American Journal of Computing Research Repository, 2016, Vol. 4, No. 1, 1-6
Available online at http://pubs.sciepub.com/ajcrr/4/1/1
© Science and Education Publishing
DOI:10.12691/ajcrr-4-1-1

Migrating University's Database to the Cloud - Framework and Validation

Lydiah W. Gachanja[*], Andrew M Kahonge

Department of Computing and Informatics, University of Nairobi, Kenya
*Corresponding author: gachanjawairimu@gmail.com

Abstract All migration projects require careful planning and good methodology to ensure proper execution. Migrating enterprise data to cloud is a difficult task. Efforts to standardize cloud data migration has been going on and several frameworks have been proposed; most of them are technology-dependent and vendor-dependent frameworks. In an attempt to tackle this problem a technology-vendor-independent database migration is presented. It is achieved by adapting and revising several migration methodologies. To realize the framework, dummy MySQL database and a web-based data migration tool was used to migrate data to Google public cloud. It emerges that the framework is worth pursuing, because ensures most critical aspects of cloud migration are put into consideration. The framework will be of significant contribution to cloud data migration maturity and standardization and will help educational establishments in migrating their database to cloud because all security aspects are put into consideration. The tool used did not support the last two step of the proposed framework therefore we propose further work on designing a tool that supports the framework.

Keywords: *higher education ,framework, database as a service, cloud database, cloud computing*

Cite This Article: Lydiah W. Gachanja, and Andrew M Kahonge, "Migrating University's Database to the Cloud - Framework and Validation." *American Journal of Computing Research Repository*, vol. 4, no. 1 (2016): 1-6. doi: 10.12691/ajcrr-4-1-1.

1. Introduction

Higher Education landscape around the world is in a constant state of flux and evolution, mainly because of significant challenges arising from effort in adopting new and emerging technologies [7]. The modern age educational scenario has lead to the growth in data as the quantity of information, data collected and processed for the planning and management of educational activities has been constantly increasing. In order to provide various facilities to the students, staff, faculty members and other stakeholders, the university needs storage and computing systems that would integrate multiple services and concerned requests. Cloud Computing offer a solution to many challenges facing higher education institutes.

Cloud computing has made the computing world to shift from enterprise centric to data-centric workloads driven by the Big Data revolution, by reinventing utility/elastic computing as the new mantra for IT industry [4]. For its promise to reduce infrastructure costs and provide virtually unlimited computational power and data storage, as [2] discuss, in recent years, cloud computing has gained significant acceptance in both the enterprise application management and scientific computing.

While active research in this field provides novel concepts, techniques and principles towards building cloud-native applications, there is a significant effort, led by enterprises, to migrate traditional application databases to cloud [5]. Database as a Service (DBaaS) is a cloud service where traditional database management system (DBMS) is transformed into scalable, elastic and autonomic database platform. It can be implemented using relational database management system or non-relational database management system.

One of the biggest trends in IT nowadays is moving database workloads to the cloud. Migrating towards cloud requires a clear and well planned strategy that supports cloud computing capabilities. For example when considering migrating part of database layer to cloud, which provides data persitence and manipulation capabilities, it is necessary to address aspects of granuarity of interactions and data confidentiality, and to enable applications to interact with remote data sources. Moving database structures and their related data sets to the cloud is not an easy task. [1] Observes that migrating database to cloud cannot be done overnight and that successful migration is achieved step by step.

In this work, we present a vendor and technology independent framework for migrating part and/or whole university's database to cloud, and position it to existing application migration frameworks. The framework is applicable for use in educational establishments to migrate databases in different domains and is agnostic to the types of data sources. The requirements this framework meets, have been adopted from requirements identified through research done by collaboration between software engineers and domain experts in several research projects and collaborations with enterprise [14].

For realization of our approach, we use a dummy MySQL database and Google SQL cloud. We use the framework for partial migration of this database to a public cloud; and migration is done using an online data migration tool.

2. Related Work

There is no literature for migrating database to cloud therefore; we investigated vendor dependent frameworks and guidelines for application database layer migration. We also considered available recommendations with respect to migration to cloud.

According to [15], Amazon proposes a phase-driven approach migration of an application to their cloud infrastructure consisting of the following six phases: cloud assessment, proof of concept, data migration, application migration, leverage the cloud and optimization. The data migration phase is subdivided into a selection of the concrete Amazon AWS service and the actual migration of data. We will use this methodology by applying the first four phases and refine the data migration by using our proposed framework for migrating the database to cloud. As the methodology proposed by Amazon focuses on Amazon AWS data and storage services only, we abstract from this methodology and integrate the guidelines in our proposal.

Additionally there are other product specific guidelines and recommendation [8]. Microsoft provides Windows Azure SQL Database Migration wizard and the synchronization service Windows Azure Data Sync. We will refer these tools and tutorials during data migration phase.

Google is offering for the App Engine the tool Bulk Loader which supports both the import of CSV and XML files into the App Engine Data store, and export of CSV, XML, or text files. We abstract from vendor-specific guidelines and recommendations in order to integrate them to our process. [5] Also propose a vendor-specific methodology for the migration to Oracle products and services by providing a detailed methodology, guidelines and recommendations focusing on relational databases. We will base our proposal on their methodology, abstract from it by adapting and extending it.

Apart from vendor-specific methodologies and guidelines, there are also proposals independent from specific cloud provider. [5] Identified, taxonomically classified and systematically compared existing research on cloud computing. The researcher considered the lessons learnt from the methodology, and therefore proposed and addressed the lack of migration framework to support cloud computing maturity and tool to support automated cloud data migration. We build on migration framework to support cloud computing maturity.

[6] Specified four major rules for data migration concluding that IT staff does not often know about the semantics of the data to be migrated, which causes a lot of overheard effort. With our step by step methodology, we provide details on guidelines and recommendation on data migration. We make an assumption that the institution has already taken these factors into consideration therefore our methodology will not deal these factors. [12] Proposed a methodology of data migration that consisted of the following phases: design, extraction, cleansing, importing and verification. Additionally, they categorized data migration into storage migration, database migration, application migration, business process migration and digital data retention. We abstract from this methodology and adapt database migration in order to define cloud migration scenarios.

[14] proposed a data migration methodology that was derived from [6] consisted of the following seven phase: select migration scenarios, describe desired cloud hosting solution, select data store or data services, describe source data store or data service, identify patterns to solve migration conflict, refactoring application architecture, migrate data.

[13] Proposed using cloud data patterns in solving migration conflict which is done by mapping migration scenarios to cloud data patterns. They describe data patterns as a reusable and implementation technology-independent for a challenge related to the data layer of application in the cloud for a specific context. We will adopt some of migration scenarios identified in [13]. We will also adopt some database design criteria proposed by [10] as some of categories selecting cloud databases in our methodology.

2.1. Strauch's Migration Framework Discussed

The step by step methodology that we present this section refines and adapts the migration framework [14] so as to address the identified requirements. The methodology consists of seven phases meets; *Selection of the migration scenario* which entails detailed planning of what will be migrated and how the migration will be done by formulating a migration strategy: *describe desired cloud data hosting solution*, which involves selecting suitable cloud data using specified functional requirements and properties grouped into different categories drawn from established cloud service providers: *select cloud data store or data service*; target data store or service are identified using properties set in the previous phase; *describe source data store or data service* is the fourth phase that entails discussing the source data store so as to identify conflicts that may arise during migration: *Identify pattern to solve potential migration conflicts*; entailes identifying conflicts that may result from incompatibility of different database layers using patterns identified in [13]: *refactor application architecture;* since an application is made up of three different layers, migration of an apllication data layer to cloud has an implication on other layers (presentation and business logic): The last phase is *migration of data* which entails configuration of connections to source and target data store or service. the figure below shows the Strauch migration process framework

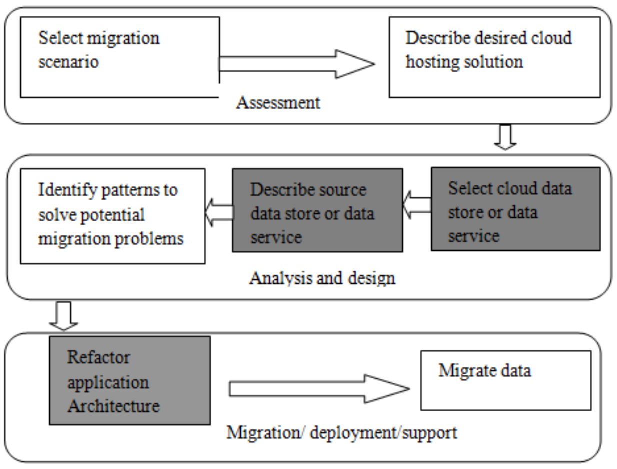

Select migration scenario
Describe desired cloud hosting solution
Assessment
Identify patterns to solve potential migration problems
Describe source data store or data service
Select cloud data store or data service
Analysis and design
Refactor application Architecture
Migrate data
Migration/ deployment/support

Figure 1. The Strauch's Framework

3. Methodology

The goal was to develop a database as a service migration framework for universities: a case of Kenyan University.

3.1. Requirements for the Proposed Migration Framework

Requirements

The functional and non functional requirements presented in this section aim to provide decision support for an application data layer migration to the cloud as proposed by [13].

Functional requirements(FR)

a) Independence from database platform: the methodology should support both relational database management systems and NoSQL data stores as discussed by [11] that have emerged recently.

b) Support of data stores and data services: the methodology should be able to support migration for both fine and course-grained types of interactions

c) On-premise and off-premise: the methodology should be able to support data stores and data services that are either hosted on-premise and off-premise, using either cloud or non-cloud technologies.

d) Management and configuration: any tool supporting this methodology should be able to provide management and configuration capabilities for data stores, data services, and migration projects. Must also support new migration projects.

e) Support various migration scenarios: Data migration depends on the context and the concrete use cases.

f) Support for incompatibility identification and resolution: the methodology has incorporated the specification of functional and non-functional requirements of database layer for the both source database layer used before migration and target data store or service.

Non-functional requirements (NFR)

a) Security – any tool using the methodology should be able to support necessary authorization, authentication, integrity and confidentiality of source of data store during export and import. And should also enforce user wide security policies when required.

b) Reusability – the methodology should be reusable with respect to integration into methodology of migration of the whole database as the one proposed by [15].

c) Extensibility – methodology should be extensible to incorporate further aspects that impacts the data migration to the cloud, such as regulations compliance.

3.2. The Difference between Strauch's Framework and the Proposed Framework

Taking into consideration [14] their methodology addresses all the requirements discussed above. However, it is a migration tool specific methodology in that, the methodology was designed to fit specific tool which makes it insecure when it comes to data migration as the user may not know whether some data is visible to other users. This is due to the fact that the user of the methodology has no access to the source code of the tool therefore hard to customize it to users' requirements.

Therefore it fails to satisfy a major requirement that we mentioned (*NFR_a*). In addition, the methodology is designed with an assumption that all errors and conflicts can be identified before and during migration, therefore testing is not required. In other words it is not clear whether after using this methodology the user will achieve optimum performance and if not no guidelines are given on how to solve the problem.

The framework deals with migrating application's data layer to cloud, therefore the need for application refactoring. On the other hand, we are interested in migrating whole database to cloud, therefore we modify this phase to accommodate database migration. Phases three and four of the methodology are not seperable because for the user to select target data store or data service they must consider the source data store or service, therefore we merge these two phases in our proposed framework.

3.3. Migration Tool

Our literature review did not result in a method that is specially for evaluating migration methodologies, so we sought for a tool that would help us realize the propose framework. To achieve this, we used a dummy MySQL database to fully migrate it to Google cloud. We choose public cloud because it was the most adopted cloud model in Kenyan universities.

We did not find any tool that could fully support our framework, therefore we settled to use a data migration assistant tool found on http://www.cloud-data-migration.com/ because it was the only tool that supported some aspects of the framework. Cloud data migration assistant is a free automated online tool that was designed to support migrating application's data layer to cloud; therefore it was not meant for migrating whole or part of enterprise database to the cloud.

4. The Proposed Framework

To address the identifeid deficienceis, we propose a vendor and database technology independent step-by-step methodology which refines and adapts the one discussed above. The Figure 2 below shows our six step framework

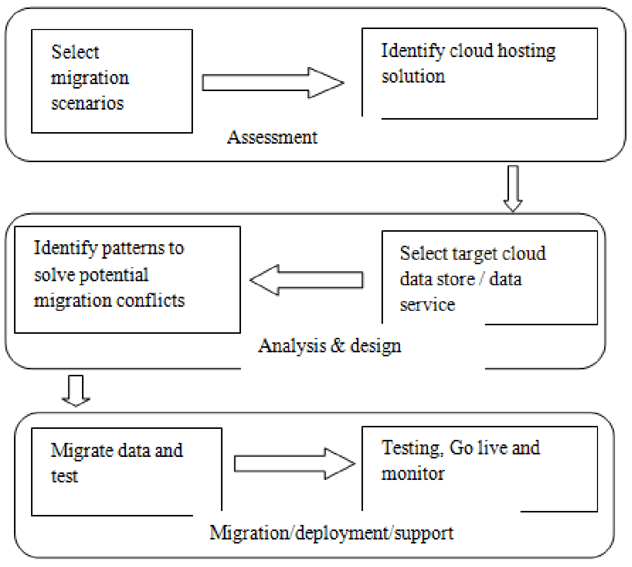

Select migration scenarios
Identify cloud hosting solution
Assessment
Identify patterns to solve potential migration conflicts
Select target cloud data store / data service
Analysis & design
Migrate data and test
Testing, Go live and monitor
Migration/deployment/support

Figure 2. The proposed framework

The steps of the methodology are explained below:

Step 1: Select Migration Scenario

The first step is selection migration scenario, where the user choose cloud migration scenarios that will fit their requirements i.e. cloud bursting, geographical replication, sharding, working on data copy *(FR_f)* [13] etc. These migration scenarios cover both migration directions between on-premise and off-premise (FR_d).; it also helps the user to formulate a migration strategy by considering migration properties such as live or non live, complete or partial migration, temporary or permanent migration. At this point any conflict between selected migration scenario and migration strategy should be solved.

Step 2: Identify cloud data hosting solution

This is focuses on the specification of functional and non-functional requirements with respect to target data store or data services. Therefore we derive an initial set of properties grouped in different categories based on [10] criteria of good database design and some categories provided by cloud providers such as Microsoft, Amazon etc. The table below shows the categories and the properties we consider. These categories cover both relational and non relational databases.

Table 1. Set of categories and properties for specification of requirements of cloud hosting solution

Category	Properties	Available options
Scalability	Degree of automation Type Degree Time to launch new instance	Manual, automated Horizontal/vertical Virtually unlimited/limited
Security	Storage encryption Transfer encryption Firewall Authentication Authorization Integrity Confidentiality	Yes/No Yes/No Yes/No Yes/No Yes/No Yes/No Yes/No
Availability	Replication Replication type Replication method Replication location Automatic failover Degree	Yes/No Master-slave, master-master Synchronous, Asynchronous Same data center, different data center Yes/No 99.9%, 99.999%
Storage	Storage type	RDBMS or NoSQL
Interoperability	Migration deployment and support Data portability Data exchange format	Yes/No None, import, export XML, JSON, proprietary

Step 3: Select target cloud data store or data service

This is done by mapping the properties of cloud data hosting solution specified in previous step to the set of available data stores and data services that have been categorized according to the same functional and non-functional properties. Implementation of this step requires data stores and data services to be previously specified according to the set of functional and non-functional properties either directly by cloud provider or by the user of this methodology. Since it is not sufficient to consider only where data store has migrated to, the functional and non-functional of the source of data store or data service are also described in order to identify and solve potential migration conflicts e.g. the database technology used *(FR_h)*.

Step 4: Identify patterns to solve potential migration conflicts

There are many challenges that come along with using cloud technology such as incompatibilities with database layer previously used or accidental disclosing of sensitive data. Incompatibilities may lead to inconsistencies between functionality of an existing traditional database layer and characteristics of an equivalent cloud data hosting solution. Hence, in step 4 conflicts are identified by checking the compatibility properties of the target data store selected in step 3 with the properties of the source data store or data service used before migration *(FR_h)* [13]

Step 5: Migrate data and Testing

This stage entails reassessing all other phases to ensure that all the requirements are met; institutional data access and security constraints are well implemented *(NFR_a)*. Then the migration is implemented and testing is done. It also entails configuration of the connections to the source and target data stores or service. Source data store or service and target data store or services are synchronized.

Step 6: Going Live and Monitoring

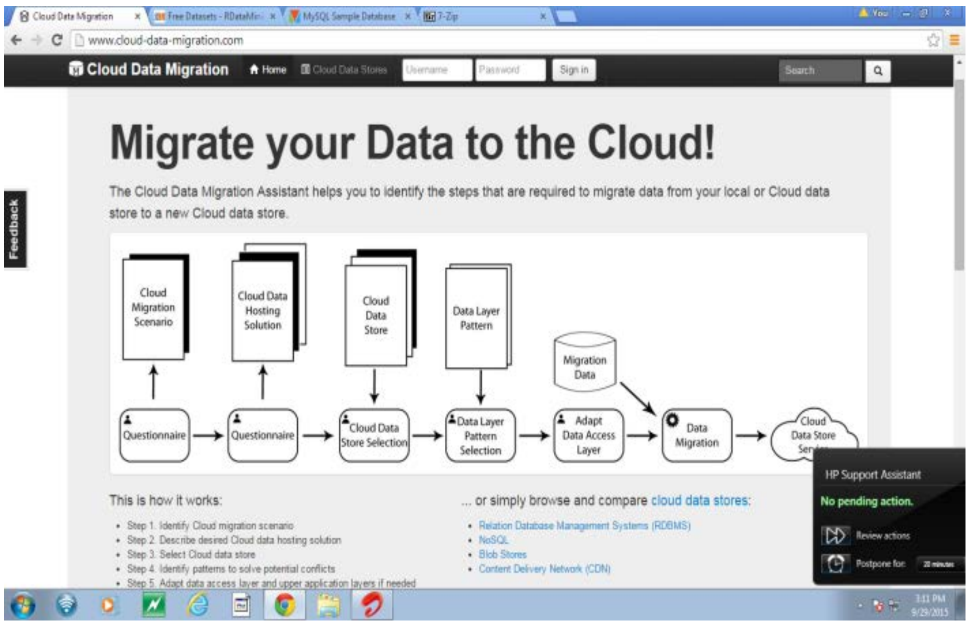
This phase involves full migration implementation and going live. Monitoring is done to ensure that the target environment meets the users' requirements. The cloud data migration tool used in this study offers basic monitoring parameters. Guidelines required in this phase are determined by the cloud provider chosen by the user. For private cloud we propose that the user should set their guidelines guided by their requirements.

5. Results

The goal was to test the validity of our framework. This was done by migrating database to a public cloud; present the results and challenges encountered during migration process.

5.1. Framework Realization

In this section we present how the proposed framework was realized and challenges encountered during this process. We used an online cloud data migration assistance tool describe in section 3.3 above. This is an automated data migration tool that offers user control of his/her data in the sense that it offers most of the facilities needed during migration depending on which migration scenario you select. The tool also offers choices of the important data security concepts and constraints. The figure below shows the tool's homepage


Cloud Data Migration

Migrate your Data to the Cloud!

The Cloud Data Migration Assistant helps you to identify the steps that are required to migrate data from your local or Cloud data store to a new Cloud data store.

This is how it works:

- Step 1. Identify Cloud migration scenario
- Step 2. Describe desired Cloud data hosting solution
- Step 3. Select Cloud data store
- Step 4. Identify patterns to solve potential conflicts
- Step 5. Adapt data access layer and upper application layers if needed

... or simply browse and compare cloud data stores:

- Relation Database Management Systems (RDBMS)
- NoSQL
- Blob Stores
- Content Delivery Network (CDN)

Figure 3.

In the first phase we choose cloud burst migration scenario where data is moved temporarily to the cloud to deal with high traffic situations. After the peak is over and local data sources are big enough to handle the traffic, the data is moved back. In step two we selected properties like security, interoperability, storage types, maintenance services i.e. monitoring and performance measuring parameters.

We selected Google cloud SQL public cloud because it provides most of properties we selected and described in the previous phase. We also selected confidential pattern to solve any data confidential conflicts between source data store and target data store. The tool automatically configures all the selected items and gives a summary of your choices; this helped us to make changes to solve any identified problem. The actual migration depended on Google Cloud SQL set procedures; therefore we configured the database to be migrated using Google set procedures, for final migration. Unfortunately, we were not able to reach this stage using our framework because we lacked a toll that could enable this to happen.

5.2. Challenges Encountered during Migration

Our framework 5th phase required we first implement the data migration and test it before going live. Using cloud data migration assistant tool it was not possible to implement this phase since the tool does not supported it. Also the final phase of the proposed migration was not implemented fully using the tool because performance and monitoring parameters are set during 2nd and 3rd phases.

We selected a public cloud model as our target data to implement our migration. We noted that not all security and privacy considerations we had selected were supported by public cloud providers, therefore we conclude that for better data security and privacy results, a private cloud or hybrid cloud should be considered rather than a public cloud.

Since the tool we used was designed to migrate database layer of an application, we observe that it is not the best for migrating the whole database to cloud. We did not get any other free migration tool that could support whole database migration to cloud, therefore propose further study on a tool that can support the proposed framework fully. We also suggest further work on the last two phases specifically on database testing and monitoring.

6. Conclusion and Further Work

We conclude that the framework is a very useful tool for database migration to cloud in educational establishments, since it helps the user to define data security and privacy according to the enterprise's policies; also the fact that the framework is vendor and technology independent. With the use of a customized migration tool, the framework offers a secure way of migrating enterprise database to cloud. For best results for an education establishment, we recommend use of private or hybrid cloud model to ensure proper data security and privacy issues are taken into consideration.

The framework will also play a great role in maturity of migrating database to cloud as there is no other framework that are so far specified for this purpose. We recommend that for this framework to be fully realized, a data migration tool that supports all its features should be developed. We further recommend the framework to be tested in real life case study and also identifying improvements on the framework especially target data store or service performance optimization.

References

[1] Andrikopoulos, V., Binz, T., Leymann, F. and Strauch, S. (2013) 'How to adapt applications for the cloud environment', in *Computing*, Vol. 95, No. 6, pp.493-535, Springer.

[2] Armbrust, M. et al. (2009) 'Above the clouds: a Berkeley view of cloud computing', Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley.

[3] Elmore, A. Curino, C, Agrawal, D, & Abbadi, A (2013) 'Towards Database Virtualization for Database as a Service', proceedings of VLDB Endowment, Vol. 6, No. 11, viewed 23rd June 2014 db.disi.unitn.ed/pages/VLDBprogram/pdf/tutorial/tut.pdf.

[4] Jain, A & Pandey, U(2013), Role of Cloud Computing in Higher Education, *International Journal of Advanced Research in Computer Science and Software Engineering 3(7)*, pp. 966-972, viewed 2nd June 2014 www.ijarcsse.com.

[5] Jamshidi, P., Ahmed, A. and Pahl, C. (2013) Cloud Migration Research – A Systematic Review. *IEEE Transactions of Cloud Computing*, Vol. 1, Issue 2, viewed 12th July 2015 http://doras.dcu.ie/19636/1/TCC-AuthorsVersion.pdf.

[6] Laszewski, T. and Nauduri, P. (2011), Migrating to the cloud-Oracle Client Server Modernization, viewed on 17th July 2015 http://www.oracle.com/technetwork/articles/cloudcomp/migrating-to-the-cloud-chap-3-495856.pdf.

[7] Masud, A, Yong J, & Huang, X (2012), 'Cloud Computing for Higher Education: A Roadmap.' Proceedings of the IEEE 16th International Conference on Computer Supported Cooperative work in design, viewed 21st June 2014 https://www.researchgate.net/publication/234801620_Cloud_Computing_for_Higher_Education_A_Roadmap.

[8] Morris, J. (2012), Practical Data Migration, 2nd Edition , BSD The Chartered Institute for IT UK.

[9] Paul, N (2009), Why use stored procedures? Viewed on 12th June 2014 http://sqlblog.com/blogs/paul_nielsen/archive/2009/05/09/wh y-use-stored-procedures.aspx.

[10] Sabalage, P.J. and Fowler, M. (2012) NoSQL Distilled: A brief Guide to the Emerging World of Polyglot Persistence, viewed 2th July 2015 http://bigbe.su/lectures/2014/15.3.pdf.

[11] Strauch, S., Andrikopoulos, V., Bachman, T. and Leyman, F. (2013a) migrating application data to the cloud using cloud data patterns, *in proceedings of CLOSER'13* SciTepress viewed on 12th July 2015 http://www.iaas.uni-stuttgart.de/RUS-data/INPROC-2013-16%20-%20Migrating%20Application%20Data%20to%20the%20Cloud%20Using%20Cloud%20Data%20Patterns.pdf.

[12] Strauch S., Andrikopoulos V., Karastoyanova D., Leymann F. (2014), Migrating Enterprise Application to Cloud: Methodology and Evaluation, *International Journal of Big Data Intelligence* viewed 14th June 2015 www.innderscience.com/ijbdi.

[13] Varia, J. (2010) Migrating your Existing Application to AWS cloud. A Phase Driven Approach to Cloud Migration, viewed 12th July 2015 http://media.amazonwebservices.com/CloudMigration-main.pdf.

[14] Victor, RK, Sigar, KO, Odongo GY (2013), Meta-Modelling Cloud Computing Architecture in Distance Learning, *International Journals of Computer science issues*, Vol. 10, Issue 3, No. 1, viewed 16th October 2014 http://ijcsi.org/papers/IJCSI-10-3-1-66-72.pdf.

[15] Waleed, A (2013), Cloud database: database as a service, *International Journal of Database Management Systems (IJDMS)* Vol.5, No.2, viewed 24th June airccse.org/journal/ijdms/papers/5213ijdms01.pdf.