Journal of Engineering and Applied Sciences 14 (21): 7899-7912, 2019
ISSN: 1816-949X

Toward a Social Networking Context: Managing and Processing Data over Multi-Cloud Environment

Ahmed Barnawi, Abdullah Al-Barakati, Fouad Bajaber, Maraam Al-hafidy,
Abdullah Almalais and Seyed Buhari
Department of Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
ambarnawi@kau.edu.sa

Abstract: Cloud computing environment is one of the widely growing software deployment platforms in the world of technology. There is a continuous increase in using the cloud-based storage as service from developers. Cloud storage services provide numerous advantages such as high scalability, availability and pay-as-you-go cost model. In addition, cloud providers are offering different options for storage services. For example, Amazon provides S3 service as a scalable, durable and available distributed object store. Azure offers SQL databases as traditional SQL databases. In this domain, one of the biggest challenges is the interoperability among different storage systems provided by the various cloud providers; it is due to the lack of the uniform methods for accessing interfacing and managing the stored data. In this research we extend our previous work to develop a new framework to bridge the some of the gaps between the social web and the social enterprise worlds using the business process management techniques and concepts. We designed and developed a social coordination approach that uses the social networks between persons, tasks and machines to recommend corrective actions in response to specific conflict patterns. In this study, we have presented our approach to address such a challenge by building a system that enables the developers to manage the data hosted in different storage systems of various cloud providers for a single point of interface. We show, through evaluation that our approach is able to provide high usability integrity and facilitate the navigation and interoperability among the different cloud providers. We find out that the performance of MCloud is satisfying and reasonable. In addition, the developers reveal that MCloud has helped them to manage the hosted data in different cloud providers.

Key words: Cloud computing, data management interoperability, vendor lock-in, multi-cloud, storage systems, SQL databases

INTRODUCTION

Cloud computing is a promising technique that becomes one of the hottest core technical topics in the modern software development era and has greatly changed the modern IT industry (Yang and Jia, 2014). The National Institute of Standards and Technology defined the cloud computing as "a model for enabling convenient, resource pooling, ubiquitous, on-demand access which can be easily delivered with different types of service provider interaction" (Singh *et al*., 2016) as shown in Fig. 1. Cloud storage is one of the most essential services

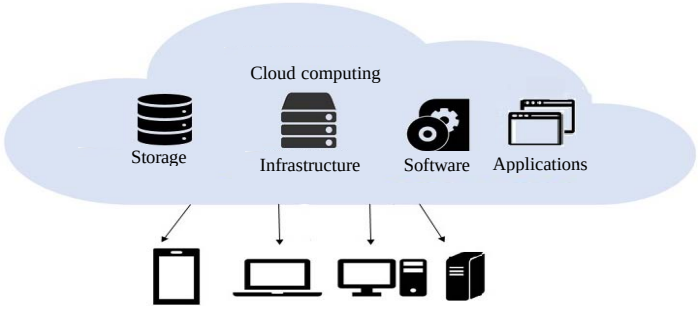

Fig. 1: Cloud computing environment

Corresponding Author: Ahmed Barnawi, Department of Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia, ambarnawi@kau.edu.sa

of cloud computing which offers storage-as-a-service that allows data owners to store their data in the cloud (Rafique *et al*., 2017).

The different types of data are generated as structured data, unstructured data or semi-structured data. Structured data are relatively easy to input, query, store and analyze (e.g., numbers, words and dates). Semi-structured data are the data that do not follow a conventional database system. Semi-structured data may be in the form of structured data that are not organized in relational database models. Unstructured data are data that do not follow a specified format (e.g., videos and images) (Hashem *et al*., 2015; Rusu *et al*., 2013). There is a requirement to deal with various types of storage systems that hold these different types of data.

In practice, 4 different data storage are common across the cloud vendor (Livenson and Laure, 2011), we have explained these types in more detail:

- Blob storage is useful for storing unstructured data
- Table storage is preferred for semi-structured data also known as "NoSQL"
- Queue storage which is often used for guaranteed item delivery that is the at-least-once delivery of work items
- Relational table storage is based on traditional SQL databases useful for structured data

The requirement is not only storing the data but also, managing them. Moreover, the availability of different types of data storage from various providers is a point of strength in the cloud environment which allows the developers to choose from them based on the requirements of the applications. We found out, in our experiment explained that 62% of developers depended on using storage services from various cloud providers to build their applications. However, this situation has increased the burden upon the developers to manage their hosted data in multi cloud platforms, especially, in case there is a lack of interoperability between the cloud providers.

To address these challenges, we proposed building a unified interface to manage the data hosted in different storage systems in multi-cloud, the proposed system is called Manage Cloud (MCloud). MCloud standardizes interfaces for Blob storage, Table storage, Queue storage and relational table storage, to allow developers to be independent from cloud vendors and to provide a transparent way for them to access and manage their data. For Blob storage, MCloud supports Amazon Simple Storage Service and IBM Bluemix Object Storage. In addition, it supports DynamoDB for Table storage and Amazon Simple Queue Service for the Queue storage type. Finally, it supports Amazon Relational Database Service and Azure SQL database for relational table storage.

Problem statement: Different cloud providers such as Amazon and Microsoft Azure have supported the previous storage systems and offered them to their customers. The user should use the front end or API to get access to or store the data. Unfortunately, those cloud providers have been incompatible with each other and created these services with different APIs and interfaces. Therefore, the incompatibility in standards and formats while getting access to the cloud has become a big issue causing a vendor lock-in problem (Kolb and Rock, 2016). Consequently, one of the major challenges in cloud computing environment is cloud interoperability. Interoperability is defined as the ability of heterogeneous systems to work and interact together. For cloud computing interoperability means the ability for multiple cloud providers to research together with minimal or no user's effort. However, most of the cloud providers have built their services without consideration of interoperability (Arunkumar and Venkataraman, 2015; Alomari *et al*., 2014).

Some developers are more willing to use a particular storage type of a specific provider depending on the service costs and features. That is why the developers use multi-cloud. The experiment indicated that 75% of developers who tended to access and manage their data, faced some problems dealing with different cloud interfaces and some difficulties in managing the hosted data such as the vendor lock-in problem. Furthermore, they experienced a lack of interoperability between these different cloud providers. In this domain as interoperability became a big issue, 62% of developers executed some tasks sequentially from one cloud provider to another as shown in our experiment.

Storage as a service: Cloud storage is one of the most important cloud computing services that the providers offer to their consumers. In addition, it is one of the fundamental services used by the companies to store large amounts of data every day (Bocchi *et al*., 2014). Moreover, it is used in many web applications such as online social networks and web portals to serve the clients all over the worldwide (Liu *et al*., 2017).

Cloud storage is a model of online storage where data is stored in virtualized storage pools that are generally hosted by storage service providers. Cloud storage provides a simple and scalable way to store, access and share data (Yang and Jia, 2014). Consumers buy or lease storage capacity from cloud providers on a pay-as-you-go business model.

In addition to hosting their data on the cloud, consumers can avoid the high initial expenses of setting up infrastructure, large equipment and daily maintenance (Yang and Jia, 2014). Now a days, many cloud services are available as storage services such as Amazon RDS and Azure SQL database service. Cloud storage has many essential characteristics and provides many solutions that attract an increasing number of potential users they are as the following (Bocchi *et al*., 2014; Li *et al*., 2016):

- Cloud storage does not involve any set up cost and thus, it helps customers to save costs of buying and maintaining expensive hardware
- Cloud storage service provides a convenient and reliable way to store and share data from anywhere, on any device and at any time
- Cloud storage service is provided by cloud providers on a simple 'pay-as-you-go' billing model
- Providing users with different types of system storages such as Blob storage, Table storage, Queue storage and relational table storage

In practice, four different types of data storage are common between the cloud vendors each one of them is described below.

Binary Large Object (BLOB) storage: It is a type of storage for unstructured data that can include video, audio, photos and archived email messages; it enables the users to store data on virtualized disks and access them anytime from any point on the internet. Blob storage, also known as object storage (Ranjan *et al*., 2015). Examples are Amazon Simple Storage Service (S3) and IBM Bluemix Object Storage.

Table storage: Table storage is preferred for semi-structured data which are tabular data that are not organized in relational database models (Hashem *et al*., 2015). Table storage is a service that stores structured NoSQL data on the cloud, providing a key/attribute store with a schemaless design where value can be gotten quickly by using the key. Tables store data as collections of entities where entities are similar to rows each entity has a primary key and a set of properties. A property is a name/value pair similar to a column. An example of table storage is Amazon DynamoDB service.

Queue storage: The Queue storage service connects the components or applications through the cloud. Queue storage stores and exchanges messages between the components these components are on the cloud or on-premise. Each message has a small body and some attributes such as time-to-live which you can use for configuring the service. An example of Queue storage is Amazon SQS.

Relational table storage: Relational table storage is based on traditional SQL databases it is used to store structured data including numbers, words and dates. Relational database systems store index and query data in an efficient manner. More recently, several relational table storages have emerged such as Microsoft Azure SQL and Amazon relational database service (Hashem *et al*., 2015; Narasayya *et al*., 2015).

Literature review: In the past, several researches by Kolb and Rock (2016), Anonymous (2017a-k), Vijaya and Neelanarayanan (2015), Chung *et al*. (2015), Yin and Wang (2015) and Sellami *et al*. (2016) related to the unified cloud interfaces were published whether as independent ones or as a part of a broader approaches. The majority of them has focused on the infrastructure provisioning model or the unified interfaces for application deployment and management among cloud platforms. Furthermore, existing approaches for PaaS focused on supporting a unified deployment of applications. In addition, the recent researches by Kolb and Rock (2016), Vijaya and Neelanarayanan (2015) and Sellami *et al*. (2016) have focused more on the management capabilities for applications in the cloud such as developing, deploying and migrating multiple data stores. In this section provide an overview of the related works by classifying them into: the way they solve the problems of standards or proxies by the level of service model the approaches work on and demonstrating how our research differs and contributes to the existing approaches.

Standards: The standards approach is proposed to overcome the vendor-in lock and to deal with the data stored in a cloud environment. A standard is defined as "a definition or format that has been approved by a recognized standards organization or is accepted as a de facto standard by the industry" (Anonymous, 2017a-k). The significance of standards has come from the fact that it allows a combination of the hardware and software from different companies to be used together, especially, standard user interfaces that can make it much easier to learn how to use new applications. Several standards have been proposed in many areas such as programming languages, operating systems, data formats and communications protocols (Anonymous, 2017a-k).

Some standardization organizations propose standards to overcome the vendor-in lock, support the interoperability and to deal with the data stored in a cloud environment. Storage Network Industry Association (SNIA) proposed Cloud Data Management Interface (CDMI) which is an ISO/IEC standard that enables cloud solution providers to meet the growing need for data interoperability in the cloud (Anonymous, 2017a-k). Open Cloud Computing Interface (OCCI) is classified as a standardized approach across different cloud providers. It provides a set of specifications for cloud tasks such as deployment, dynamic scaling and monitoring across different cloud providers (Vijaya and Neelanarayanan, 2015).

OCCI is a rotocol and API for all kinds of management tasks, suitable to create a remote management API for IaaS, PaaS and SaaS Model-based services (Anonymous, 2017a-k). From a user's perspective, standards are extremely important in the computer industry because they allow interoperability via. the combination of products from different providers (Anonymous, 2017a-k). However, the biggest obstacle to standards industry is that most standard proposals suffer from the lack of acceptance and participation by cloud providers (Vijaya and Neelanarayanan, 2015) while MCloud does not depend on cloud provider's acceptance and participation.

Proxies: A proxy server is a server (computer or a software system) that acts as an intermediary between an endpoint device such as a computer and another server from which a user or client makes a request for a service (Anonymous, 2017a-k). Another approach to integrating multiple CSPs is to use proxy servers which act as an intermediary among multiple CSPs providing transparent access and gathering data from multiple CSPs (Chung *et al*., 2015). For example, CDMI-compatible proxy which offers a transparent integration layer on top of both cloud systems and local storage infrastructures. However, CDMI-compatible proxy supports only two types of storage mentioned in study 3, Blobs and Queues storages (Livenson and Laure, 2011).

RACS is introduced by Abu-Libdeh *et al*. (2010) as a proxy that transparently spreads the storage load over many providers and allows customers to avoid vendor lock-in. Scalia Papaioannou *et al*. (2012) uses a proxy between the client and the cloud storage providers to enable the data owners to avoid vendor lock in and satisfy certain availability and durability constraints in a cost-effective way. Moreover, the proxy can gather user data to and from multiple CSPs, providing transparent access for users. However, the biggest obstacle to this approach is a single point of failure (Chung *et al*., 2015). A Single Point of Failure (SPOF) is a potential risk in which one fault or malfunction causes an entire system to stop operating (Anonymous, 2017a-k).

Approaches at SaaS level: Software as a Service (SaaS) is a service model in the cloud computing that enables the consumer to use the provider's applications running on a cloud infrastructure. Depending on this service model has enhanced the emergence of some applications to avoid vendor lock in problem in cloud computing environment and enables the user to manage the files stored in different cloud service providers in a transparent manner. One of these applications is Cloudfuze which is a centralized interface that enables the user to access and manage the files stored in different cloud providers (Yin and Wang, 2015).

In addition, Cyrus Chung *et al*. (2015) is a client-defined architecture that integrates multiple CSPs into one unified cloud. Cyrus enables the user to access and manage the files stored in different cloud service providers and allows them to share files and specify their desired performance levels. The limitation of these applications is that they only support Blob storage type while MCloud supports the four types of storage system mentioned.

Approaches at PaaS level: Platform as a Service (PaaS) is a service model in the cloud computing in which the provider provisions the software development tools and programming languages to enable the consumer to develop his/her own software. Depending on this service model, some approaches have grown to avoid vendor lock in problem in cloud computing environment while the developer build his/her own software. SimpleCloud is an API that allows using the storage services independent of cloud platforms (Vijaya and Neelanarayanan, 2015). It allows the developers to write the portable code that can interoperate with multiple cloud vendors, however, it only supports PHP language for web applications. MCloud provides a common interface to access and manage the hosted data in the multiple cloud providers, regardless of the programming language the developers use to build their applications. CDPort Alomari *et al*. (2014) proposes a common data model and a standardized API for NoSQL database. Moreover, CDPort supports the transformation and exchange of data that is stored on NoSQL databases. It provides only NoSQL storage type while MCloud supports the 4 types of storage system mentioned.

Some researches have focused on assisting the developers to manage their applications. Such as

Sellami *et al*. (2016) which facilitates the developer's task and enables the development of applications using multiple data stores. The developer can use this approach to develop, deploy and migrate multiple data stores to the applications in cloud environments. A unified interface for application deployment and management among cloud platforms is presented by Kolb and Rock (2016). Openshift is a platform for developers to build, test, deploy and run cloud applications, it supports no-lock-in at PaaS level. By using openshift, the developer can focus only on designing and coding as it handles all the infrastructure and middleware management (Vijaya and Neelanarayanan, 2015). MCloud supports no-lock-in at storage-as-a-service level.

All the studies in previous paragraph propose different solutions to assist the developers to manage their applications over multiple platforms. Furthermore, our research focuses on supporting the developers to manage the data hosted in a cloud environment it provides storage as a service to the developers. Therefore, a developer can access the storage from different cloud providers, provision it and view its details and manage it. Table 1 lists several approaches related to addressing the question of thisstudy and highlights some of their limitations.

Table 1: Related works and its limitations

Solution	Approach	Limitations
Data Management Interface (CDMI)	Standards	Standards experience a lack of acceptance and participation by cloud providers
Open cloud computing interface	Standards	
CDMI-compatible proxy	Proxies	Does not support NoSQL storage and relational table storage
RACS	Proxies	This approach is a single point of failure
Scalia	Proxies	
Cloudfuze	Approaches at SaaS level	Only supports Blob storage type
Cyrus	Approaches at SaaS level	
SimpleCloud	Approaches at PaaS level	Language dependent, only supports applications written in PHP language
CDPort	Approaches at PaaS level	It provides only NoSQL storage type
Openshift	Approaches at PaaS level	Supports no-lock-in at PaaS level

MATERIALS AND METHODS

Mcloud system: MCloud is a system that provides different storage services across multiple cloud providers. This platform supports all types of storage in cloud computing environment (Blob storage, Queue storage, Table storage and relational table storage). MCloud reinforces the interoperability among cloud providers in a transparent way, especially, for the user who uses different storage systems of various cloud providers. This platform has three main goals:

- Building universal methods to manage different storage systems in the cloud
- Developing standardized methods for accessing interfacing and querying the stored data in different storage services
- Providing the user with a transparent way in order to handle their data stored in different storage systems

Mcloud supports the four-different data storage services as shown in Fig. 2 and it is described in the following:

Blob storage: MCloud supports two of the most common Blob storages: Amazon Simple Storage Service (S3) and IBM Bluemix Object Storage.

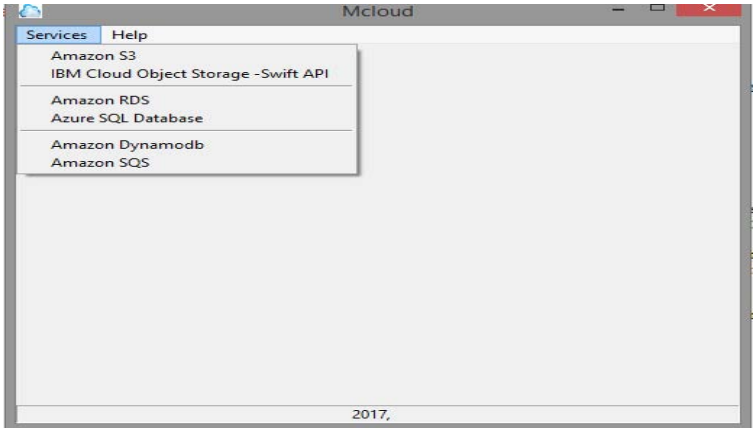

Fig. 2: Home page of MCloud

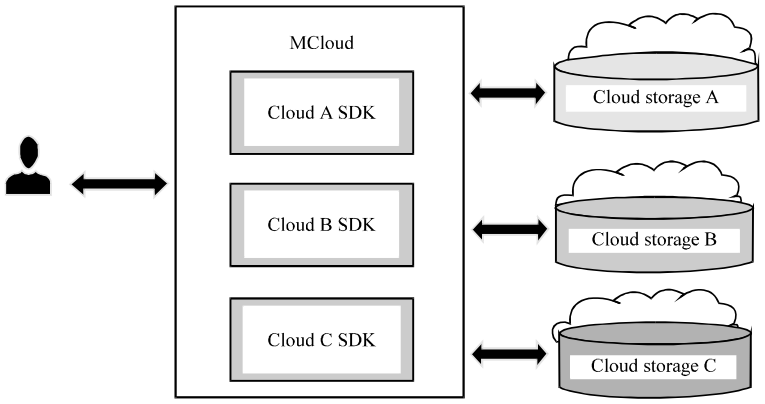

Fig. 3: MCloud data access mechanisms

Table storage: MCloud facilitates the control over Amazon DynamoDB service.

Queue storage: MCloud enables the user to manipulate Amazon Simple Queue Service (SQS).

Relational table storage: MCloud supports two of the most common relational table storages: Amazon Relational Database Service (RDS) and Azure SQL database.

MCloud design: The proposed interface aims to unify the core management functions of the data hosted in multi-cloud. We focus completely on the creation of a management interface which covers all storage types mentioned. MCloud uses a layer for wrapping three SDKs of different clouds into a common interface to get access to the data hosted in different storage systems as shown in Fig. 3.

To achieve the desired results, we built the interface using the SDKs of some cloud providers. First, Boto3 SDK provided by Amazon was used to enable MCloud to support four services:

Amazon S3 Amazon DynamoDB service Amazon RDS and Amazon SQS. Second, Azure SDK provided by Microsoft Azure was used to enable MCloud to support Azure SQL database. Third, using the Swiftclient SDK to enable MCloud to support object storage service provided by IBM Bluemix.

MCloud implementation: We implemented the MCloud system in Python language using the three SDKs mentioned previously. We built an MCloud package that contained classes and imported three SDKs: Boto3, Azure and Swiftclient as shown in Fig. 4. The package diagram shows how the various classes are grouped into packages. There is one top-level class which is MCloud App and that allows the system to be run. MCloudApp class depends on the MCloud package which contains the classes that represent the system as a whole and also import Boto3, Azure and Swiftclient packages. Boto3 is Software Development Kit (SDK) for Amazon Web services which allows Python developers to write the software that enables the use of services like Amazon S3 and Amazon SQS (Anonymous, 2019a-c). Azure SDK is the Microsoft Azure package and it does not contain any code in itself. It installs a set of packages that provide Microsoft Azure functionality (Anonymous, 2019a-c). Swiftclient package is SDK for IBM Bluemix object storage service (Anonymous, 2019a-c).

Cloud storage services supported by MCloud: We provide an overview of the six services that MCloud supports. Table 2 provides a summary of the services supported by MCloud for each storage system.

Amazon simple storage Service (S3): Amazon Simple Storage Service (Amazon S3) is an online service provided by Amazon Cloud provider and used to store and retrieve any amount of data at any time and from anywhere. The S3 is characterized by many features that make it an attractive storage service for developers such as a pay-as-you-go, fully scalable, fast and reliable service (Beach, 2014; Anonymous, 2017a-k) Amazon S3 stores data as objects within buckets. An object consists of a file and optionally any metadata that describes that file. To store a file in Amazon S3, upload the file to a bucket. Amazon S3 account can have one or more buckets. A user can create, delete a bucket and view the list of folders and files stored in the bucket. Amazon S3 enables the user to manage each bucket, delete a folder and view the list of files stored in the folder in addition to uploading, downloading and deleting files. Figure 5 shown the interface of S3 in case some buckets were created previously.

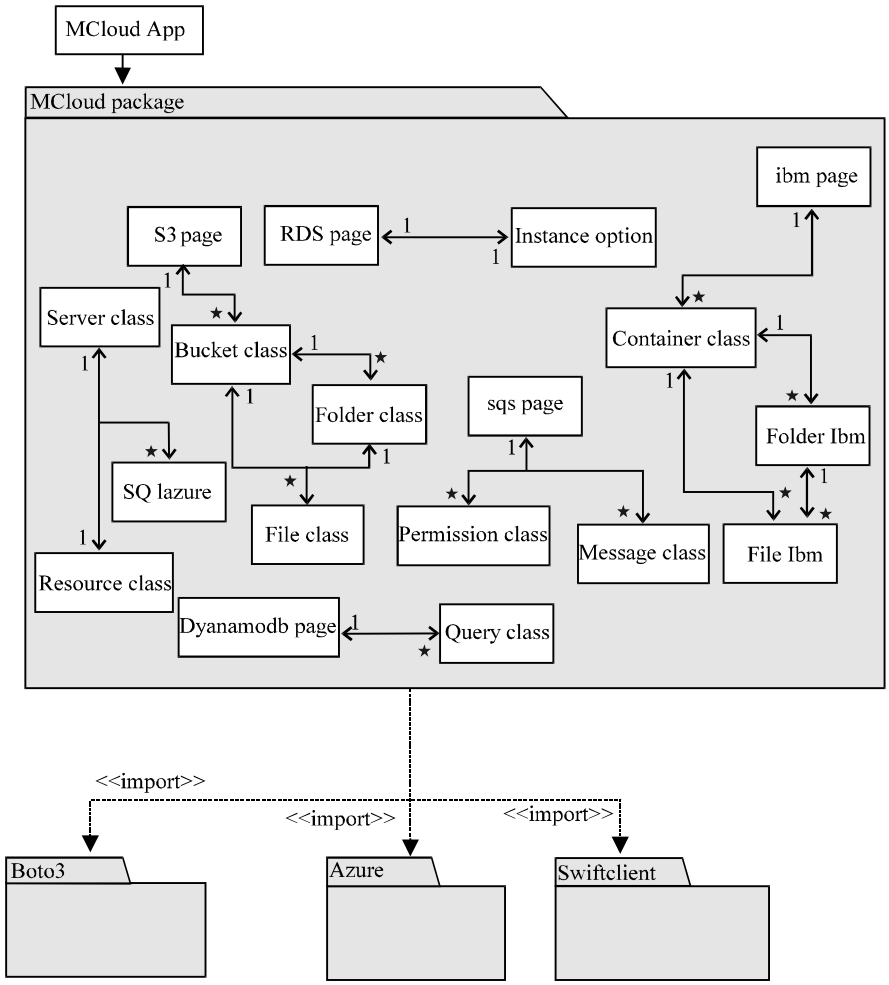

Fig. 4: Package diagram for MCloud system

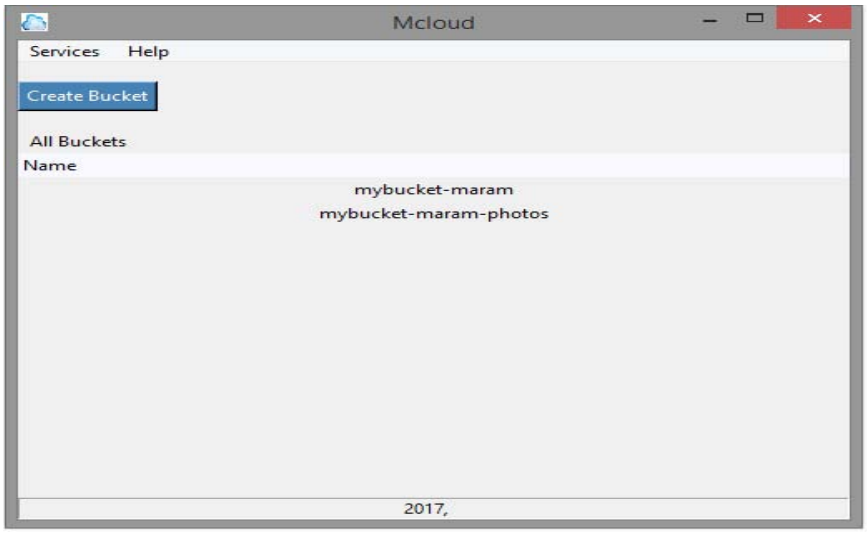

Fig. 5: Window in case some buckets wer created previously

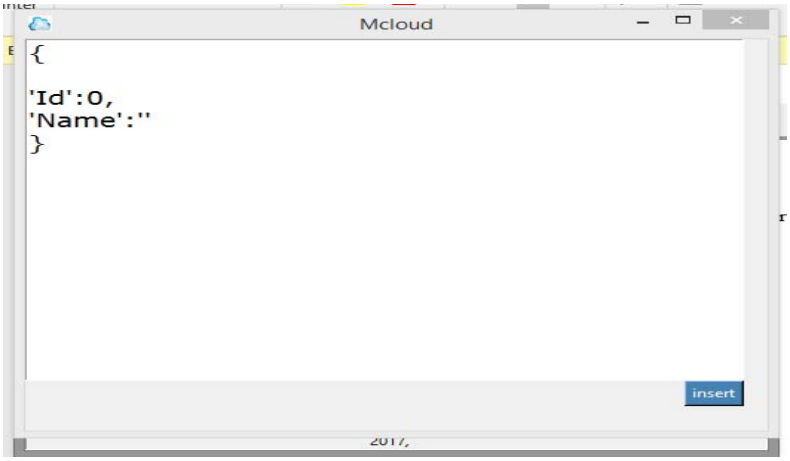

Fig. 6: Window for creating an item

Table 2: A summary of the services provided by MCloud

Storage system	Services
Amazon S3	Create a bucket
	Upload a file
	Download a file
	Create a folder
	Delete a file
	Delete a bucket
	Empty a bucket
	Delete a folder
	View a bucket contains
	View a folder contains
Amazon DynamoDB	Create a table
	Delete a table
	Insert items in table
	Update an item
	Delete items
	Query for items
Amazon RDS	Create a DB instance
	View a DB instance details
	Modify a DB instance
	Delete a DB instance
Amazon SQS	Create a Queue
	Send a message
	View/Delete a message
	View a Queue details
	Purge a Queue
	Configure a queue
	Delete a Queue
	Create a permission
	Delete a permission
Azure SQL Database	Create SQL databases
	Create a resource groups
	View items of a resource group
	Delete a resource groups
	Create a server
	Set a server firewalls
	Delete a server
	Delete SQL database
	View a database on a server
IBM Bluemix object storage	Create a container
	View files stored in a container
	Upload a file
	Download a file
	Create a folder
	Delete a file
	Delete a folder
	Delete a container

DynamoDB (NoSQL database) service: Amazon DynamoDB is a fully managed NoSQL database service used to create a database table that can store and retrieve any amount of data and serve any level of request traffic. Amazon DynamoDB service is characterized by a fast and predictable performance with seamless scalability. Amazon DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity while maintaining consistent and fast performance (Anonymous, 2017a-k). DynamoDB stores data in tables. Each table contains multiple items and an item is defined as a group of attributes that are uniquely identifiable among other items. Figure 6 shown the window for creating an item in DynamoDB service.

Relational Database Service (RDS): Amazon Relational Database Service (Amazon RDS) is a web service that sets up, operates and scales a relational database in the cloud. It provides cost-efficient, resizable capacity for a relational database and manages the common database administration tasks. DB instance is the basic building block of Amazon RDS (Anonymous, 2017a-k). The window for details of a DB instance shown in Fig. 7.

Amazon Simple Queue Service (SQS): Amazon Simple Queue Service (Amazon SQS) is a queue hosting service for storing messages as they travel among applications or micro-services. Amazon SQS is reliable and highly-scalable hosted queue service. The main components of Amazon SQS are Queue and messages in the queues. Amazon SQS provides the following features: access to messages and high availability for producing and consuming messages, access control of messages and who can send/receive messages to/from a queue (Anonymous, 2017a-k). Create a queue shown in Fig. 8.

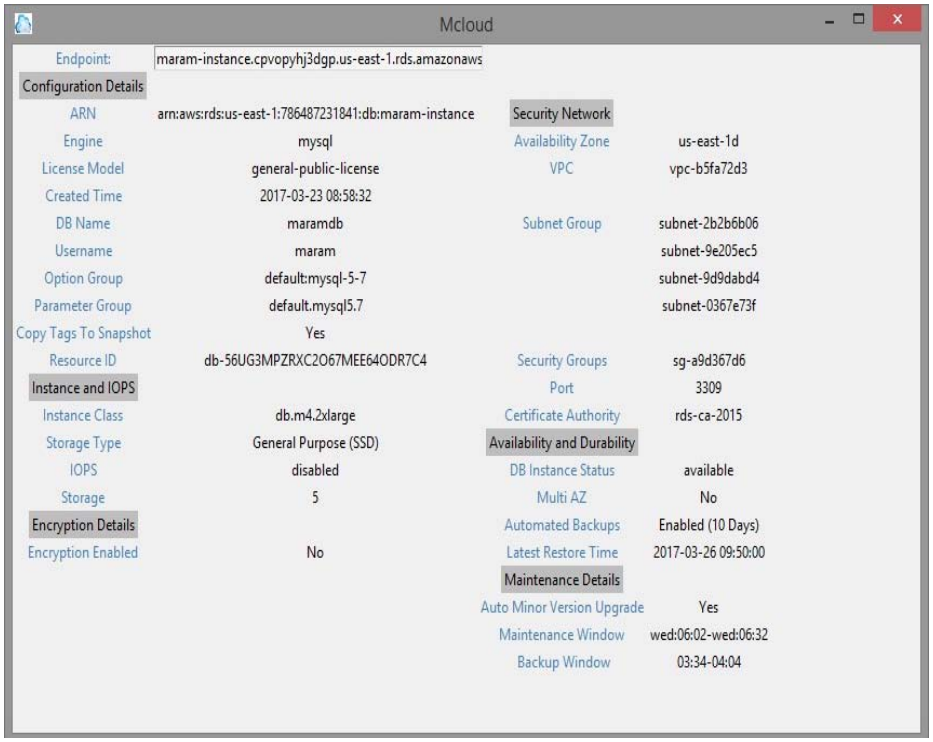

Fig. 7: Window for details of a dB instance

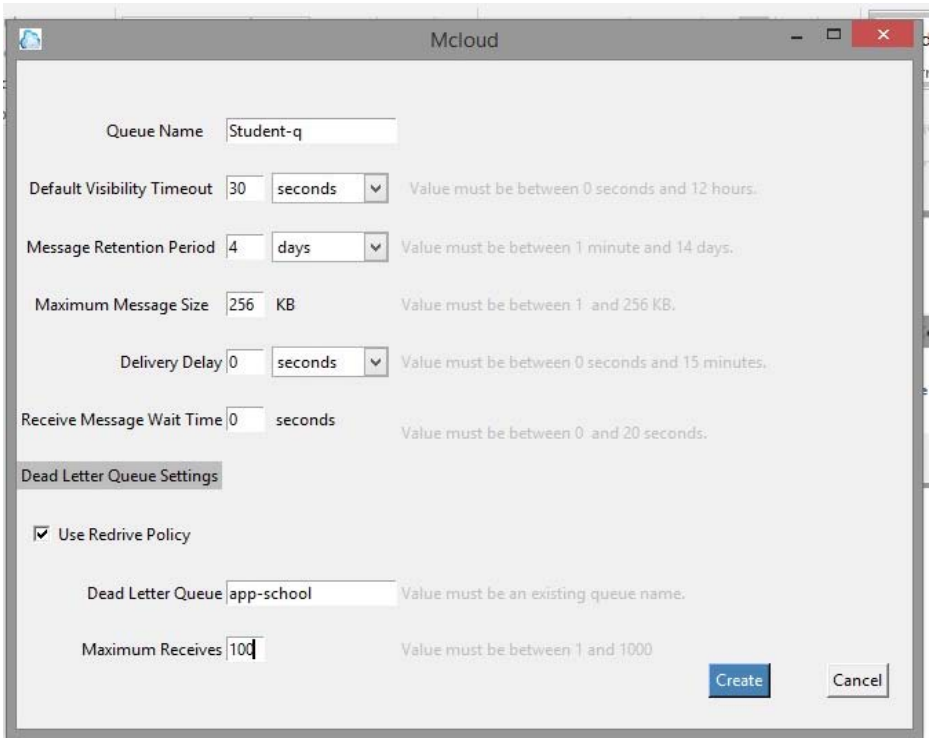

Fig. 8: Window for creating a queue

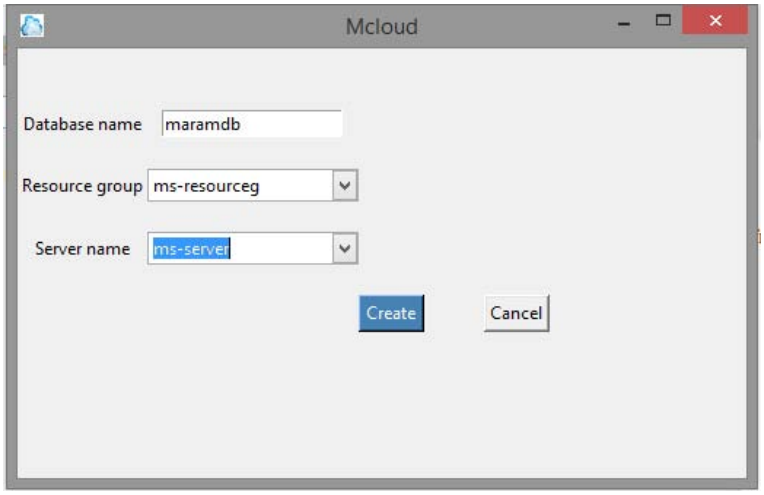

Fig. 9: Window for cerating SQL database

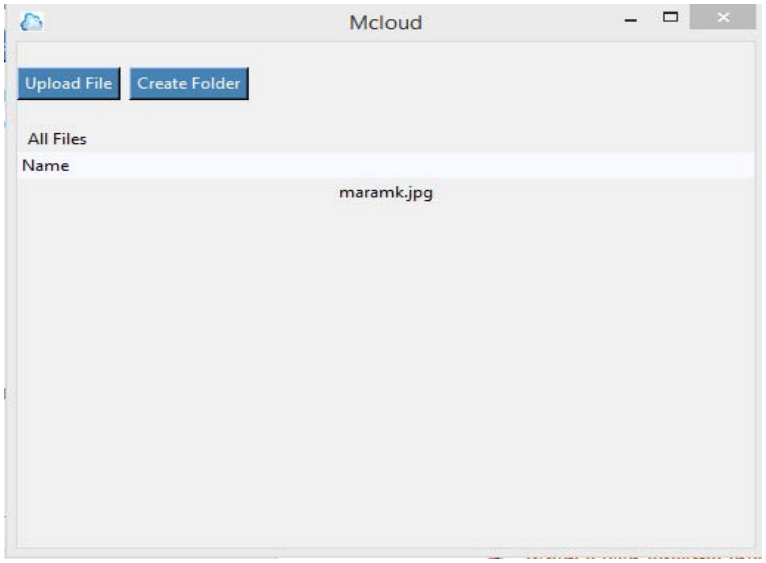

Fig. 10: Main window of the sepcific container

Azure SQL database service: Azure SQL database is a relational database-as-a service using the Microsoft SQL server engine. SQL database is a reliable, high-performance and secure database that a user can use to build applications without needing to manage infrastructure (Anonymous, 2019a-c). An Azure SQL database is created within an Azure resource group. Azure resource group is the infrastructure of an application such as a virtual machine, storage account, database and database server. An Azure SQL database is associated with an Azure SQL database server which provides a connection endpoint for database access. Azure SQL database service protects data by providing an option to set a server firewall to control access to the server (Anonymous, 2017a-k). The window for creating SQL database shown in Fig. 9

IBM Bluemix object storage service: IBM object storage Bluemix provides a scalable, pay-as-you-go and cost-effective storage service for unstructured cloud data. IBM object storage Bluemix enables a user to store data and get access to it (Sellami *et al*., 2016). This service stores files in a container. Object storage account can have one or more containers. A user can create, delete a container and view the list of folders and files that are stored in the container. IBM object storage Bluemix

service enables the user to manage each container a user can create, delete a folder and view the list of files that are stored in the folder in addition uploading, downloading and deleting file. The main window of the specific container in case the container is not empty shown in Fig. 10.

MCloud evaluation and validation: To evaluate and validate MCloud, we focus on measuring its navigation, usability integrity and performance by following two strategies the case study to evaluate the navigation, usability and integrity; it includes an experiment and a questionnaire, performance measurement.

Case study methodology: To evaluate and validate MCloud, we specify the navigation, usability and integrity and then measure them. Navigation refers "to the possible sequences of pages accessible to the user" (Han, 2006) in MCloud it may mean the ease of switching across different services on different cloud platform providers. In addition, usability is defined by ISO as "the capability of a software product to be understood, learned, used and attractive to the user when used under specified conditions" Carvajal *et al*. (2013) in MCloud it means the ease of use with respect to system integration and possible interoperability. Furthermore, integrity refers "to the validity and accuracy of data and also it ensures that the data retrieved is the same as the data stored or transmitted" (Premkumar *et al*., 2016; CTI., 2016) it means that MCloud meets the requirements of the developers and provides the correct outputs.

Therefore, the case study requires a number of developers who have different backgrounds and experiences being involved to execute some predetermined tasks on different platforms using both MCloud and the cloud services directly in order to figure out the user's experience with and without the developed interface. In addition, we check out the system integrity through asking the developers to note down the output of some tasks.

Finally, we distribute a questionnaire, to the developers in order to measure the navigation and usability, recognize their experience in using MCloud and know their opinions on whether MCloud services would be useful in practice or not.

In order to evaluate MCloud, the tasks were classified into two parts: tasks to verify the navigation and usability of MCloud and others to assess the integrity of MCloud. As for the first part, we intentionally designed the exercise across which the developer would have to switch between different cloud platforms to perform specific instructions. In the second part, the alternating instructions were combined with a request to perform tasks from predetermined databases, Queue and container and note down the outcome. In both parts, we do not restrict the developers to perform the instructions sequentially or separately for each provider.

RESULTS AND DISCUSSION

Figure 11 shows 75% of developers have experience in using cloud storage services while 25% do not have any experience. That means most of developers depended on cloud computing environment to build their applications.

About 62% of developers used storage services from various cloud providers as shown in Fig. 12. While 38% of them used one cloud provider or did not have any experience in using storage services. This emphasizes the growing use of cloud storage from various providers and the requirement for standardized methods to manage them.

About 62% of developers executed the tasks sequentially while 38% of them did each provider's tasks individually as shown in Fig. 13. Most of developers executed their tasks sequentially, therefore, it was difficult for them to manage their data because they moved

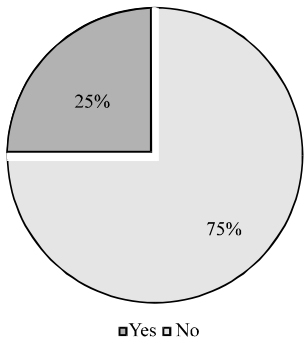

Fig. 11: Do the developers have experience in developing an application based on the cloud storage?

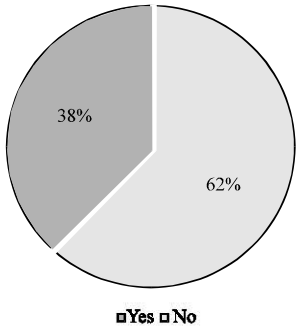

Fig. 12: Do the developers use storage services from various cloud providers

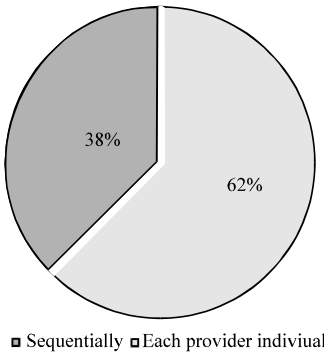

Fig. 13: How did the developers deal with the instructions? (Execute the tasks)

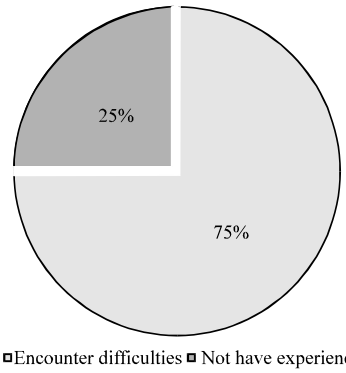

Fig. 14: Are there difficulties regarding the management of hosted of hosted data on different cloud platforms

from one provider to another. This difficulty was overcome by MCloud that facilitated the navigation between the cloud providers.

Figure 14 shows that 75% of developers encountered difficulties in managing the hosted data in various cloud providers while 25% of them had no experience in using storage services. Most of the developers faced difficulties during managing the hosted data in various cloud providers due to the vendor lock-in problem. Moreover, they experienced a lack of interoperability between these different cloud providers and a lack of uniform methods for accessing interfacing and managing the stored data in these storage systems. However, the developers found out that MCloud overcame the difficulty of managing the hosted data in various cloud providers and the lack of interoperability by providing standardized methods for accessing interfacing and managing the stored data in storage systems.

The developers found out that MCloud was easy to use, the menu services were well organized, the functions were easy to find and the main functions in SDK were available. In addition, they observed that MCloud provided useful guides and clear error messages. Thus, MCloud provides high usability. All developers agreed that MCloud supported easy navigation across the cloud providers and that MCloud helped them manage the hosted data in different cloud providers efficiently. The developers found out that MCloud facilitated the navigation and interoperability among the different cloud providers. By analyzing the answers of each developer from answer sheet, we found that they matched the correct outputs. Thus, we proved the integrity of MCloud.

Table 3: Summary of the complete time of each task on both MCloud and platform of the provider

Tasks	Mcloud (sec)	Amazon (sec)	IBM (sec)	Azure (sec)
Creating a container, making a folder in it and uploading a file to this folder in the IBM Bluemix object storage	36	-	31	-
Creating a bucket, making a folder in it and uploading a file to this folder in the Amazon S_3	46	26	-	-
Creating a DB instance in Amazon RDS	44	36	-	-
Downloading the file from Bluemix object storage and then deleting it	15	-	12	-
Downloading the file from Amazon S3 then, deleting it	22	12	-	-
Deleting the DB instance in Amazon RDS	13	12	-	-
Removing SQL DB in Azure SQL database	12	-	-	9

Evaluate the performance: To evaluate the performance of MCloud, we measured the time needed to complete a task for some specific tasks on both MCloud and the API platform of each provider. We repeated each task for ten times on MCloud and the API platform of each provider and counted the whole time for each time. Finally, we calculated the average time needed to complete the task on both MCloud and the service provider. To ensure the accuracy of such evaluation, we used the same inputs for the related tasks. For instance, we uploaded the same file to both Amazon S3 and MCloud (Table 3).

Overhead is defined as "any combination of excess or indirect computation time, memory, bandwidth or other resources that are required to perform a specific task" (CTI., 2016). In MCloud, overhead means the extra time the task needs to be executed. Through the previous experiment that measured the complete time of some tasks on both MCloud and the API platform of each provider, MCloud recorded a small increase from 3-20 sec compared to the provider's cloud platforms. We found that overhead was satisfactory and reasonable when, we considered the easy navigation between the different cloud providers supported by MCloud.

CONCLUSION

As we mentioned, there are several solutions aiming to give transparent access to multiple storage systems. However, one solution cannot achieve all user requirements. There is still a problem in managing the data hosted in different storage services of various cloud providers. In addition, there is a lack of the unified methods for accessing and managing this stored data. Therefore, we provide an approach to unify the interfaces for the storage services provided by various cloud providers. We demonstrated, through evaluation that our approach was able to provide high usability and integrity and facilitate the navigation and interoperability between the different cloud providers. Moreover, we found out that the performance of MCloud was satisfactory and reasonable. In addition, the developers revealed that MCloud helped them to manage the hosted data in different cloud providers.

RECOMMENDATIONS

In future research, we aim to add more services which provide billing information about the storage the user used and an approach for selecting the best storage type depending on the user requirements and the storage prices.

ACKNOWLEDGEMENT

King Abdul-Aziz city for Science and Technology (KACST) has funded this project, under grant No. (AP-35-20). The researchers, therefore, acknowledge and thank KACST for its technical and financial support.

REFERENCES

Abu-Libdeh, H., L. Princehouse and H. Weatherspoon, 2010. RACS: A case for cloud storage diversity. Proceedings of the 1st ACM Symposium on Cloud Computing, June 10-11, 2010, Indianapolis, IN., pp: 229-240.

Alomari, E., A. Barnawi and S. Sakr, 2014. CDPORT: A framework of data portability in cloud platforms. Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services (iiWAS'14), December 4-6, 2014, ACM, Hanoi, Viet Nam, ISBN:978-1-4503-3001-5, pp: 126-133.

Anonymous, 2017a. Amazon Relational Database Service (RDS) documentation. Amazon Web Services, Inc, Seattle, Washington, USA. https://docs.aws.amazon.com/rds/index.html#lang/en_us

Anonymous, 2017b. Amazon Simple Queue Service (SQS) documentation. Amazon Web Services, Inc, https://docs.aws.amazon.com/sqs/index.html#lang/en_us

Anonymous, 2017c. Amazon dynamo DB documentation. Amazon Web Services, Inc, Seattle, Washington, USA. https://docs.aws.amazon.com/dynamodb/index.html#lang/en_us

Anonymous, 2017d. Azure SQL database documentation-tutorials, API reference. Microsoft Azure, USA. https://docs.microsoft.com/en-us/azure/sql-database/

Anonymous, 2017e. Cloud storage Initiative|SNIA. Storage Networking Industry, SNIA Ltd, USA. https://www.snia.org/forums/csti

Anonymous, 2017f. Getting started with amazon simple storage service-amazon simple storage service. Amazon Web Services, Inc. Seattle, Washington, USA. https://docs.aws.amazon.com/AmazonS3/latest/gsg/GetStartedWithS3.html

Anonymous, 2017g. Getting started with object storage. IBM Cloud Sydney, Australia. https://console.bluemix.net/docs/services/ObjectStorage/index.html

Anonymous, 2017h. Open cloud computing interface-open Standard|Open Community. Open Cloud Computing Interface, USA., http://occi-wg.org/

Anonymous, 2017i. Single point of failure (SPOF). TechTarget Marketing Company, Newton, Massachusetts, USA. https://searchdatacenter.techtarget.com/definition/Single-point-of-failure-SPOF

Anonymous, 2017j. What is proxy server? Definition from WhatIs.com. TechTarget, Newton, Massachusetts, USA., https://whatis.techtarget.com/definition/proxy-server

Anonymous, 2017k. What is standard? Webopedia definition. QuinStreet Performance-Based Advertising Company. Foster City, California, USA. https://www.webopedia.com/TERM/S/standard.html

Anonymous, 2019a. BOTO3: The AWS SDK for python. AW Services UK Ltd. Southampton, UK. https://pypi.org/project/boto3/

Anonymous, 2019b. Microsoft Azure client libraries for python. Python Software Foundation, Wilmington, Delaware, USA. https://pypi.org/project/azure/2.0.0/

Anonymous, 2019c. Open stack, python-swiftclient: Open stack object storage API client library. Python Software Foundation Software developer, Wilmington, Delaware, USA. https://pypi.org/project/python-swiftclient/

Arunkumar, G. and N. Venkataraman, 2015. A novel approach to address interoperability concern in cloud computing. Procedia, Comput. Sci., 50: 554-559.

Beach, B., 2014. Pro Powershell for Amazon Web Services: DevOps for the AWS Cloud. APress, New York City, USA., Pages: 287.

Bocchi, E., M. Mellia and S. Sarni, 2014. Cloud storage service benchmarking: Methodologies and experimentations. Proceedings of the 2014 IEEE 3rd International Conference on Cloud Networking (CloudNet), October 8-10, 2014, IEEE, Luxembourg, Luxembourg, ISBN:978-1-4799-2730-2, pp: 395-400.

CTI., 2016. The Handbook of Human Services Management: Business, Management. Cram101 Textbook Reviews, USA., ISBN:9781467229494, Pages: 44.

Carvajal, L., A.M. Moreno, M.I. Sanchez-Segura and A. Seffah, 2013. Usability through software design. IEEE Trans. Software Eng., 39: 1582-1596.

Chung, J.Y., C. Joe-Wong, S. Ha, J.W.K. Hong and M. Chiang, 2015. CYRUS: Towards client-defined cloud storage. Proceedings of the 10th European Conference on Computer Systems (EuroSys'15), April 21-24, 2015, ACM, Bordeaux, France, ISBN:978-1-4503-3238-5, pp: 1-17.

Han, M., 2006. Navigation and request routing in web applications. Ph.D Thesis, University Bethlehem, USA. https://dl.acm.org/citation.cfm?id=1292797

Hashem, I.A.T., I. Yaqoob, N.B. Anuar, S. Mokhtar and A. Gani *et al*., 2015. The rise of "Big Data" on cloud computing: Review and open research issues. Inf. Syst., 47: 98-115.

Kolb, S. and C. Rock, 2016. Unified cloud application management. Proceedings of the 2016 IEEE World Congress on Services (SERVICES), June 27-July 2, 2016, IEEE, San Francisco, USA., ISBN:978-1-5090-2617-3, pp: 1-8.

Li, Z., Y. Dai, G. Chen and Y. Liu, 2016. Toward Network-Level Efficiency for Cloud Storage Services. In: In: Content Distribution for Mobile Internet: A Cloud-based Approach, Li, Z., Y. Dai, G. Chen and Y. Liu (Eds.). Springer, Singapore, ISBN:978-981-10-1462-8, pp: 167-196.

Liu, G., H. Shen and H. Wang, 2017. An economical and SLO-guaranteed cloud storage service across multiple cloud service providers. IEEE. Trans. Parallel, Distrib. Syst., 28: 2440-2453.

Livenson, I. and E. Laure, 2011. Towards transparent integration of heterogeneous cloud storage platforms. Proceedings of the 4th International Workshop on Data-intensive Distributed Computing (DIDC'11), June 8-8, 2011, ACM, San Jose, California, USA., ISBN:978-1-4503-0704-8, pp: 27-34.

Narasayya, V., I. Menache, M. Singh, F. Li and M. Syamala *et al*., 2015. Sharing buffer pool memory in multi-tenant relational database-as-a-service. Proc. VLDB. Endow., 8: 726-737.

Papaioannou, T.G., N. Bonvin and K. Aberer, 2012. Scalia: An adaptive scheme for efficient multi-cloud storage. Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'12), November 10-16, 2012, ACM, Salt Lake City, Utah, ISBN:978-1-4673-0804-5, pp: 1-20.

Premkumar, P., D. Shanthi and M. Jeevanandha, 2016. Enhancing data integrity assurance through detection of data violation using block based determinant approach on cloud storage. Adv. Nat. Appl. Sci., 10: 272-281.

Rafique, A., D. Van Landuyt, V. Reniers and W. Joosen, 2017. Towards an adaptive middleware for efficient multi-cloud data storage. Proceedings of the 4th Workshop on CrossCloud Infrastructures and Platforms (CrossCloud'17), April 23-26, 2017, ACM, Belgrade, Serbia, ISBN:978-1-4503-4934-5, pp: 1-6.

Ranjan, R., B. Benatallah, S. Dustdar and M.P. Papazoglou, 2015. Cloud resource orchestration programming: Overview, issues and directions. IEEE. Internet, Comput., 19: 46-56.

Rusu, O., I. Halcu, O. Grigoriu, G. Neculoiu and V. Sandulescu*et al*., 2013. Converting unstructured and semi-structured data into knowledge. Proceedings of the 11th International Conference on RoEduNet, January 17-19, 2013, IEEE, Sinaia, Romania, ISBN:978-1-4673-6114-9, pp: 1-4.

Sellami, R., S. Bhiri and B. Defude, 2016. Supporting multi data stores applications in cloud environments. IEEE. Trans. Serv. Comput., 9: 59-71.

Singh, S., Y.S. Jeong and J.H. Park, 2016. A survey on cloud computing security: Issues, threats and solutions. J. Network Comput. Appl., 75: 200-222.

Vijaya, A. and V. Neelanarayanan, 2015. Framework for platform agnostic enterprise application development supporting multiple clouds. Procedia, Comput. Sci., 50: 73-80.

Yang, K. and X. Jia, 2014. Security for Cloud Storage Systems. Springer, Berlin, Germany, ISBN:978-1-4614-7873-7, Pages: 82.

Yin, K.Z. and H.H. Wang, 2015. MCACM: A cloud storage access control model for multi-clouds environment based on XACML. Appl. Mech. Mater., 715: 2451-2454.