A Holistic Service Provisioning Solution for Federated Cloud Infrastructures

Attila Kertesz, Gabor Kecskemeti, Zsolt Nemeth
MTA SZTAKI
H-1518 Budapest, P.O. Box 63, Hungary
{*keratt,kecskemeti,zsnemeth*}*@sztaki.hu*

Marc Oriol, Xavier Franch
Universitat Politecnica de Catalunya
08034 Barcelona, c/Jordi Girona 1-3, Spain
moriol@lsi.upc.edu, franch@essi.upc.edu

***Abstract*—Cloud Computing builds on the latest achievements of diverse research areas, such as Grid Computing, Service-oriented computing, business process modeling and virtualization. As this new computing paradigm was mostly lead by companies, several proprietary systems arisen. Recently, alongside these commercial systems, several smaller-scale privately owned systems are maintained and developed. In this paper we present our research results performed within the S-Cube European FP7 NoE project to enable automated service provisioning for users on a highly dynamic infrastructure consisting of multiple Cloud providers. We developed a Federated Cloud Management architecture that provides unified access to a federated Cloud that aggregates multiple heterogeneous IaaS Cloud providers in a transparent manner. We have also incorporated an integrated monitoring approach that enables more reliable provider selection in these heterogeneous environments.**

***Keywords*-Cloud Computing, Cloud Brokering, On-demand deployment, Service Monitoring.**

I. Introduction

Cloud Computing [1] offers simple and cost effective outsourcing in dynamic service environments and allows the construction of service-based applications extensible with the latest achievements of diverse research areas, such as Grid Computing, Service-oriented computing, business processes and virtualization. Cloud-based highly dynamic service environments [3] require a novel infrastructure that incorporates a high-level monitoring approach to support autonomous, on demand deployment and decommission of service instances. Virtual appliances (VA) encapsulate a complete software system prepared for execution in virtual machines (VM). Infrastructure as a Service (IaaS) cloud systems provide access to remote computing infrastructures by allowing their users to instantiate virtual appliances on their virtualized resources as virtual machines. Nowadays, several public and private IaaS systems co-exist provided by public service providers (like Amazon [6]) or by smaller scale privately managed infrastructures. Cloud solutions are also spreading fast in the academia with the emerging open-source tools, such OpenNebula [7], but these solutions can hardy interoperate.

Regarding related approaches, Buyya et al. [2] suggests a Cloud federation oriented, opportunistic and scalable application services provisioning environment called InterCloud. They envision utility oriented federated IaaS systems that are able to predict application service behavior for intelligent down and up-scaling infrastructures. Then, they list the research issues of flexible service to resource mapping, user and resource centric QoS optimization, integration with in-house systems of enterprises, scalable monitoring of system components. Though they address self-management and SLA handling, the unified utilization of other distributed systems are not studied, and they do not consider incorporating infrastructures of diverse Cloud providers.

In 2009, Amazon Web Services launched Amazon CloudWatch [6], that is a supplementary service for Amazon EC2 instances that provides monitoring services for running virtual machine instances. It allows to gather information about the different characteristics of resources, and based on that users and services are able to dynamically start or release instances to match demand as utilization goes over or below predefined thresholds. The main shortcoming is that this solution is tied to a specific IaaS cloud system and introduces a monetary overhead, since the service charges a fixed hourly rate for each monitored instance.

II. Federated Cloud Management Architecture

In order to provide unified access to a federated Cloud that aggregates multiple heterogeneous IaaS Cloud providers in a transparent manner, we designed the Federated Cloud Management (FCM) architecture that represents an interoperable solution for establishing a federated cloud environment. In this solution, users are able to execute services deployed on cloud infrastructures transparently, in an automated way. Virtual appliances for all services should be stored in a generic repository called FCM Repository, from that they are automatically replicated to the native repositories of the different Infrastructure as a Service cloud providers.

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement 215483 (S-Cube).

 DOI 10.1109/S-Cube.2012.6225504

When a user sends a service call to the system, he/she submits a request to the "*Generic Meta-Broker Service*" (GMBS) [4] specifying the requested service with a service description, the operation to be called, and its possible input parameters. The GMBS checks if the service has an uploaded VA in the generic repository, then it selects a suitable Cloud Broker for further submission. The matchmaking is based on static data gathered from the "*FCM Repository*" (e.g., service operations, description), and on dynamic information of special deployment metrics gathered by the Cloud Brokers. Currently we use the average VA deployment time and the average service execution time for each VA. VA deployment time assumes that the native repository already has the requested VA, thus includes only the service provision time on a specific IaaS cloud. The role of GMBS is to manage autonomously the interconnected cloud infrastructures with the help of the Cloud Brokers by forming a federation.

Each "*Cloud Broker*" has an own queue for storing the incoming service calls, and manages one virtual machine queue for each VA. Virtual machine queues represent the resources that currently can serve a virtual appliance specific service call. The main goal of a Cloud Broker is to manage the virtual machine queues according to their respective service demand. The default virtual machine scheduling is based on the currently available requests in the queue, their historical execution times, and the number of running VMs. The secondary task of the Cloud Broker involves the dynamic creation and destruction of the various VM queues. Virtual Machine Handler components are assigned to each virtual machine queue. These components process the virtual machine creation and destruction requests placed in the queue. The requests are translated and forwarded to the corresponding IaaS system. This component is a cloud infrastructure-specific one, that uses the public interface of the managed infrastructure. Independently from the virtual machine scheduling process the Cloud Broker also handles the queue of the incoming service calls. As a result, these calls are dispatched to the available VMs created in the previously discussed manner. In order to optimize service executions in highly dynamic service environments, our architecture organizes the virtual appliance distribution as a background process with the automatic service deployment component that can decompose virtual appliances to smaller parts. With the help of the minimal manageable virtual appliances, the Virtual Machine Handler is able to rebuild these decomposed parts in the IaaS system on demand, that results in faster VA deployment and in a reduced storage requirement in the native repositories.

In this architecture users are able to execute services deployed on cloud infrastructures transparently, in an automated way. The FCM Repository contains information on these services (including service descriptions and their virtual machine images or VAs). When a service is deployed on a new host, the service deployment component registers its new endpoint to the registry. Upon decommissioning, these endpoint registrations are removed from the registry. During operation, the *SALMon* [5] monitoring subsystem allows the components in FCM to order regular testing on the deployed services according to pre-defined metrics based on the service availability data from the registry. In our system, users send service calls as request submissions to the *GMBS* component. "*Federated call submissions*" specify the requested service with a service description, the operation to be called, and its possible input parameters. The GMBS checks if the service is registered to the registry, and if so, it selects a suitable *Cloud Broker* for further submission, otherwise rejects the request. Based on service usage patterns (e.g. average service response time, call frequency) the GMBS orders the monitoring of the deployed service from SALMon. The monitoring results allow sophisticated matchmaking algorithms based on static data gathered from the registry and on dynamic information of special metrics gathered by SALMon and the CBs. GMBS forms a cloud federation by enabling the autonomous management of the interconnected cloud infrastructures through CBs.

III. Conclusion

In this paper, we proposed a Federated Cloud Management solution that acts as an entry point to cloud federations. Its architecture incorporates the concepts of meta-brokering, cloud brokering and on-demand service deployment. The meta-brokering component provides transparent service execution for the users by allowing the system to interconnect the various cloud broker solutions managed by aggregating capabilities of these IaaS cloud providers. We have also introduced how a sophisticated service monitoring approach is used to consider provider reliability.

References

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. FGCS, vol. 25, no. 6, pp. 599–616, June 2009.

[2] R. Buyya, R. Ranjan, and R. N. Calheiros. InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. LNCS, vol. 6081, 2010.

[3] E. Di Nitto, C. Ghezzi, A. Metzger, M. Papazoglou, and K. Pohl. A journey to highly dynamic, self-adaptive service based applications. Automated Softw. Eng, v15, pp. 313–341, 2008.

[4] A. Kertesz and P. Kacsuk. GMBS: A new middleware service for making grids interoperable. FGCS, v26, pp. 542–553, 2010.

[5] M. Oriol, X. Franch, J. Marco, D. Ameller. Monitoring adaptable soa-systems using salmon. In Workshop on Service Monitoring, Adaptation and Beyond, pp. 19–28, 2008.

[6] Amazon Web Services. http://aws.amazon.com/ec2/, 2009.

[7] OpenNebula cloud. http://opennebula.org/, 2011.