RESEARCH PAPERS
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

2013 Special Number

DATA PORTABILITY AMONG PROVIDERS OF PLATFORM AS A SERVICE

Darko ANDROCEC

ABSTRACT

Platform as a service model has certain obstacles, including data lock-in. It is expensive and time-consuming to move data to the alternative providers. This paper presents data storage options in platform as a service offers and identifies the most common data portability problems between various commercial providers of platform as a service. There are differences among their storage models, data types, remote APIs for data manipulation and query languages. Representing data models of platform as a service and data mappings by means of ontology can provide a common layer to achieve data portability among different cloud providers.

KEY WORDS

data portability, platform as a service, Cloud Computing

INTRODUCTION

Cloud Computing is a new paradigm for the provision of computing infrastructure, platform or software as a service [4]. The most comprehensive definition of Cloud Computing is provided by NIST: “Cloud computing is a pay-per-use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” [11]. Platform as a service is a Cloud model where providers offer virtualized servers and associated services for running existing applications or developing and testing new ones [16].

However, this new business model for the provision of computing infrastructure has certain obstacles, including provider lock-in. The aforementioned problem is characterized by expensive and time-consuming migration of application and data to alternative Cloud providers, the inability or limited ability to connect to computing resources, applications and data outside the selected Cloud Computing service, and the dependence on a specific programming language used by the selected Cloud Computing vendor. Currently, each Cloud

Darko ANDROCEC - Faculty of Organization and Informatics, Pavlinska 2, HR 42000 Varazdin, Croatia, darko.androcec@foi.hr

provider develops its own specific technology solutions, remote application programming interfaces (APIs), and some even create new programming languages. If Clouds are not interoperable, it is difficult or even impossible to achieve collaboration among computing resources of different Cloud service providers, and possible migration to another provider is a complex and an expensive task. There are three main models of Cloud Computing: infrastructure as a service, platform as a service, and software as a service. The focus of this work is on issues of data portability among various offers of platform as a service.

The rest of the paper is structured as follows. In the next section, I list related work on Cloud Computing interoperability. After that, I present possible storage models for platform as a service layer. Thereafter, I identify the most common data interoperability problems in platform as a service model. Finally, I conclude and propose some solutions to found data interoperability problems.

RELATED WORK

Cloud Computing interoperability has recently become very active research field. Rodero-Merino et al. [18] propose a new abstraction layer for infrastructure as a service. This layer is closer to the service lifecycle and it provides automatic deployment, definition and management of services. Ajith and Maximilien [2] described their own Altocumulus middleware to homogenize different Cloud solutions and the associated Cloud best practice model. Bernstein and Vij [3] present their InterCloud Directories and Exchanges mediator to enable connectivity and collaboration among Cloud vendors. They define their Cloud Computing resources ontology by means of the Resource Description Framework (RDF). Merzky, Stamou and Jha [13] demonstrate a proof-of-concept of application-level interoperability among different Clouds and Grids by means of the SAGA-based implementation of MapReduce. They developed a range of Cloud adaptors for SAGA. MapReduce is a programming framework developed within Google and is used to simplify data processing across massive data sets, and SAGA is a programming interface which provides the ability to develop distributed applications in an infrastructure independent way. Ranabahu and Sheth [17] present the usage of well-established semantic technologies to overcome vendor lock-in issues in Cloud Computing. They distinguish four types of semantics for an application: data semantics (definitions of data structures, their relationships and restrictions), logic and process semantics (the business logic of the application), non-functional semantics (e.g. access control and logging) and system semantics (deployment descriptions and dependency management of the application). Buyya, Ranjan and Calheiros [5] present the vision, challenges and architecture of a utility-oriented federation of Cloud Computing environments. They advocate the creation of a federated Cloud Computing environment that supports dynamic expansion or contraction of capabilities. The reference architecture for semantically interoperable Clouds (RASIC) was proposed by Loutas et al. [12]. The architecture’s main aim is to enable the design, deployment and execution of services on top of semantically interlinked Cloud Computing offerings.

Promising research activities, related to Cloud Computing interoperability, are carried out in the European research projects: Mosaic [15], Cloud4SOA [6], Contrail, VisionCloud, REMICS, and MODAClouds. However, data portability issue among platform as a service providers is still unresolved.

STORAGE FOR PLATFORM AS A SERVICE

There are two main storage models in platform as a service: NoSQL and relational databases. NoSQL is next generation database that has some of the following characteristics: non-relational, distributed, open source and horizontally scalable [14]. More characteristics that often apply to this paradigm are: schema-free, easy replication support, simple API, eventually consistent / BASE (not ACID), a huge amount of data and more [14]. Relational databases can be run on the cloud as a virtual machine images or as a services. These databases are difficult to scale, but cloud providers attempt to address this issue. Now we will look at three prominent platform as a service offers (namely, Google App Engine, Microsoft Azure and Salesforce) and describe in detail their storage models.

Google App Engine has three options for data storage: App Engine Datastore, Google Cloud SQL and Google Cloud Storage. The App Engine Datastore is a schemaless object datastore [9]. The primary data repository for this offer is High Replication Datastore where data is replicated across multiple data centers using Paxos algorithm. The data store holds data objects named entities; each entity has one or more properties of one of supported data types; and each entity is identified by its kind and key. Google Cloud SQL [10] enables you to create, configure and use relational MySQL databases in Google's cloud. It has almost all of the capabilities and functionality of MySQL [1]. The Google Cloud Storage is an experimental service that provides storage for big objects and files (up to terabytes in size).

There are three main storage offerings on the Azure platform [7]: Local Storage, Windows Azure Storage, and SQL Database. Local Storage provides temporary storage for a running application and it represents a directory that can be used to store files. Windows Azure Storage consists of blobs (storage of unstructured binary data), tables (a schemaless collection of row like entities, each of which can contain up to 255 properties) and queues (storage for passing messages between applications) that are accessible by multiple applications. SQL Database is based on SQL Server technology and provides relational database for the Azure platform.

In Salesforce, an organization is the equivalent of a database, but with built-in user identity and security [8]. Objects are similar to tables in relational databases, they contain fields and records. Objects are related to other objects by using relationship fields, such as lookup and master-detail relationships, instead of primary and foreign keys. There are two types of objects: standard objects (predefined, created automatically by Salesforce) and custom objects (objects that you create in your organization).

IDENTIFIED DATA PORTABILITY PROBLEMS

The first identified problem is a difference between data storage models of various commercial providers of platform as a service. For example, it is difficult or even impossible to move data from a NoSQL model of one provider to a SQL model of another platform as a service provider. Even if we choose the same models (e.g. SQL) in two various offers, these models will still have significant differences due to provider's design and used technology. For example, each provider supports their own set of data types. Data types differ in name, value space, permitted range of values, precision of data etc. Some offers also have predefined standard objects or tables, e.g. Salesforce list standard objects in their documentation (some object/table names are reserved) and it also adds some standard fields to any new custom object (object created by user).

Data import or export is often complicated. Most providers offer only basic CSV or XML exports (list of columns and row data), and from them you can’t determine data types, primary keys, possible relationships between tables (e.g. foreign keys) etc. You must use remote APIs of cloud providers to get that information. APIs are not standardized, so you need to cope with different functions, input and output parameters and different means to access remote API functionalities (for example, by using libraries for programming languages and/or SOAP or REST web services). Various platform as a service providers also use their own versions of data query languages. Salesforce provides the Salesforce Object Query Language (SOQL) and Salesforce Object Search Language (SOSL). Google Query Language (GQL) is a language for retrieving entities or keys from the Google App Engine data store, and its syntax is similar to that of SQL. SQL Azure uses T-SQL as the query language.

CONCLUSION

Cloud Computing is a new paradigm for the provision of computing infrastructure, platform or software as a service that enables significant cost reduction and flexibility. This is a strong motive for many organizations from the public and private sector to turn to Cloud Computing services. Lack of established Cloud Computing standards still presents a challenge to organizations interested in cloud services. When an organization chooses a specific cloud service provider, it also gets the vendor’s specific protocols, standards and tools, making a potential future migration complex and costly. The focus of this paper is on issues of platform as a service data portability.

I found that there are two main storage models in platform as a service offers: NoSQL and relational databases. The first identified data portability problem is a difference among data storage models of various commercial providers of platform as a service. The storage models of three prominent platform as a service providers (Google, Salesforce and Microsoft) are presented in the third section. Furthermore, each provider supports their own set of data types that differ in name, value space, permitted range of values, precision of data etc. The next issue lays in the fact that platform as a service providers use their own data query languages. Remote APIs for data manipulation are not standardized, and most providers offer only basic CSV or XML exports.

The data portability problems can be solved by using unified model and mapping. I Plan to develop an ontology that identifies remote API operations for data manipulation and data mappings among the heterogeneous APIs. The synonymous API operations of various platform as a service vendors can have different types and numbers of input and output parameters. Furthermore, platform as a service offerings often use proprietary and non-standard databases (relational and non-relational). Representing these data models by means of ontology can provide a common layer for information exchange.

REFERENCES

1. About Google Cloud SQL, https://developers.google.com/cloud-sql/docs/introduction, Accessed 20.02.2013.
2. AJITH, R., MAXIMILIEN M. 2009. A Best Practice Model for Cloud Middleware Systems. Proceedings of the Best Practices in Cloud Computing: Designing for the Cloud workshop in ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA). Orlando FL, USA. pp.41-51.

3. BERNSTEIN, D., VIJ, D. 2010. Intercloud Directory and Exchange Protocol Detail using XMPP and RDF. In *The 6th World Congress on Services*. Miami, USA, pp. 431-438.
4. BUYYA R., YEO C S., VENUGOPAL S., BROBERG J., BRANDIC I. 2009. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems, Amsterdam, Netherlands, pp. 599-616.
5. BUYYA, R., RANJAN, R., CALHEIROS R.N. 2010. InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services. In: *Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing*. Germany, Springer.
6. D1.2 Cloud4SOA Cloud Semantic Interoperability Framework. 2.6.2011, http://www.cloud4soa.eu/sites/default/files/D1.2_Cloud4SOA%20Cloud%20Semantic%20Interoperability%20Framework.pdf, Accessed 14.05.2012.
7. Data Storage Offerings on the Windows Azure Platform, http://social.technet.microsoft.com/wiki/contents/articles/1674.data-storage-offerings-on-the-windows-azure-platform.aspx, Accessed 20.02.2013.
8. Database.com Workbook, 2013, http://www.salesforce.com/us/developer/docs/workbook_database/workbook_database.pdf, Accessed 20.02.2013.
9. Datastore overview, https://developers.google.com/appengine/docs/java/datastore/overview, Accessed 20.02.2013.
10. Google Cloud SQL, https://developers.google.com/cloud-sql/, Accessed 20.02.2013.
11. LINTHICUM, D. S. 2009. Cloud Computing and SOA Convergence in Your Enterprise, Addison-Wesley, New York, USA.
12. LOUTAS, N., PERISTERAS V., BOURAS T., KAMATERI E., ZEGINIS D. 2010. Towards a Reference Architecture for Semantically Interoperable Clouds. In: *2010 IEEE Second International Conference on Cloud Computing Technology and Science*, pp. 143-150.
13. MERZKY, A., STAMOU, K., JHA, S. 2009. Application Level Interoperability between Clouds and Grids. Workshops at the Grid and Pervasive Computing Conference, pp. 143-149.
14. NoSQL, http://nosql-database.org/, Accessed 20.02.2013.
15. PETCU, D., MACARIU, G., PANICA S., CRACIUN C. 2012. Portable Cloud applications—From theory to practice. Future Generation Computer Systems.
16. Platform as a service definition, 2010, [online], http://searchcloudcomputing.techtarget.com/definition/Platform-as-a-Service-PaaS
17. RANABAHU, A., SHETH, A. 2010. Semantics Centric Solutions for Application and Data Portability in Cloud Computing. In: *2nd IEEE International Conference on Cloud Computing Technology and Science*, pp. 234-241.
18. RODERO-MERINO, L., VAQUERO, L.M., GIL, V., GALAN, F., FONTAN, J., MONTERO, R. S., LLORENTE, I.M. 2010. From infrastructure delivery to service management in clouds. Future Generation Computer Systems 26, pp. 1226-1240.