A Multi-agent System to Monitor SLA for Cloud Computing

Benjamin Gâteau
Public Research Centre Henri Tudor, SSI Dept., 29 Av. John F. Kennedy, L1855 Luxembourg City, Luxembourg

Keywords: Dynamic Infrastructure, Cloud Computing, SLA Management, Multi-agent Systems.

Abstract: More than a technological solution Cloud Computing is also an economical advantage and already play an important roles in the information technology's area. Thereby and in order to ensure a QoS commitment between a provider and a customer, Service Level Agreements (SLA) describe a set of non-functional requirements of the service the customer is buying. In this paper, we describe how we can use Multi-Agent Systems (MAS) to manage SLA and we present the monitoring tool we develop with the SPADE framework.

1 INTRODUCTION

Evolution of high level ICT infrastructures in Europe brings some difficulties due by a complex management and the need of more and more energy. One current fashionable solution to this problem is the use of cloud computing services. Cloud computing is a ICT model allowing an easy access through networks to mutualised and configurable resources able to be quickly activated and deactivated.

Cloud computing has become a mainstream technology offering mutualisation of IT infrastructures as services along several paths such as Software (SaaS), Platform (PaaS), and Infrastructure (IaaS). Companies such as Amazon, Microsoft, IBM, and Google, to name but a few, offer such services, which rely on virtualization and pay-as-you-go business models. Flexibility and elasticity are also important features of cloud computing made possible by the concept of "dynamic Infrastructure."

Dynamic infrastructure is an information technology paradigm concerning the design of datacenters so that the underlying hardware and software can respond dynamically to changing levels of demand in more fundamental and efficient ways than before. The basic premise of dynamic infrastructures is to leverage server virtualization technology to pool computing resources wherever possible, and then to allocate these resources on-demand. This allows for load balancing and is a more efficient approach than keeping massive computing resources in reserve to run tasks that take place, for example, once a month. The potential feature benefits include enhancing performance, scalability system availability and uptime, and the ability to perform routine maintenance on either physical or virtual systems all while minimizing interruption to business operations and reducing cost for IT. Dynamic infrastructures also provide the fundamental business continuity and high availability requirements to facilitate cloud or grid computing.

Dynamic infrastructures allow the use of resource when it is needed. For instance, Figure 1 represents the case when clients of the service provider don't use their virtual machine at a certain time (night for instance). It could happen that only few virtual machines are running on each physical machine. So physical servers are not fully used and gathering virtual machines on one physical server could permit to save energy and use less resource. But by moving virtual machines, the provider must ensure that the customers pay for the service he previously specified and that the Quality of Service (QoS) is still the same.

QoS delivery affects the value of the service for the client and significantly depends on IaaS or PaaS provider's infrastructure. Therefore there is need to divide responsibility/risk between *X*aaS provider and customer. To describe the responsibilities a formal description of non-functional requirements from the client's point of the view is required.

In (Gâteau, 2011) we aimed at defining a SLA infrastructure that provides the same guarantees and proofs that we can find in e-contract management, with more flexibility. In this paper we are considering the monitoring part of the SLA management. In the CLOVIS (Cloud computing improvement through risk and SLA management) project (Morin et al., 2012; Stamou et al., 2012) we intend to link SLA management and Information System Security Risk Ma-

Gâteau B..
A Multi-agent System to Monitor SLA for Cloud Computing.
DOI: 10.5220/0004970106470652
In *Proceedings of the 16th International Conference on Enterprise Information Systems* (ICEIS-2014), pages 647-652
ISBN: 978-989-758-028-4

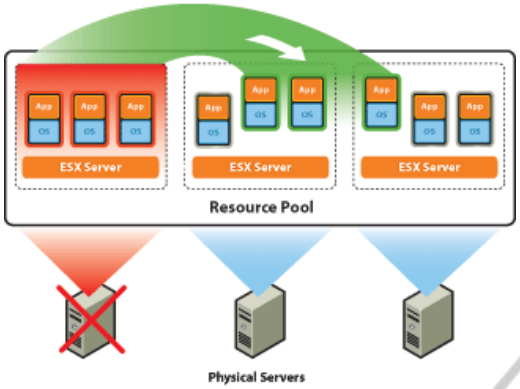

Figure 1: Moving VM in a VMware Infrastructure.

nagement (ISSRM). For that, we planned to improve cloud computing governance, risk management and compliance by producing a complete framework intended for people in charge of the security of service-oriented architectures such as cloud computing platforms. Beyond the provision of such outputs to the ISSRM community, their development is intended to wider raise the incentive of the security management issue in the emerging cloud computing concept.

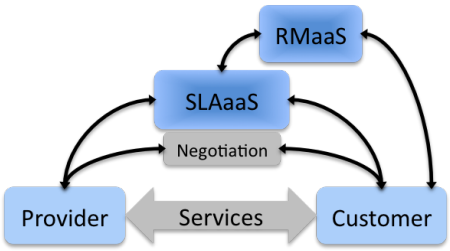

Figure 2: CLOVIS framework.

The high-level model of the CLOVIS framework assumes the existence of a risk management (RM) tool, which can provide a customer with an adequate risk assessment according to a customer's service criteria and requirements (RMaaS). Such service requirements can then be utilized as input to correctly match with available service offers, i.e. pre-instantiated SLAs that are submitted by service providers. SLOs and risk controls can be manipulated for this purpose. In paper (Katerina Stamou and Morin, 2013), the SLAaaS components was presented and a monitoring component as an active agreements management was introduced.

As we can see in Figure 2 a user (i.e. customer) has two choices: (i) use the RMaaS module to get assisted with the selection of requirements that he/she wants to have in cloud offerings or (ii) directly submit functional criteria to the SLAaaS module and request a matching with available service offers. This approach provides customers with a more granular manipulation of existing SLA offers compared to the ones currently consumed in the market. When a matching set is found, a negotiation process can take place between a customer and the selected provider to agree and sign a SLA. The provision of the service(s) can then be initiated.

Once the customer chose and signed a SLA with a provider through the RMaaS and the SLAaaS, he/she potentially wishes to be sure that the terms they agreed on are fulfilled. If a term (security or guarantee terms implemented by provider in order to reduce risks) is violated during service provisioning, the risk underneath is possibly more likely to happen. Thats why we propose a monitoring tool in order to provide feedback on the status of each term of the SLA and their potential impact on risks. Monitoring will enable both customers and providers to ensure SLAs are properly enforced but also, will be a mean to "close the loop": from deducing cloud computing service requirements and their instantiation based on security risks; to auditing risks based on the monitoring of cloud computing service delivery. At this end, the use of the fully distributed architecture supported by a multi-agent system defined in (Gâteau, 2011) will be used.

This paper is organized as follows. We make a quick survey on SLA management in the next section, then we present our normative multi-agent system solution before proposing a solution to monitor SLA with an electronic institution based on multi-agent system for the cloud computing.

2 SLA MONITORING

SLA describes a set of non-functional requirements of the service the customer is buying. The agreement usually contains also penalties when the requirements are not met. An example of a non-functional requirement would be "RTO - Return to Operation time for a service in case of a failure. To describe a non-functional requirement we needed an objective to be achieved (e.g. RTO under two minutes) and a set of indicators that prove the objective is met (e.g. new instance bootstrap time). The objective to be achieved is called "Service Level Objective (SLO) and the indicators are called "Key Performance Indicators (KPIs). SLO is the objective of service quality that has to be achieved. It is represented by a set of measurable KPIs with thresholds to decide if the objective is fulfilled or not. The fulfillment of an SLOs describes a state of service when all of the SLOs key performance indicators are within a specified thresholds.

KPIs usually consist from one or more raw monitored values including min, avg and max specifying the scale. They can also represent some aggregated measurement (e.g. average output) within a sliding window that is combined from one or more monitoring outputs. The provider has to be able to measure and affect the KPIs otherwise it would not make sense to guarantee them. The cloud computing infrastructures are usually large scale, therefore SLAs need to be formally described to enable their automated handling and protection.

Automated SLA protection is based on a set of policy rules. Each policy rule is formed by one or more conditions (KPI's value matching pattern) and one or more actions. KPIs are periodically evaluated according to defined policies. If one or more conditions are met, then appropriate actions are triggered. An example of the policy action can be increasing number of service instance. The action is triggered by a server load KPI matching the policy rule. This enables to automatically keep the load KPI under certain value specified by the SLO and avoid violating the general SLA. These kinds of rules are named *elasticity rules*.

Specification of the policies (rules and action) is a complex task in large scale dynamic system. Analysis of historical KPI data and triggered actions can be used to specify new or modify existing policies. This enables system adaptation and automated SLA protection evolution. System can for example learn from historical data about periodical service load peaks and generate specific rules to keep the system with throughput specified in SLA . Another research challenge is the specification of models able to describe and simulate these kinds of dynamic systems.

In his PhD thesis relating to E-contract modeling and e-enactement (Krishna, 2010), P. Radha Krishna describe the clauses of an electronic contract as obligations being part of a SLA. He also says: "*E-contract management solutions should maintain, monitor and manage contract rules derived from these SLAs. Contract parties should verify QoS parameters by performing an SLA monitoring, which involves monitoring the performance status of the offered service. The e-contract management system could assess the SLA requirements and apply penalties if there is any deviation.*"

The SLA4D-Grid project (Wieder et al., 2009) defines a SLA management layer on top of an existing infrastructure providing e-contracting capabilities. The infrastructure specifies, implements and deploys a SLA-based service stack for e-Contracting. The authors say: "*The SLA4D-Grid project is designing and implementing an SLA management layer. The functions of the developments cover the complete SLA life-cycle, including SLA design, contract establishment, SLA provisioning, and SLA monitoring.*".

The SLA@SOI project (Comuzzi et al., 2010) is a FP7 project dealing with the definition of a SLA management framework. The consortium defined a reference architecture, specified a SLA template and the methodology to translate SLAs into monitoring specifications (for the EVEREST environment). With the RESERVOIR project, they the defined how using cloud standards for the interoperability of cloud frameworks.

In this global trend, multi-agent systems (MAS) have been introduced to achieve the automation in creation, execution and monitoring of e-contracts by agents on behalf of users. The resulting contracts consist in digital agreement between contractual parts where rights and duties in terms of deliverable, costs and delays of the participants are explicitly represented. However such contracts often lead to inflexible relations between participants. The obtained result is in contrary to the requirements of open and dynamic system that are stressed by the actual business paradigms aiming at improving the competitiveness of companies like dynamic virtual enterprises and dynamic service outsourcing (Hoffner et al., 2001). Moreover, few of the existing research works take into account the monitoring of contracts clauses (Padovan et al., 2002) that bind agents together.

In (Boissier and Gâteau, 2007) we proposed an Electronic Institution model based on MAS to manage electronic contract by specifying obligations. In (Gâteau, 2011) we aimed at doing the same for the management of SLA for Cloud Computing. In this paper we will describe how we can used this model to monitor SLA signed between a consumer and a provider of cloud computing services.

3 A MULTI-AGENT BASED MONITORING FRAMEWORK

3.1 The Model Principle

In (Gâteau, 2011) we proposed a multi-agent support for the enactment and monitoring of the different SLO specified in the SLA. SLAs specify the agreement between customers and the provider concerning their participation to the distributed execution of the job. A SLA must describe both the functioning and the structure organizing this functioning. Moreover it contains explicit legal dimensions bearing on the involved participants. In order to take this into ac-

count we propose to use a "normative organizational model", called $\mathcal{M}$OISE^{Inst}, to express SLA. This normative organizational model is accompanied by a specialized "normative middleware", called $\mathcal{S}$YNAI[1] to monitor and enforce legal aspects expressed in the SLAs.

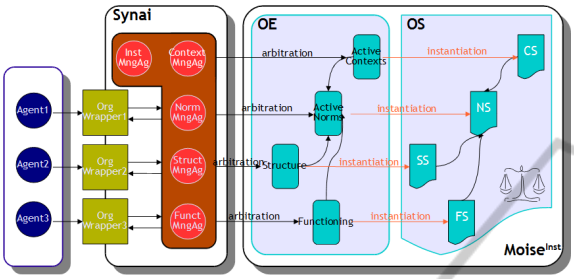

Figure 3: $\mathcal{M}$OISE^{Inst}, normative organization specification model, and $\mathcal{S}$YNAI, normative middleware.

In this paper we propose a first implementation based on the SPADE framework (Smart Python Agent Development Environment) in the context of the CLOVIS framework. This implementation doesn't literally respect the $\mathcal{M}$OISE^{Inst} model. As reminder, $\mathcal{M}$OISE^{Inst} (Gâteau et al., 2007) is founded on the $\mathcal{M}$OISE^{+} organizational model[2](Hubner et al., 2002) and is composed of the following components that are used to specify an organisation of agents in terms of structure, functioning, evolution and norms (OS of the Figure 3):

- A *Structural Specification* (SS) defines: (i) the *roles* that agents will play in the organization, (ii) the *relations* between these roles in terms of authority, communication or acquaintance, (iii) the *groups*, additional structural primitives used to define and organize sets of roles;
- A *Functional Specification* (FS) defines global *business processes* that can be executed by the different agents participating to the organization according to their roles and groups;
- A *Contextual Specification* (CS) specifies, a priori, the possible evolution of the organization in terms of a *state/transition graph*;
- A *Normative Specification* (NS) defines the deontic relations gluing the three independent specification (SS, FS, CS). This NS clearly states rights and duties of each roles/groups of SS on sets of goals (missions) of FS, within specific states of CS.

[1] $\mathcal{S}$YNAI: SYstem of Normative Agents for Institution.

[2] $\mathcal{M}$OISE^{+}: Model of Organization for multI-agent System.

A BNF[3] complete definition of OS is available in (Gâteau, 2007).

3.2 The System

On the Figure 4, the architecture of our implemented solution is presented. The SPADE framework owned several agents dispatched between the monitoring system and the client's VM machine in the provider's infrastructure. 'S' is the server agent. Its role is to receipt call from probe agents when they are launched on a customer's VM.

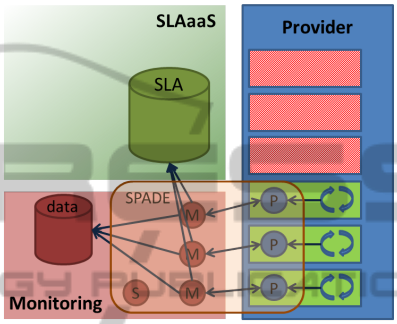

Figure 4: Architecture of the Monitoring System linked to the CLOVIS framework.

'M' is a MUX Agent and its role is to make the link between the monitoring system, the main component of the CLOVIS framework, namely the SLAaaS one and the VM to monitor. He is launched when the server agent receives a call from a probe agent. The newly created MUX agent is associated to the probe agent and will receive monitoring information from the probe agent. It will save all data in a database (we choose mongoDB).

'P' is the probe agent located on the VM of the provider and sending the measures values to its bound MUX agent. The probe agent set up the function it will call periodically and send back the result to the MUX agent.

In the SPADE system (as in several multi-agent platform), agents exhibits services and execute some behaviour. By making the link with $\mathcal{M}$OISE^{Inst}, Role and Group of the SS are represented by the services of the agents. The Goals that the agent must achieved are the behaviour they execute. And nothing represents the CS. The NS defines the link between the roles and the goal. Here, we could consider that the fact that an agent with such services must execute such behaviour is hard-coded. We are really far from the dynamic specification! Indeed, the OS is not represented or managed by the Server Agent or another Manager Agent.

[3] BNF: Backus-Naur Normal Form.

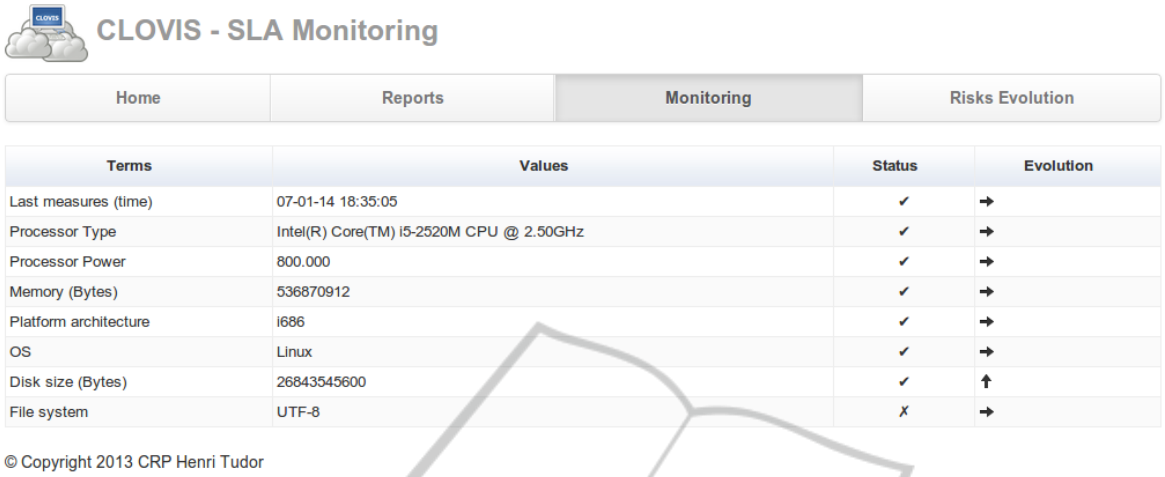

Figure 5: Web interface of the Monitoring System.

3.3 The Process

When a consumer of cloud computing services (IaaS in our context) signed a contract (and a SLA) with a cloud computing provider, the provider give him access to the VM he chose. In order to activate the monitoring, he as to install, configure (client ID and password and the address of the monitoring system where the server agent and the MUX agent are located) and execute the probe agent. Once the agent is running (and joined the multi-agent system hosted on the monitoring system), it sends a message to the server agent in order to announce its presence. The server send to it as feedback the address of its new MUX agent. Then the probe agent ask the MUX agent the details (terms) of the contract (SLA) signed between the provider and the customer. Once it has the answer it make the measures and sent them to the MUX agent which make them viewable by the user through a web interface as depicted in Figure 5.

We noted that protocols were very important in this implementation. SPADE propose behaviour defined through specific template. When a agent used this template to send a message to another agent, this is the behaviour running with this template which intercepts the message and processes it. We can consider that as a mean to define a kind of dialogical specification.

4 CONCLUSION

The context of this paper is the CLOVIS project in which the approach enables customers to take into account their needs in terms of security, when selecting a cloud service. We proposed a multi-agent system developed with SPADE and respecting the $\mathcal{M}$OISE^{Inst} model to monitor SLAs. We thus go a step further by not only ensuring that the service meets its initial specifications but also enabling customers to adjust their needs. This implementation of the $\mathcal{M}$OISE^{Inst} model is an alternative to Utopia (Schmitt et al., 2011) based on the JAVA language and the JADE framework (which is less flexible than SPADE developed in Python). This prepare future works dealing with the use of MAS respecting the $\mathcal{M}$OISE^{Inst} model in order to gather distributed data coming from specific agents and making others agents executing some action in order to respect the global goal of the organisation. We plan to use this future platform to complete existing building automation and making them smart.

ACKNOWLEDGEMENTS

This work is supported by the CLOVIS project jointly funded by the Swiss SNF and Luxembourg FNR Lead Agency agreement; under Swiss National Science Foundation grant number 200021E-136316/1 and Luxembourg National Research Fund (FNR) grant number INTER/SNSF/10/02.

REFERENCES

Boissier, O. and Gâteau, B. (2007). Normative multi-agent organizations: Modeling, support and control, draft version. In G. Boella, L. v. d. T. and Verhagen, H., editors, *Normative Multi-Agent Systems. Dagstuhl Seminar Proceedings 07122, Interna-*

tionales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany.

Comuzzi, M., Kotsokalis, C., Rathfelder, C., Theilmann, W., Winkler, U., and Zacco, G. (2010). A framework for multi-level sla management. In Dan, A., Gittler, F., and Toumani, F., editors, *Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops*, volume 6275 of *Lecture Notes in Computer Science*, pages 187–196. Springer Berlin / Heidelberg.

Gâteau, B. (2007). *Modlisation et Supervision d'Institutions Multi-Agents*. PhD thesis, Ecole Nationale Superieure des Mines de Saint Etienne. defended at CRP Henri Tudor, Luxembourg.

Gâteau, B. (2011). A mas to manage and monitor sla for cloud computing - a draft position paper. In *CLOSER*, pages 687–692.

Gâteau, B., Boissier, O., Khadraoui, D., and Dubois, E. (2007). Controlling an interactive game with a multi-agent based normative organizational model. In Vázquez-Salceda, J., Boella, G., Boissier, O., and Matson, E., editors, *"Coordination, Organizations, Institutions, and Norms in Agent Systems II"*, volume 4386 of *LNCS*, pages 86–100. Springer Berlin / Heidelberg.

Hoffner, Y., Field, S., Grefen, P., and Ludwig, H. (2001). Contract-driven creation and operation of virtual enterprises. *Comput. Networks*, 37(2):111–136.

Hubner, J. F., Sichman, J. S., and Boissier, O. (2002). $\mathcal{M}$OISE$^+$: towards a structural, functional, and deontic model for mas organization. In *Proceedings of the first International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 501–502, Bologna, Italy. ACM Press. ISBN 1-58113-480-0.

Katerina Stamou, Jocelyn Aubert, B. G. and Morin, J.-H. (2013). Preliminary requirements on trusted third parties for service transactions in cloud eco-systems. In *HICSS*, pages 4976–4983.

Krishna, P. R. (2010). *E-contract modeling and e-enactement*. PhD thesis, International Institute of Information Technology, Hyderabad - 500 032, INDIA.

Morin, J.-H., Aubert, J., and Gâteau, B. (2012). Towards cloud computing sla risk management: Issues and challenges. In *HICSS*, pages 5509–5514.

Padovan, B., Sackmann, S., Eymann, T., and Pippow, I. (2002). A prototype for an agent-based secure electronic marketplace including reputation tracking mechanisms. *International Journal of Electronic Commerce*, 6(4):93–113.

Schmitt, P., Bonhomme, C., Aubert, J., and Gâteau, B. (2011). Programming electronic institutions with utopia. In Aalst, W., Mylopoulos, J., Sadeh, N. M., Shaw, M. J., Szyperski, C., Soffer, P., and Proper, E., editors, *Information Systems Evolution*, volume 72 of *LNBIP*, pages 122–135. Springer Berlin / Heidelberg.

Stamou, K., Morin, J.-H., Gâteau, B., and Aubert, J. (2012). Service level agreements as a service - towards security risks aware sla management. In *CLOSER*, pages 663–669.

Wieder, P., Hasselmeyer, P., and Koller, B. (2009). Enhancing a national academic computing infrastructure with e-contracting capabilities. In Cunningham, P. and Cunningham, M., editors, *Proceedings of eChallenges e-2009 Conference*. ISBN: 978-1-905824-13-7.