control theory has been suggested to manipulate server utilization (Abdelzaher et al., 2001).

Network/Communication Overheads Awareness Attention has been given to the control of network bandwidth consumption inside a datacenter. Stage and Setzer (Stage and Setzer, 2009) aim to minimize the risk of overloading the system's links by predicting workloads and network utilization rates. (Piao and Yan, 2010) suggests a communication-overhead minimization solution that places VMs to PMs with smallest data transfer times. Meng et al. (Meng et al., 2010b) indicate the need for traffic-aware VM placement to improve scalability.

Applications Affinities Awareness Considering dependencies among VMs (e.g. memory sharing possibility) and their hosted applications could be beneficial to configuration planning. Appaware (Shrivastava et al., 2011) e.g., accepts as input a VMs dependency Graph and exploits VM communication patterns to minimize the overall placement cost. Meng et al. leverage multiplexing to identify compatible VMs for being jointly provisioned so as to achieve capacity savings (Meng et al., 2010a).

2.1 Discussion

As increased expenditures and complexity in datacenter resources management remain open issues, we note the limitations of existing approaches. RL and Control Theoretic frameworks suffer from poor scalability, being inappropriate for Cloud applications. Utility based solutions aim to provide local or global optimizations. The first category achieves faster response times though it can be inaccurate, while the latter faces feasibility issues due to computational costs.

Utility based approaches can be further classified into proactive or reactive. Proactive are based on resource utilization predictions to keep a system running. However, predictions include intrinsic prediction errors and introduce inaccuracies as well as complexity. Reactive schemes achieve their objectives by recovering from failures, which might be risky for the anticipated Cloud agility. The biggest concern with objective functions, is that they remain rigid to a set of initial objectives and system constraints.

However, due to volatile granularities of human (i.e., Cloud users, Cloud providers) requirements, unpredictable events occurrences, complex interactions between workloads and shared infrastructure, the datacenter state continuously evolves. Simultaneously, multiple, (often) conflicting goals need to be accomplished in order to deliver a competent solution targeted to modern enterprises e.g, resource wastage, power consumption, network communication overheads and others. Additionally, the duration of adaptations decisioning and implementation, is strictly bounded by a time-sensitive window after which the target configuration becomes stale. Hence, the system optimization objectives may vary from time to time and the optima have to be found in time, posing new challenges for the design of robust resource provisioning solutions for Clouds.

3 ENVISIONED APPROACH

As the number of possible configurations grow exponentially with the number of optimization objectives mentioned above, research community faces the issues of complexity and uncertainty. We aim to tame these burdens combining Model Driven Engineering (MDE) tools with multi-objective optimization strategies as Figure 1 shows. We assume a diagnostics

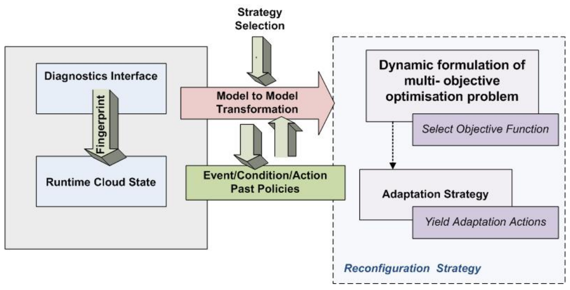
Strategy Selection
Diagnostics Interface
Fingerprint
Runtime Cloud State
Model to Model Transformation
Event/Condition/Action Past Policies
Dynamic formulation of multi- objective optimisation problem
Select Objective Function
Adaptation Strategy
Yield Adaptation Actions
Reconfiguration Strategy

Figure 1: Overview of Envisioned Solution.

interface to fingerprint the Cloud runtime states and provide automated identification of performance crisis or changes in Cloud Provider's optimisation goals. For example, in scenario *a* could be identified that the provider focuses on pursuing green environmental policies minimizing power electricity and thermal dissipation costs, in scenario *b* that the provider prioritises to greedily increase its profit offering IaaS services to the maximum possible number of clients while in scenario *c* that the provider has to quickly overcome a network attack overloading the system links.

Management automation enables operational efficiencies related to change of control. Therefore, we intent to automate the transition from diagnosed Cloud states to appropriate reconfiguration strategies exploring model transformations and constraints (enforcing system viability). Each reconfiguration strategy entails the dynamic formulation of a multi-objective Optimization Problem, specifying an objective function to capture only the most critical current Cloud optimisation goals (e.g., minimization of resources wastage, control of power consumption etc.) in order to limit the possible solutions space and support timely reconfigurations. The purpose of enabling the dynamic formulation of multiple possible objective functions is to allow expression of preference for optimal feasible system states from an ideal state, in the Cloud's valid states-space. The update of the objective function (i.e., replacement of objectives) will be aligned with the runtime evolution of the Cloud.

Thereafter, the definition of an adaptation strategy within the reconfiguration, provide means to solve the specified multi-objective optimization problem, and can encompass -according to scenario- solutions already existing in the literature, promoting interoperability. Finally, successful transformations will be cached in the system as customisable Event/Condition/Actions (ECA) to facilitate future reconfiguration decisions. This policy-driven design will help reduce maintenance complexity and costs through the ability to reuse. Moreover, the low level configuration syntax is abstracted, enabling higher level management tools to require knowledge only about the policy semantics.

As a first step towards realizing the aforementioned, we acknowledge that an abstract meta-model is needed in the Cloud resources landscape. There exist different approaches on modelling resources and the environment where they are contained (Becker et al., 2009; Huber et al., 2012). In general such approaches are either targeted for design time analysis or do not suffice to cover the whole extent of a Cloud's topology.

The purpose of the meta-model is to support automated extraction of the possible valid Cloud states, and capture properties relevant to performance analysis in order to enable runtime reasoning of the configurations. Hence, the meta-model comprises information about: (a) the actual capabilities and limitations of the system; (b) trade-offs developed inside an operating system which are unknown at design time (i.e., VM communication patterns); (c) points of variability (i.e., dynamic aspects of the Cloud infrastructure that can be adapted at run-time) (d) expression of the system's runtime optimization targets.

We have focused on a high-level meta-model of the Cloud landscape and in particular information needed to describe feasible infrastructure variability points, in terms of physical hosts and virtual machines. Detailed specification of aspects as storage, network, or computing requirements will need to be added to this meta-model for application to real scenarios. However, we omit them in this paper due to lack of space.

4 THE CLOUD META-MODEL

In this section we present the Cloud landscape meta-model. As shown in Figure 2 the meta-model was designed utilizing the Eclipse Modelling Framework (EMF) [2].

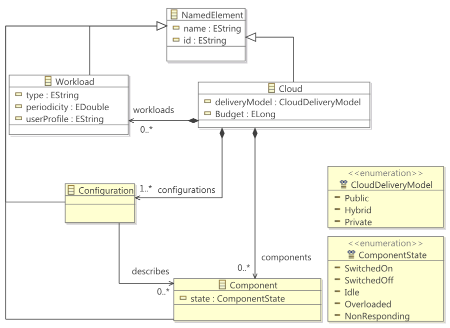

Figure 2: Cloud System Metamodel.

The root entity comprising all the other entities is the *Cloud.* The *deliveryModel* property refers to the commercial Cloud realization (i.e., public, hybrid or private). The property *budget* denotes the provider's invested capital. A Cloud's purpose is to serve an arbitrary amount of *Workloads.* The *Workload* properties *type*, *periodicity* and *userProfile* specify the intensity and characteristics of the imposed workloads.

The Cloud can have many possible *Configurations*, denoting the system's hardware and software components as well as the possible mappings between them. The Cloud *Components* can be encountered in different states, as enumerated in *ComponentState*, indicating their health and availability. We distinguish five different states: "Switched-on" indicating that the component is in use; "Switched-off" indicating that the component has been deactivated; "Idle" indicating that no function or service is running on this component; "Overloaded" indicating that there are not enough resources in this component to complete a requested task; and "Non-responding" indicating a possible failure.

4.1 Cloud Components

The central entity *Component* abstracts the Cloud infrastructure. As Figure 3 shows, we distinguish between two component types, the *HardwareComponent* denoting the Hardware Infrastructure and the *SoftwareComponent* denoting the Software Infrastructure.

The software landscape can be summarized as *DeployedServices* executing on Virtual Machines (*VM*)

[2]http://www.eclipse.org/modeling/

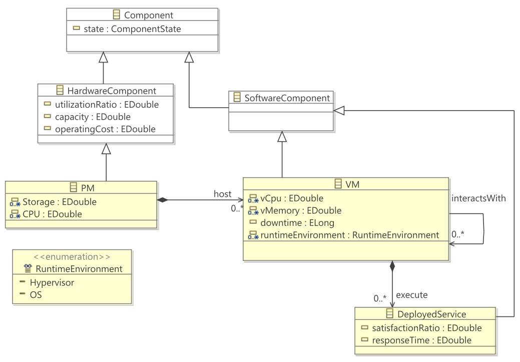

Figure 3: Cloud Infrastructure.

while assisted by *RuntimeEnvironment* Entities, e.g., the hypervisor. Other properties that the *VM* class entail, are *vCPU* i.e., the quantity of physical CPU assigned to it, *vMemory* i.e., the quantity of physical Memory assigned to it, and the *downtime* i.e., time period when a VM is unavailable as a VM migration is being carried out. The relation *interactsWith* models shared communication among VMs.

The *HardwareComponent's* properties *utilizationRatio capacity* and *operatingCost* provide feedback relevant to the efficiency and cost of a configuration. Core entity of the hardware landscape are the Physical Machines (*PM*), comprised of *storage* and *CPU.* PMs can host multiple VMs during their operation.

4.2 Cloud Variability

In Figure 4 the volatility of the Cloud's context and its capability to adapt accordingly are depicted. Imagine for example, the case of two VMs sharing resources and assume that the workload of one increases leading to an SLA violation. The Cloud configuration will have to be adapted by e.g., increasing the size of the overloaded VM so as to support its hosted services without SLA violations.

A Cloud can have many possible *High Level Goals* e.g., capital/operating costs optimisation, enforcement of resilience etc. The *Detector* interface is responsible to diagnose the most urgent among them and dynamically form a corresponding multi-objective optimisation problem, modelled by the *Revenue* entity. The Revenue essentially denotes a - closely aligned with the runtime context - objective function, expressing the quantity that has to be currently maximised (e.g., server utilisation ratio) or minimised (e.g., network Traffic) to achieve an efficient system configuration.

The realisation of the Revenue yields an *AdaptationStrategy*, which provides means towards a Cloud reconfiguration. The *AdaptationStrategy* is imple-

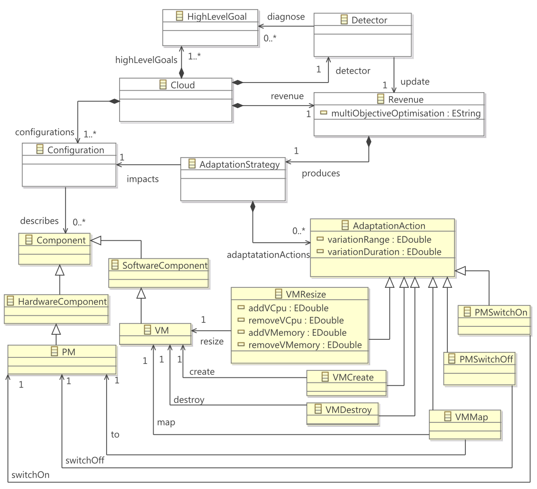
Figure 4: Cloud Variability.

mented via a concatenation of *AdaptationActions*. The *AdaptationActions* entity captures the degrees of freedom of a Cloud resources landscape describing the types of actions that can be performed towards transitioning from an old configuration to an improved target. The identified possible action types span: (a) *VMResize* i.e., how much virtual memory or virtual CPU should a VM receive (b) *VMCreate* i.e., creation of a new VM (c) *VMMap*: (re)selection of PMs to host the aforementioned VMs (d) *VMDestroy* i.e., switch off a VM (e) *PMSwitchOn* i.e., activation of a new PM (f) *PMSwitchOff* i.e., switching-off a PM.

The property *variationRange* specifies the range in which the aforementioned entities may vary, while the property *variationDuretion* indicates time limits for the reconfiguration actions.

5 USE CASE

In this section we provide a use case to indicate the applicability of the proposed meta-model.

Let's consider the simple Cloud infrastructure of Figure 5, consisting of three PMs (PM0, PM1, PM2) hosting over Xen hypervisor two intercommunicating VMs (VM0, VM1), each one executing a different enterprise service (DeployedService0, DeployedService1), serving the customer's workload. The Provider's High level goal has been diagnosed as minimisation of OPEX expenses, hence the problem's revenue can be formalised as e.g., min $\sum_{i=1}^{n} c_i +$ min $\sum_{i=1}^{m} t_j$ (where c_i symbolizes residual ratio of resources and t_j the performance overhead due to reconfiguration). This revenue basically imposes the need to minimize the idle infrastructure. We assume that in

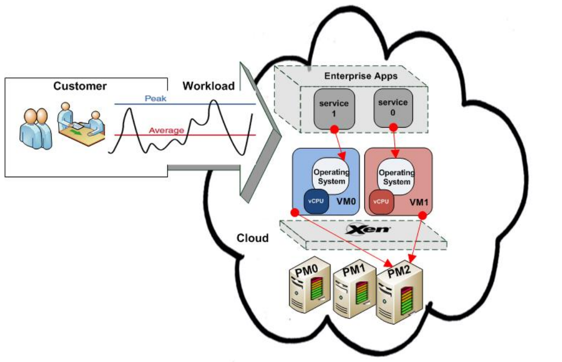
Figure 5: Cloud Use Case.

the initial configuration PM2 hosts VM0 and VM1, while PM0 and PM1 are idle. Moreover, DeployedService0 runs on VM1 and DeployedService1 is executed on VM0.

Let's assume that the PM utilisation thresholds are violated for an extended period of time, resulting to non satisfactory OPEX. The Cloud has to reconfigure in order to provide a resources configuration solution that will support improved asset usage, therefore optimised OPEX.

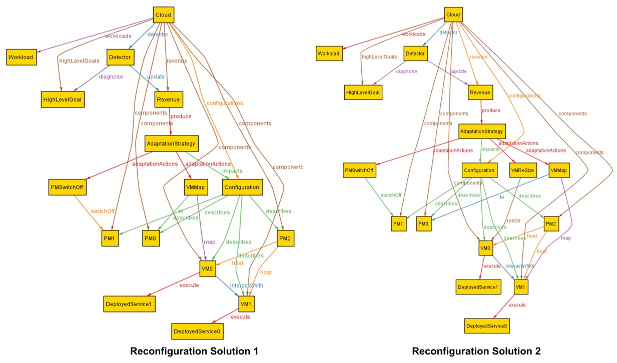
Figure 6: Possible Reconfiguration Solutions.

The description of our meta-model structures, enables automated extraction and exploration of the valid reconfiguration solutions space of the problem. Figure 6 shows two valid reconfiguration instances. In the first, the concatenation of reconfiguration actions yielded to optimise the revenue include: switch-off PM1 which does not host any VMs, and migration of VM0 from PM2 to PM0. From further observation of the instance we notice that VM0 and VM1 are intercommunicating, hence we gather that DeployedService1 and DeployedService0 have dependencies (e.g. may share memory pages). This feedback can assist future decisioning of these VMs sizing and location. The second reconfiguration instance, models another possible solution of the same optimisation problem where PM1 is switched-off, VM0 receives less memory resources and and VM1 is migrated from PM2 to PM0. Figure 7 depicts the application of the first reconfiguration solution to the use case cloud.

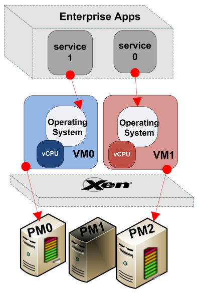

Figure 7: Cloud Infrastructure After Reconfiguration.

The proposed modelling concepts provide means for automated exploration and reasoning of the Cloud resources landscape space. Hence, both reconfigurations pictured in Figure 6 show possible solutions of the problem discussed in the use case. However, we can not answer which solution dominates the other, or which solution will provide a faster alleviation of the identified issues. Therefore, our next step towards realizing our vision is the design of a heuristic to control the evolution of reconfigurations solution space as Cloud objectives change, enforcing runtime constraints (e.g. time range of adaptation steps).

6 CONCLUSIONS

Cloud Computing promises infrastructure elasticity at a low cost. However, modern datacenters are becoming increasingly complex. In this paper we are concerned with the issue of resource management in Clouds. Particularly, we suggest that a combination of MDE and dynamic multi-objective optimization can efficiently manage the resulting trade-offs between the various possible Cloud optimisation goals.

As a first step towards realizing our vision we presented a meta-model to describe the Cloud Computing resources landscape with focus on the provider's perspective. The presented modelling approach comprises information regarding the provider's optimisation objectives, Cloud physical and virtual infrastructure as well as points of dynamic infrastructure variability. The meta-model aims to serve as a basis to facilitate automated extraction of Cloud resources feasible space, towards selecting optimal configurations.

As part of our on-going work, we intend to design a heuristic to control the evolution of reconfiguration space during replacement of problem objectives. We also aim to formalise the possible reconfigurations within the model as run-time model transformations based on Cloud revenue and available adaptation actions. Further plans, include the use of Kevoree [3] experimental Cloud platform to validate our approach.

REFERENCES

Abdelzaher, T., Shin, K. G., and Bhatti, N. (2001). Performance Guarantees for Web Server End-Systems: A Control-Theoretical Approach. *IEEE Transactions on Parallel and Distributed Systems*.

Becker, S., Koziolek, H., and Reussner, R. (2009). The Palladio component model for model-driven performance prediction. *J. Syst. Softw.*

Chuen, C., Mark, T., Niyato, D., and Chen-khong, T. (2011). Evolutionary Optimal Virtual Machine Placement and Demand Forecaster for Cloud Computing. *International Conference on Advanced Information Networking and Applications*.

David Breitgand, Alessandro Maraschini, J. T. (2011). Policy-Driven Service Placement Optimization in Federated Clouds. Technical report, IBM Research.

Ferreto, T. C., Netto, M. A. S., Calheiros, R. N., and De Rose, C. A. F. (2011). Server consolidation with migration control for virtualized data centers. *Future Generation Computer Systems*.

Huber, N., Brosig, F., and Kounev, S. (2012). Modeling dynamic virtualized resource landscapes. In *ACM SIGSOFT*.

Josyula, Orr, P. (2012). *Cloud Computing: Automating the Virtualized Data Center*.

Khanna, G., Beaty, K., Kar, G., and Kochut, A. (2006). Application Performance Management in Virtualized Server Environments. In *NOMS 2006*, pages 373–381.

Kusic, D., Kephart, J. O., Hanson, J. E., Kandasamy, N., and Jiang, G. (2008). Power and performance management of virtualized computing environments via lookahead control. *Cluster Computing*.

Meng, X., Isci, C., Kephart, J., Zhang, L., Bouillet, E., and Pendarakis, D. (2010a). Efficient resource provisioning in compute clouds via VM multiplexing. In *7th international conference on Autonomic computing*.

Meng, X., Pappas, V., and Zhang, L. (2010b). Improving the scalability of data center networks with traffic-aware virtual machine placement. In *29th conference on Information communications*.

Piao, J. T. and Yan, J. (2010). A Network-aware Virtual Machine Placement and Migration Approach in Cloud Computing. In *9th International Conference on GCC*, pages 87 –92.

Shrivastava, V., Zerfos, P., Lee, K.-w., Jamjoom, H., Liu, Y.-h., and Banerjee, S. (2011). Application-aware virtual machine migration in data centers. *IEEE INFOCOM*.

Stage, A. and Setzer, T. (2009). Network-aware migration control and scheduling of differentiated virtual machine workloads. In *ICSE Workshop*.

Tesauro, G., Jong, N., Das, R., and Bennani, M. (2006). A Hybrid Reinforcement Learning Approach to Autonomic Resource Allocation. In *ICAC '06*.

[3] http://www.kevoree.org/