Cloud Interoperability via Message Bus and Monitoring Integration

Vincent C. Emeakaroha, Philip D. Healy, Kaniz Fatema, and John P. Morrison

Irish Centre for Cloud Computing & Commerce
University College Cork, Ireland
{vc.emeakaroha,p.healy,k.fatema,j.morrison}@cs.ucc.ie

Abstract. In recent years, we have seen the rapid emergence of a plethora of Cloud providers with individual infrastructures, APIs and application description formats. This heterogeneity has resulted in vendor lock-in, which reduces consumer flexibility in terms of negotiation power, reaction to price increases and freedom to change provider. Achieving interoperability between Clouds is a means of addressing this issue. To realise this, the use of open standards has been suggested, but the existing standards are focused mainly on portability instead of interoperability. Besides, this heterogenous nature of Clouds makes inter-Cloud monitoring to facilitate interoperable Cloud management difficult. In this paper, we present a novel integrated approach to achieve interoperability between Clouds and to facilitate the management of service provisioning using multiple Clouds. The approach is based on the integration of a holistic message bus system with monitoring techniques. We present the design and implementation descriptions, and based on a use case scenario, we demonstrate a practical realisation of our approach.

Keywords: Cloud Computing, Interoperability, Interoperable Cloud Management, Message Bus, Data Interchange Format, Monitoring Framework.

1 Introduction

In recent years, many Cloud providers such as Amazon, Google, Microsoft have emerged, each with their own infrastructure, API and application description format [18]. Each promotes its own approach and incompatible implementation, which inhibits the adoption of a particular standard for Cloud service provisioning. Interoperability is a means of addressing these issues, allowing the numerous benefits of Cloud computing, *e.g.,* unlimited resource availability, to be realised. Interoperability embodies the ability to easily exchange information and data between a collection of providers in a usable form. Lack of interoperability is seen as a hinderance to Cloud adoption because enterprises fear vendor lock-in [9]. Once an enterprise chooses a provider, it becomes very costly to change to another. One consequence of vendor lock-in is reduced negotiation power in reaction to price increases.

D. an Mey et al. (Eds.): Euro-Par 2013 Workshops, LNCS 8374, pp. 65–74, 2014.

The use of open standards and co-operation between Clouds are suggested tactics for addressing interoperability [2,4,9]. However, some authors argue that using standards is just one piece of the solution because achieving interoperability also requires sound architecture principles and dynamic negotiation capabilities [10]. However, the reality is that existing standardisation efforts are focused mainly on portability, which is the ability to upload services and data from one provider to another, instead of on interoperability.

Service and resource provisioning in Clouds may be performed in compliance with a set of predefined non-functional properties specified and negotiated by means of a Service Level Agreement (SLA) [3]. To facilitate the management of service offerings and the guaranteeing of agreed SLA objectives, advanced SLA enforcement strategies, including appropriate monitoring techniques, are necessary. The heterogeneous nature of individual Cloud platforms makes monitoring to facilitate the management of interoperable or hybrid Clouds challenging. Existing Cloud monitoring tools are mainly designed for specific platforms and have limited support for interoperability. We argue that this issue can be addressed by implementing a robust message bus system to facilitate inter-Cloud management via monitoring.

In this paper, we present a novel integrated approach for achieving interoperability in Clouds and for facilitating the operational management of service deployments across interoperable Clouds. The approach is based on the integration of a holistic message bus system with monitoring techniques. We propose a monitoring framework that gathers data for supervising Cloud service provisioning and for making informed decisions. The message bus system provides a holistic communication mechanism that is capable of supporting messaging at different levels of abstraction within and between Clouds. It realises the basis for communicating information, including monitored data, within and between Clouds in an interoperable manner with the help of open data interchange formats.

The rest of the paper is organised as follows: Section 2 gives some background information and discusses the related work. In Section 3, we present the design of our monitoring and SLA enforcement framework. Section 4 discusses the message bus system, including its architecture and the data interchange format. In Section 5, we present the integrated approach and a use case scenario demonstration while Section 6 concludes the paper and highlights our future work in this direction.

2 Background and Related Work

There are several ongoing Cloud standardisation projects relating to interoperability. NIST[1], OMG[2] and DMTF[3] play an important role in these projects. As part of their efforts, they have defined use cases describing the manner of

[1] National Institute of Standards and Technology.
[2] Object Management Group.
[3] Distributed Management Task Force.

interactions between Cloud providers and consumers. From an interoperability perspective, the relevant use cases can be summarised as follows [9]:

- *User Authentication*: The ability of a user who has established an identity with a Cloud provider to use the same identity to access another provider.
- *Workload Migration*: The act of moving a workload or service that executes in a Cloud to another without major changes.
- *Data Migration*: The ease of exchanging data that resides in a Cloud with others. This can be, for example, monitored data gathered while executing outsourced services.
- *Workload Management*: The development of custom management tools that are capable of managing multiple Cloud resources from different providers.

The work in this paper is focused on addressing interoperability issues regarding workload and data migrations, and the development of custom management tools. The issue of authentication is out of scope and is not further considered.

In the following, we analyse the previous work on achieving interoperability in Clouds. Nguyen *et. al* [11] address interoperability from the perspective of developing interoperable Cloud services that can be simultaneously deployed on multiple IaaS Clouds. Their approach uses a high-level abstraction layer to provide a unified interface to developers for managing the entire service life cycle. However, they do not address means of realising interoperable resource outsourcing. Sotiriadis *et al.* [12] propose a decentralised meta-broker concept for inter-Cloud resource management. In their concept, a broker is responsible for exchanging and monitoring a consumer resource request. On top of these brokers, they place meta-brokers to enable inter-Cloud operation. Their approach does not consider holistic messaging.

Hsieh *et al.* [8] propose a design for a secure interoperable Cloud-based Personal Health Record (PHR) service. To achieve interoperability, they use the Continuity of Care Document (CCD) format for both storing and exchanging individual PHR information. Bahga *et al.* [1] discuss a Cloud-based approach for interoperable electronic health record system named Cloud Health Information System Technology Architecture (CHISTAR) that uses a generic design methodology to achieve semantic interoperability. Both of these approaches are centered on healthcare systems and do not consider compute resource sharing among Clouds.

Similar to our approach, Tovarnak *et al.* [14] discuss requirements for the producer of monitoring information to address issues of representation, processing and distribution. To this end, they propose the use of an extensible data format to structure monitored information to be usable by multiple consumer and thereby achieve interoperability. Their approach is event-based and not general purpose. Williams *et al.* [17] present *Xen-Blanket*, a thin deployable virtualisation layer that can homogenise diverse Cloud infrastructures. But their concept considers only the deployment of VM instances across Clouds.

To the best of our knowledge, none of the existing work considers the integration of monitoring with a holistic message bus system that uses a common data

interchange format to realise interoperability and thereby support interoperable Cloud management.

3 Cloud Monitoring Techniques

Monitoring techniques can facilitate the interoperable management of Cloud service provisioning. In this section, we describe a novel resource monitoring and SLA enforcement framework.

3.1 Resource Monitoring and SLA Enforcement Framework

Efficient resource and SLA management in Clouds require appropriate monitoring. Normally, SLA parameters such as *availability, throughput,* and *response time* represent the performance goals of applications. However, in Clouds, applications are provisioned with underlying infrastructure resources, which are characterised by low-level resource metrics such as, *CPU, memory*, and *storage*. Thus, there is a gap between the SLA parameters and the low-level resource metrics. To bridge this gap and guarantee the SLAs of applications, we propose a monitoring framework, which is an extension of the Low-level resource Metrics to High-level SLA (LoM2HiS) monitoring and mapping framework [5]. The framework monitors Cloud infrastructure resource metrics to determine their status while applications are being executed in a Cloud environment. This framework aims to address two issues: (i) monitoring of low-level resource metrics and (ii) mapping of the monitored metrics into SLA parameters so as to monitor the application SLA at runtime.

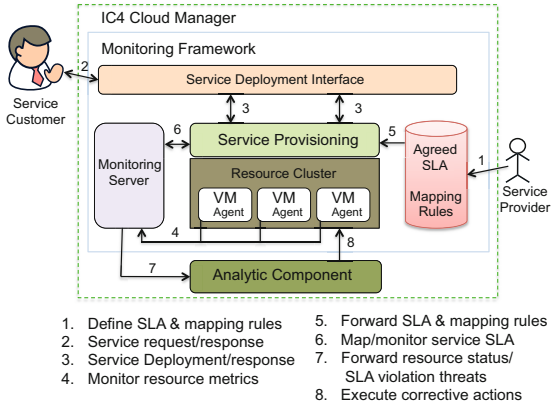

Fig. 1. Monitoring Framework Overview

Figure 1 depicts an overview of the monitoring framework. The provider defines the mapping rules for converting the low-level resource metrics monitored by the embedded agents. The Monitoring Server does the mapping and the evaluation of the application SLAs at runtime. When a violation threat is detected,

the analytic component is informed and triggers corrective actions. The LoM2HiS framework has been successfully used for resource monitoring and detecting SLA violations in private Cloud environments and in related areas [6, 7, 13].

3.2 Extensions to the LoM2HiS Framework

The original LoM2HiS framework [5] is designed for monitoring individual Cloud platforms. The monitoring concept uses a centralised approach where the monitoring agents sends the monitored data to a central managing server. Furthermore, the communication of the monitored metrics were limited to one environment due to the communication mechanism being based on the Java Messaging Service (JMS) API and Apache ActiveMQ.

In this redesigned version, we propose a decentralised approach to monitoring where we introduce a hierarchy and divide the resources into clusters with each cluster having its own monitoring server as shown in Figure 1. The monitoring server processes the monitored raw data, applies the mapping rules and evaluate the SLA objectives to detect and inform on SLA violations. In addition, the communication mechanism is now based on a holistic message bus system, which offers us the opportunity to realise inter-Cloud monitoring and messaging. Details of this communication mechanism are presented in the next section.

4 Holistic Message Bus System

The term *holistic* in our context means a complete solution. Thus, we strive to design and implement a unified messaging system usable for communication at the different layers and levels in Clouds. This work should not be confused with the commercial "message bus" platform[4], which provides email services.

4.1 Communication Capability

Cloud computing involves different components and layers of interaction. To enable seamless communication, we aim to realise a mechanism with the following capabilities:

1. *Inter-application communication*: Enables applications executing in a Cloud or in multiple Clouds to interact and exchange information.
2. *Intra-layer communication*: This ability embodies the interaction among resources and their control entities in a Cloud layer.
3. *Cross-layer communication*: Enables the management of layers from the provider perspective. The cross layer communication supports resource allocation, load-balancing virtual machines and application deployment.
4. *Inter-Cloud communication*: Enables the outsourcing of resources or service executions between Clouds. It facilitates federation and Cloud bursting.
5. *Notification/Alerting*: Enables one way communication notifying or alerting users and administrators about the occurrence of particular events.

[4] `http://www.messagebus.com/`

From a provider's perspective, these messaging types cover the communication requirements for efficient management of Cloud provisioning. In the next section, we discuss the data interchange format, which is fundamental to our approach for achieving interoperability.

4.2 Data Interchange Format

The data interchange format is a technology-neutral method of organising and transferring data between entities such as applications, devices and organisations. It provides a means of serialising data for transmission over wire level protocols.

There are two widely used groups of data interchange formats (i) self-describing formats like *Extensible Markup Language (XML)*, *JavaScript Object Notation (JSON)*, and (ii) schema-based binary formats like *MessagePack*. The self-descri-bing formats are human readable and easy to understand but are not very compact, hindering efficient transmission. However, schema-based formats allow for efficient transmission but require receivers to be aware of the schema in advance of message delivery and are not human readable.

Thus, in our approach, we are applying a hybrid of self-describing and schema-based formats in order to achieve human readability and compactness for efficient transmission. To this end, we are investigating a means of effectively combining JSON and MessagePack. JSON was chosen over XML because it has been shown to have better performance [16].

To format different applications or data types, we group them into different categories. So, for each category, we define a unique data interchange format for formatting and serialising that category. In our current prototype, we implement two categories (i) infrastructure resource metrics and (ii) application data. The infrastructure resource metric format is used to structure the low-level monitored data from physical/virtual machines and the application data format is used to structure application specific data. The data interchange formats enable the message bus system to perform interoperable communications at different levels in Clouds.

4.3 Design and Implementation

The proposed message bus system is designed to realise the aforementioned communication capabilities. Figure 2 depicts an abstract view of the message bus system.

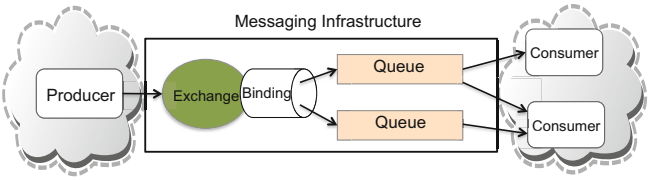

Fig. 2. Cloud Message Bus Overview

As shown in Figure 2, it consist of three components: (i) producer, (ii) messaging infrastructure and (iii) consumer.

Producer
The producer in some cases produces the data/message and in other cases receives the data/message for transmission from a source. It first analyses the data to identify its category in order to apply the correct data interchange format for structuring and serialising the message. The producer has a simple interface for receiving data and this makes it easy to integrate into different levels in Clouds. It is currently written in Java.

Messaging Infrastructure
The messaging infrastructure provides the functions of a message broker. It asynchronously delivers messages from producers to consumers (synchronisation decoupling). The producer does not need to know the nature or location of a consumer. It simply delivers its messages to the broker, which in turn routes them to the appropriate consumer (space decoupling). The broker therefore enables space, time and synchronisation decoupling [15]. This feature facilitates the necessarily loose relationship between a producer and a consumer, which is essential in a distributed system like Clouds.

In our implementation, we use the RabbitMQ broker API[5], which is based on the Advanced Messaging Queue Protocol (AMQP[6]). The broker consist of exchanges, bindings and queues. An exchange is the endpoint of a producer. It receives each message sent and forwards it to the appropriate queue, which is the endpoint of the consumer. The binding connects exchanges and queues, and message forwarding is based on the routing key contained in the message header.

Consumer
The consumer is the receiving end of the communication. It is responsible for receiving and de-serialising the message body. It is implemented in Java and provides a simple interface for easy integration into different Cloud levels.

Our message bus prototype is implemented in Java and uses MessagePack as the data interchange format. In the next section, we discuss a practical integration of the monitoring framework with the message bus system.

5 Integration of Monitoring with Messaging

This integration enables interoperability and facilitates efficient inter-Cloud provisioning management, for example, to enforce agreed SLA objectives in order to avoid violation penalties.

5.1 Use Case Scenario

This use case demonstrates the use of our integrated approach to manage a service outsourcing scenario involving two Clouds. As shown in Figure 3, Cloud A outsources the execution of an application (*A2*) to Cloud B.

[5] `http://www.rabbitmq.com`
[6] `http://www.amqp.org`

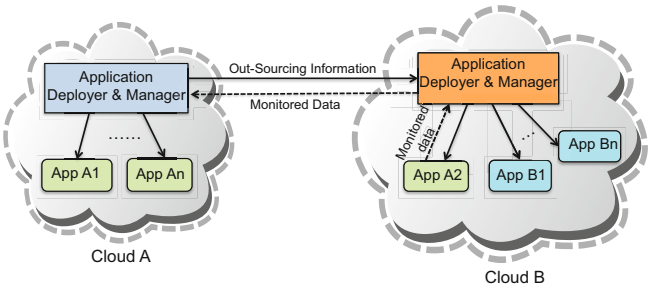

Fig. 3. Service Outsourcing Scenario Overview

In this scenario, the owner of application *A2* establishes an SLA with Cloud A for the execution of this application on his/her infrastructure. Thus, it is his/her responsibility to enforce the agreed SLA. However, if Cloud A does not have the resources to execute this application but Cloud B does, how can the provider transparently leverage the resources of Cloud B for this purpose? And how can the application execution on Cloud B be managed to guarantee the agreed SLA?

The holistic message bus addresses the first question by establishing an inter-Cloud communication channel and uses the application data interchange format to transmit the application for execution on Cloud B. The second question is addressed by the integration of monitoring with the message bus. In this manner, the monitoring framework monitors the resource consumption on Cloud B (as shown in Figure 3) and transmits the monitored data back to Cloud A for processing and enforcement of the agreed SLA terms for the *A2* application.

5.2 Summary

In summary, achieving interoperability among Cloud providers/platforms is essential to realise the full potential of Cloud Computing, especially with the growing market and rapid increase in the number of Cloud providers. Many types of service offering are emerging that can be used in varying constellations. This offers diverse opportunities, for example in cost reduction, by using a composite of low-priced services from different providers instead of subscribing for all the services from a single provider, which can lead to lock in and higher prices. This shows the importance of interoperability in advancing the business aspect of Cloud Computing.

The approach to interoperability proposed in this paper has the potential to shape the future of Cloud service provisioning. It is a practical solution that can unlock and exploit the benefits of Cloud bursting, public and private Cloud inter-operation and Cloud federation. The key advantage of our approach is that it uses emerging techniques, especially the data interchange format and the message bus.

We are currently working on a performance evaluation of the use case scenario described above on a Cloud incubator platform being developed in our research centre. The results of this evaluation will be reported in a follow-up paper.

6 Conclusion and Future Work

In this paper, we proposed an integrated approach to address the issues of interoperability in Clouds. Our goal is not only to achieve platform-neutral inter-Cloud communication but also to extend the capability of Cloud management to enable resource out/insourcing and Cloud bursting.

The proposed monitoring framework provides the ability to measure and determine, in real time, the resource consumption of services/applications executing on Cloud resources. It facilitates efficient management of resources and supports the enforcement of agreed SLA objectives.

The holistic message bus system represents a complete solution for communication in Clouds. It provides intra- and inter-Cloud communication, which can include interactions among applications and across Cloud layers. This is enabled by our prototype data interchange format. The integration of monitoring with this message bus system facilitates interoperability and interoperable Cloud management.

In the future, we will implementing a translation software that mediates between customers with proprietary message types. We are working on defining a mapping interface that would automatically adapt proprietary messages to our defined common data interchange format before transmission, reversing this process when a response is received. Furthermore, we intend to extend the message bus system with fault tolerance strategies to transparently handle error and fault situations.

Acknowledgement. The research work described in this paper was supported by the Irish Centre for Cloud Computing and Commerce, an Irish National Technology Centre funded by Enterprise Ireland and the Irish Industrial Development Authority.

References

1. Bahga, A., Madisetti, V.: A cloud-based approach to interoperable EHRs. IEEE Journal of Biomedical and Health Informatics 17, 894–906 (2013)
2. Bessis, N., Sotiriadis, S., Cristea, V., Pop, F.: Modelling requirements for enabling meta-scheduling in inter-clouds and inter-enterprises. In: 2011 Third International Conference on Intelligent Networking and Collaborative Systems (INCoS), pp. 149–156 (2011)
3. Boniface, M., Phillips, S.C., Sanchez-Macian, A., Surridge, M.: Dynamic service provisioning using GRIA SLAs. In: Di Nitto, E., Ripeanu, M. (eds.) ICSOC 2007. LNCS, vol. 4907, pp. 56–67. Springer, Heidelberg (2009)
4. Buyya, R., Ranjan, R., Calheiros, R.N.: InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services. In: Hsu, C.-H., Yang, L.T., Park, J.H., Yeo, S.-S. (eds.) ICA3PP 2010, Part I. LNCS, vol. 6081, pp. 13–31. Springer, Heidelberg (2010)

5. Emeakaroha, V.C., Brandic, I., Maurer, M., Dustdar, S.: Low level metrics to high level SLAs - LoM2HiS framework: Bridging the gap between monitored metrics and SLAs parameters in cloud environments. In: 2010 International Conference on High Performance Computing and Simulation (HPCS), pp. 48–54 (July 2010)
6. Emeakaroha, V.C., Calheiros, R.N., Netto, M.A.S., Brandic, I., De Rose, C.A.F.: DeSVi: An architecture for detecting SLA violations in cloud computing infrastructures. In: Proceedings of the 2nd International ICST Conference on Cloud Computing, CloudComp 2010 (2010)
7. Emeakaroha, V.C., Netto, M.A.S., Calheiros, R.N., Brandic, I., Buyya, R., De Rose, C.A.F.: Towards autonomic detection of SLA violations in cloud infrastructures. Future Generation Computer Systems 28(7), 1017–1029 (2012)
8. Hsieh, G., Chen, R.J.: Design for a secure interoperable cloud-based personal health record service. In: 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom), pp. 472–479 (2012)
9. Lewis, G.: Role of standards in cloud-computing interoperability. In: 2013 46th Hawaii International Conference on System Sciences (HICSS), pp. 1652–1661 (2013)
10. Lewis, G.A., Morris, E., Simanta, S., Wrage, L.: Why standards are not enough to guarantee end-to-end interoperability. In: Proceedings of the Seventh International Conference on Composition-Based Software Systems (ICCBSS 2008), pp. 164–173 (2008)
11. Nguyen, B.M., Tran, V., Hluchy, L.: A novel approach for developing interoperable services in cloud environment. In: 2013 International Conference on Information Networking (ICOIN), pp. 232–237 (2013)
12. Sotiriadis, S., Bessis, N., Antonpoulos, N.: Decentralized meta-brokers for intercloud: Modeling brokering coordinators for interoperable resource management. In: 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 2462–2468 (2012)
13. Stoegerer, C., Brandic, I., Emeakaroha, V.C., Kastner, W., Novak, T.: Applying availability SLAs to traffic management systems. In: Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC 2011), pp. 1501–1506 (2011)
14. Tovarnak, D., Pitner, T.: Towards multi-tenant and interoperable monitoring of virtual machines in cloud. In: 2012 14th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pp. 436–442 (2012)
15. Tran, N.L., Skhiri, S., Zimanyi, E.: Eqs: An elastic and scalable message queue for the cloud. In: 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), pp. 391–398 (2011)
16. Wang, P., Wu, X., Yang, H.: Analysis of the efficiency of data transmission format based on Ajax applications. In: 2011 International Conference on Information Technology, Computer Engineering and Management Sciences (ICM), vol. 4, pp. 265–268 (2011)
17. Williams, D., Jamjoom, H., Weatherspoon, H.: The Xen-Blanket: virtualize once, run everywhere. In: Proceedings of the 7th ACM European Conference on Computer Systems, EuroSys 2012, pp. 113–126 (2012)
18. Zhang, Z., Wu, C., Cheung, D.W.: A survey on cloud interoperability: taxonomies, standards, and practice. SIGMETRICS Perform. Eval. Rev. 40(4), 13–22 (2013)