Application Migration to Cloud: A Taxonomy of Critical Factors

Van Tran, Jacky Keung, Anna Liu
CSE, University of New South Wales, Australia
The Hong Kong Polytechnic University, HKSAR
National ICT Australia Ltd. Australia
Thikhanhvan.Tran@nicta.com.au,
Jacky.Keung@comp.polyu.edu.hk,
Anna.Liu@nicta.com.au

Alan Fekete
School of Information Technologies
The University of Sydney, Australia
Alan.Fekete@sydney.edu.au

ABSTRACT

Cloud computing has attracted attention as an important platform for software deployment, with perceived benefits such as elasticity to fluctuating load, and reduced operational costs compared to running in enterprise data centers. While some software is written from scratch specially for the cloud, many organizations also wish to migrate existing applications to a cloud platform. Such a migration exercise to a cloud platform is not easy: some changes need to be made to deal with differences in software environment, such as programming model and data storage APIs, as well as varying performance qualities. We report here on experiences in doing a number of sample migrations. We propose a taxonomy of the migration tasks involved, and we show the breakdown of costs among categories of task, for a case-study which migrated a .NET n-tier application to run on Windows Azure. We also indicate important factors that impact on the cost of various migration tasks. This work contributes towards our future direction of building a framework for cost-benefit tradeoff analysis that would apply to migrating applications to cloud platforms, and could help decision-makers evaluate proposals for using cloud computing.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management - cost estimation, productivity, software acquisition

General Terms

Design, Economics, Deployment, Cloud Computing

Keywords

Cloud computing, utility Computing, cost-benefit analysis, software metrics, cost factors and overheads, deployment strategy, taxonomy

SECLOUD'11, May 22, 2011, Waikiki, Honolulu, HI, USA

1. INTRODUCTION

Cloud computing has recently been the focus of much excitement in the IT community, seen by some as the next platform shift, with impact on enterprise computing that could compare to the change from main-frames to minicomputers, or from desktop PCs to networked systems. Major vendors are taking on cloud computing as a crucial strategy, governments are discussing national agendas for the coming shift, and start-ups are growing to fill niches [11].

One of the attractions for an organization using cloud resources, rather than those in an enterprise-scale data center, is that it can enjoy cost saving through larger economies of scale, since the costs of hardware, power, buildings and administrative support are typically about 5 times lower for internet-scale systems than for enterprise-scale ones. Even more significant to a rapidly-growing business is the elasticity of costs; instead of up-front purchase of an overprovisioned system, one can pay a cloud provider ongoing fees that are low at first, and smoothly increase as and when the system needs more capacity. For established businesses, there is the potential to use the cloud as additional resource (alongside existing data centers) to deal with bursts of load, perhaps seasonal, or due to intermittent activities such as stress testing. Here the cloud allows the client to delay the large commitment of funds needed to scale-up the hardware.

Many of the famous stories of cloud computing have been about startups with explosive growth, where the organization wrote or rewrote software specially to run in the cloud. However, there are cases where an orgainzation has existing application software and wants to run this on a cloud platform. Instead of a complete rewrite, one could say that we are "migrating" the software from a traditional platform such as .NET or J2EE, to a cloud-based one such as Amazon EC2 or Microsoft Windows Azure.

As with any decision about alternative ways to meet a business need, there should be a cost-benefit analysis applied when considering to migrate an application. Among the "costs" that need to be considered, in this paper, we focus on the migration effort at design and deployment time (as distinct from the operational costs when the software is running). This effort is due to discrepencies between the environment provided by a cloud platform, and that in a traditional platform. There are often differences in the version of various infrastructure, the programming models, the libraries available, even the semantics of data access are dif-

ferent; for example, cloud platforms typically provide eventual consistency rather than transactional guarantees.

As our focus on the cost of porting applications to the cloud, we report on several migration experiences, and capture our understanding in a proposed taxonomy of the tasks required for a migration. We also propose a model showing different factors that influence the costs of migration. To show the use of our taxonomy, we give measurements of the effort required for different categories of task when porting a tutorial example .NET application, so it runs in a cloud platform.

The content of this paper is arranged as follows: Section 2 describes related work, section 3 shows our approach to use a case study to derive a taxonomy of migration tasks. Section 4 describes and discusses the taxonomy we suggest, and indicates the factors that influence the migration effort. We report on the effort in the case study, mapped against the taxonomy. Section 5 reflects on our approach, and on other experiences. We conclude with Section 6.

2. RELATED WORK

Cost and benefit analysis is an important tool for IT managers to evaluate whether the benefits outweigh the costs of an IT investment. The determination of software development cost is usually the first step to achieve this goal and often a challenging task for many project managers, both overestimating and underestimating would result in unfavourable impacts to the business's competitiveness and project resource planning. The emerging trend in cloud computing requires a different perspective of understanding its costs, given that limited experience is available in the published papers. We are exploring this new situation and developing a new approach to evaluate Cloud costs, starting with migration cost as the focus of this paper, as this would enable practitioners to understand the cost related aspects when making important decisions in migrating applications to Cloud. This section provides the related work to Cloud cost estimation.

2.1 Cloud Computing

Since its emergence a few years ago, Cloud computing has been well recognized for its abilities to provide on-demand virtualized resources and services [16, 4]. Cloud computing offers a wide variety of services, including: application services (e.g. salesforce.com), storage services (e.g. Amazon S3 [12]), compute services (e.g. Google App Engine [1], Amazon EC2 [2]), and data services (e.g Amazon SimpleDB, Microsoft SQL Server Data Services, Google's Datastore).

Cloud-based systems can be classified into three main types: "Software as a Service" (SaaS), "Platform as a Service" (PaaS), and "Infrastructure as a Service" (IaaS). SaaS provides a fully functional software system for its end-users (e.g. Google Docs). PaaS provides the development platforms for developers (e.g. Google App Engine and Windows Azure). IaaS offers the flexibility for developers by providing on-demand virtual machines, users can deploy their softwares as in local servers (e.g. Amazon EC2). Different types of cloud systems and providers are selected per different specific purposes. In our study, we focus on PaaS and IaaS since they are the types that are involved in the more interesting migration scenarios.

2.2 Cloud Cost Estimation

Software costs include tangible costs (hardware and software costs), administrative costs, and development costs. Most of the time, the dominant cost is the effort cost [15]. Different cost and effort estimation techniques have been developed and applied successfully in traditional software engineering. Many of these techniques may not be able to adequately capture some of the unique and different characteristics of application development for cloud computing. Nevertheless, current methods for software sizing and effort estimation still applies, the challenge is to be able to identify additional cost overheads in the migration, involving technical redevelopment of the software targeting the Cloud. The intention of this paper is to take the unique cost overheads for Cloud application migration into consideration, which has not been evaluated in any existing related work.

Klems et al.-[7] proposed a framework to compute the value of cloud by estimating Cloud computing costs and comparing these costs to conventional IT solutions, such as hosted service or Grid computing service. Their work defines cost as the combination of a number of direct costs (e.g. facility, energy, cables, servers...) and indirect costs (e.g. cost from failing to meet business objectives). However, this framework contains incomplete lists of cost components for both direct and indirect costs, it also does not indicate how these costs can be computed, and how components in the framework link with one another to determine the estimated cost of Cloud computing.

The system proposed in [5] provides various scheduling strategies to augment the capacity of an organisation's local cluster with Cloud resources, and evaluates the trade-off between performance improvement and monetary cost spent for using the Clouds of each proposed strategy. This work only considers a portion of Cloud cost and focuses specifically on response time as benefits. This is sufficient to analyse costs and benefits amongst the proposed scheduling strategies of using Clouds, but cannot be applied to a wider scope of general application development for Clouds.

In contrary, our work is concerned with the migration effort of moving traditional on-premises application to target cloud applications. Our approach contributes towards justifying the decision of migrating applications to Cloud by exploring the cost required for the actual migration activities, which is studied through the experiment presented in this paper.

3. APPROACH

The study in this paper focuses on investigating the migration of an application to a Cloud platform. We carried out an experiment presented in a case study for the purpose of understanding the actual migration activities, followed by a discussion of our findings. We report here on experiences in doing this technical migration. We then propose a taxonomy of the migration tasks, and we show the breakdown of costs among categories of task, for a case-study which migrated a .NET n-tier application to run on Windows Azure. We also indicate unique factors that impact on the cost of migration tasks specially targeting cloud.

.Net PetShop [8] is an application designed to show best practices for building enterprise, N-tier .Net 2.0 application. It serves to highlight the key technologies and architecture to build scalable enterprise Web applications. Its Java version

called Java PetStore is also well-known for its use of illustrating how the Java EE 5 platform can be used to develop an AJAX-enabled Web 2.0 application. For these reasons, both versions of PetStore have been used in various research studies [9, 14, 17] and we believe PetShop application represents a broad class of application types, that are typically found at an enterprise organisation, and that also is a prime candidate application type for running in a cloud.

Our experiment is to migrate the PetShop application from a local server to cloud. Windows Azure and SQL Azure were selected as the PaaS Cloud platform for migration since they provide the most similar environment for PetShop .Net as in the local server. Therefore, it was expected that minimal effort would be required in the development and configuration during the migration activity. The effort required for porting PetShop .Net to Microsoft Azure and SQL Azure was recorded.

The migration of Java PetStore into Amazon EC2 and SimpleDB was also investigated to add more richness to our findings. Amazon EC2 is an IaaS Cloud, and SimpleDB is a NoSQL database with less support for full-SQL statements required in the PetStore application; therefore, different migration strategies and more re-development efforts are expected.

Function Point Analysis [3] was applied on the fully functional PetShop application to estimate its size complexity, which then can be applied to estimate its development cost. We used this estimated development cost and the recorded migration cost in our experiment on PetShop to calculate the cost overheads of migration over development.

Based on the Function Point reference cards provided by [6], PetShop is calculated to have 28 Internal Logical Files ($ILFs$), 28 External Inputs (EIs), 32 External Outputs (EOs), 36 External Inquiries (EQs), and no External Interface Files ($EIFs$), and a total of 118 Adjusted Function Points ($AFPs$). With similar settings and existing resources to the migration activities, the effort for developing PetShop application with 118 AFPs is estimated to be around 177 hours.

The efforts in hours spent on each migration task in our experiment were recorded for later analysis. It is presented in the following section, together with observations made during the study.

4. FINDINGS

Observation and experiences in our study provide a taxonomy of migration tasks in an attempt to outline a general guideline for migrating applications to Cloud. Influential cost factors are also identified in the experiment to support effort and cost estimation for each of the migration tasks.

4.1 Measured Data and Observations

When migrating PetShop to Windows Azure and SQL Azure database, some migration issues were observed and identified as:

1. We have used the existing application PetShop which was not developed by ourselves; hence, efforts were required to learn, understand, and get PetShop to work on local machine first.

2. PetShop was developed on an older platform than the current version supported by Windows Azure. This is expected to happen with many other existing applications, since Cloud computing has just emerged recently and is equipped with the latest technologies and tools, which may yield incompatibility issues. Particularly, to deploy applications to Windows Azure, Windows 7 is required, while PetShop installation file was packaged for Windows XP and could not run properly on Windows 7. We needed to deploy the PetShop source code onto Windows 7 manually.

3. The same issue is applied for PetShop database. There are existing tools offering database and data transfer from local servers to SQL Azure; however, it requires SQL Server 2008 to be installed, while PetShop was designed to work with SQL Server 2005 and cannot be installed directly on SQL Server 2008. We had to manually retrieve and run the database script on SQL Server 2008.

4. In order to deploy applications into Windows Azure Cloud, it was important to create a package file and a configuration file from the existing source code. Azure plugin for Visual Studio provides a quite straightforward method to achieve this; however, this method works with "Web application project" only, while PetShop was created as a Web-Site project, where there is no project file and it relies on ASP.NET dynamic compilation to compile pages and classes in the application. Effort was also spent on converting Web-Site project to Web Application project. Alternatively, the utility tool *cspack* provided by Azure can also be used to create the package file.

Efforts spent on addressing those issues were recorded down in terms of duration, and can be summarized as below in table 1 and 2:

Tasks	**Effort (hours)**
Install SQL Server 2005 and setup local environment in order to run the PetShop installation file	5.5
Get PetShop up and running properly	3.5
Install SQL Server 2008 to get PetShop run with later technology	2
Migrate databases from SQL Server 2005 to SQL Server 2008 and modify PetShop to work properly with SQL Server 2008	5
Install .Net 4 and modify PetShop to work on Windows 7 and .Net 4	1.5
Test Petshop	5
Total	22.5

Table 1: Recorded overhead efforts of preparing PetShop for migration

PetShop was originally designed to work with Windows XP, .Net framework 2, and SQL Server 2005. To enable PetShop run properly for the first time, these prerequisites need to be installed. Data in Table 1 shows that most time of this activity was spent on setting up the environment for PetShop to be up and running. Data in Table 2 shows that the time spent most on this activity is for overcoming the learning curve. No newer features are introduced, and Windows Azure provides similar platform to what PetShop was developed on; therefore, only minimal code modification was required.

In our experiment, learning about the application and the cloud environment, as well as installation and configuration, contribute to the cost overheads the most. Experience required to deal with unforeseen issues also counts for major additional cost. When the learning phase is finished, migrating similar type of applications will require less efforts.

Tasks	Effort (hours)
Windows Azure tutorials	6
Create Azure account and setup firewall rules	1.5
Install and explore MS Azure Training Kit	5
Tutorials: migrating databases to SQL Azure	4
Migrate PetShop database to SQL Azure	2
Modify PetShop to work with SQL Azure	4
Test PetShop on local servers against SQL Azure	2
Modify and package PetShop to Windows Azure	5.5
Deploy PetShop to Windows Azure	1.5
Test PetShop in Windows Azure with SQL Azure	5
Total	36.5

Table 2: Recorded overhead efforts of putting PetShop to Cloud platform

Figure 1 shows the overhead cost for each category of the migration tasks for PetShop which has the complexity of 118 $AFPs$. The cost overhead is calculated as the percentage of additional efforts over the estimated application development efforts (177 hours in total)

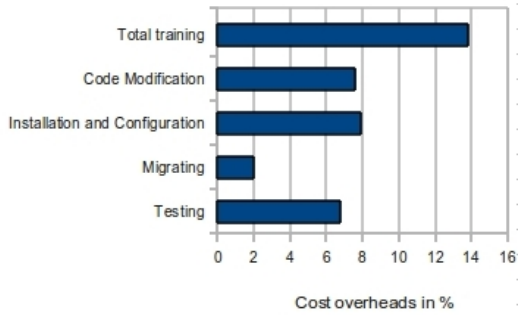

Figure 1: Migration Cost Overheads

Other additional issues were also observed as follow while considering Java PetStore with Amazon EC2 and SimpleDB:

1. Java PetStore was developed to work with JavaDB database, connected via a JDBC driver. It is not straightforward to connect Java PetStore with SimpleDB instead, since there was no JDBC driver written for SimpleDB (at the time we carried out our experiment). Writing a JDBC driver for SimpleDB from scratch with full features is not feasible.

2. Java PetStore uses JPA, which depends heavily on advanced features of JDBC drivers; therefore, SimpleDB could not be connected directly to Java PetStore.

3. There exists SimpleJPA, an open-source JPA implementation for SimpleDB. Efforts are needed on understanding this third-party library.

4. SimpleDB is a NoSQL Cloud database and does not support full-featured SQL statements, such as JOIN operations, which were required by Java PetStore. Additional efforts are needed on re-writing these operations.

5. Amazon EC2 is a type of IaaS Cloud, additional installations are required compared to the experiment on Windows Azure.

Those issues require additional efforts in addition to our experiment with Windows Azure. The additional efforts mainly fall into the categories of installation and code modification.

The measured data and observations presented above create the opportunity for further classification and future work in identifying migration issues and efforts unique to Cloud.

4.2 Taxonomy of Migration Tasks

The purpose of a *migration project* is to port an application from a local data center to a selected Cloud platform with no changes in functionalities or compromises in performance. Other aspects, such as security or networking, may require further attentions. However, in this work, we define the scope of a migration project from getting familiar with the application and the selected Cloud platform, to setting up the environment and the application ready for migration, as well as modifying and testing to ensure the application properly function in the Cloud. Any activities within this scope are regarded as *migration tasks*.

The experience and measured data from previous section, together with related work from literature review and practitioners' blogs, enable us to generalise and propose a *taxonomy of migration tasks* that any migration projects may encounter, and the migration tasks are grouped under different categories as summarized in Table 3. If T is the taxonomy of migration tasks and t is a migration task, then T is a set of t. A migration project $P \subseteq T$ consists of a list of migration tasks. Some tasks in the taxonomy can be skipped, while some tasks can be further broken down to accommodate different requirements of each project.

The following provides a summary of the taxonomy proposed in table 3.

Categories	Tasks
Training or Learning Curve	Training on the existing application: Understand system environment, specifications and configurations
	Measure system's size and Estimate system development efforts
	Training on the selected cloud platform: Understand its offerings and technologies used
	Identify any compatibility issues
	Training on third party tools: Identify and understand additional libraries, tools for data migration, and any required middlewares
Installation and Configuration	Set up development tools and environment
	Install and set up environment in IaaS Cloud
	Install third-party tools
Code modification	Database connection
	Database operation query (if using NoSQL Cloud database)
	Any required modification for compatibility issues
Migration	Prepare database for migration
	Migrate the local database to Cloud database
	Prepare system for migration
	Migrate the application
Testing	Test if local system works with database in Cloud
	Test if system in Cloud works with database in Cloud
	Write test cases and test the functionality of the application in Cloud

Table 3: Taxonomy of migration tasks

1. Training or Learning Curve - In order to ensure compliance to Cloud, a ground understanding of the application and the selected Cloud platform is required.

Efforts needed on analysing the application, understanding its components and how they are coupled together, identifying which modules are unchanged and which modules need to be modified. It is important to understand the initial system environment, specifications and configurations before planning any changes. Efforts and costs spent on this task may not be trivial for reasons such as: coding style by other developers may be difficult to study, confidentiality issues make applications are not totally transparent, and applications with many modules interacting with each other are difficult to isolate for migration purpose (if required).

When porting applications to Cloud platforms, no new features are introduced in this study. Therefore, the complexity of the application in terms of Function Points is unchanged, whereas configuration and database connection are more likely to be modified. Efforts spent on this part is directly proportional with the complexity of the application. The more complicated the application is, the more time and skills are required to understand it.

There exist quite a few major Cloud providers in the market, providing different services including PaaS and IaaS. Once Cloud services are evaluated and selected, training on these services is necessary. Some Cloud services may not fully support some features provided by similar on-premise technologies, for example, SQL Azure is the most similar to SQL Server compared to other Cloud databases, yet SQL Azure does not support distributed transactions as SQL Server does.

There have been great contributions from the Cloud community to support Cloud services integrating seemlessly with existing technologies and applications. Many open-source third-party libraries and tools have been developed. Training on these libraries and tools is also a one-time task, although it is not easy to select the appropriate libraries and tools without knowledge about them beforehand. These tools can be categorized as: additional libraries (e.g. simpleJPA for SimpleDB as discussed above), tools for data migration (e.g. Codeplex for converting and uploading databases to SQL Azure), and other utilities (e.g. Windows Azure provides *cspack* utility to pack a web site project ready for migrating to Azure).

If migrating applications to a specific Cloud platform happens for the first time, this learning curve is required; otherwise, this step can be skipped. Efforts spent on this learning task depend on the existing skills, knowledge and experiences of developers, as well as available documentation from Cloud providers. Although this training activities are one-time tasks, efforts required are not negligible.

2. Installation and Configuration - Different effort is required for these tasks, depending on different types of Cloud services selected, either PaaS or IaaS.

Development tools and environment: The application's development tools need to be installed to examine the application's components and to make any necessary code modifications.

Environment in Cloud: If the target Cloud is IaaS Cloud, efforts are required for setting up and configuring the application's environment in the Cloud server similar to its local requirements. If the target Cloud is PaaS Cloud, this step requires less effort as it is automatically handled by the Cloud providers. This activity is specific for cloud migration, as distinct from migrating an application from one platform to another where there is no such a requirement to replicate the environment.

Third-party tools: Efforts are required for installing third-party tools for training purpose and migration tasks as mentioned above.

3. Code Modification - This category depends on how different the two environments in house and in Cloud are.

Code changes: if the selected Cloud platform provides similar services and technologies to the application's environment in house, not much code modification is required. This is the case for the combination of PetShop .Net and Windows Azure in our experiment.

Database connection and query: Database connection string needs to be changed to connect to the new database server, in our experiment, the connection is modified to use SQL Azure. However, more changes are required if using SimpleDB as a non-relational database as discussed in Section 3. SimpleDB is a NoSQL database without full support of the JOIN operations, additional codings are required to provide the same functionalities and operations of the application.

Compatibility issues also require major modification and reconfiguration efforts, depending on how compatible the two environments are. Cloud technologies are generally the latest ones, while the existing applications may have been developed a few years back. During that gap, technologies may have gone through many changes and updates. There may not be a direct method to update from the old technologies to the latest ones, but more intermediate steps will be necessary. For example, PetShop .Net version 4 was developed on SQL Server 2005 while SQL Azure is only compatible with SQL Server 2008. There is no direct way to convert PetShop database from SQL Server 2005 to SQL Azure without converting to SQL Server 2008 first. Also, Cloud technologies may not provide full support for services and features offered by local servers. SQL Azure although is similar to SQL Server 2008, but it does not support distributed transactions, while SQL Server 2005 does, and PetShop .Net utilised this feature for its transactions. Code change is required to accommodate this compatibility.

4. Migration - Migration tasks do not require significant amount of effort if the preparation in previous categories are well planned and properly set up the environment for migration. These tasks include:

Prepare database for migration: SQL scripts need to be transformed appropriately to align with third party tools' requirements for database migration.

Prepare application for migration: If migrating to PaaS Cloud, the applications are required to be packed into a certain format required by Cloud platforms. If migrating to IaaS Cloud, this step can be ignored as long as all installation and environment set up have already been handled.

Migrate the application and the database: If previous tasks have been properly completed, effort required for this task is trivial and it is almost handled by the third party tools. Otherwise, plans and actions for previous tasks must be revised.

5. Testing - This step is one of the most important and necessary activities. It happens during migration to ensure each of the previous steps is completed correctly, and a full testing process needs to be carried out after migration. If test cases have been created before for local servers, they

can be reused on Clouds to ensure the application works properly. More test cases specific for Clouds may be considered. Testing needs to be done for each of the actions taken; however, big milestones for testing can be grouped as following:

If using PaaS Clouds, migrating the database to Cloud database is required first, then testing the application in local servers with the database in Cloud database. The application can then be migrated to the selected Cloud platform, which allows testing in Cloud database.

If using IaaS Clouds, developers can choose to ignore testing the application in local server against Cloud database, depending on how the environment is set up and configured.

If migrating only some components of the system to Cloud platforms, either PaaS or IaaS, intensive testing needs to be performed to ensure the entire system is integrated seemlessly and meeting certain requirements, such as security levels and performance quality. Effort required for this task is relatively significant.

These categorized migration tasks need to be carefully planned at the early stage of any migration projects. Some tasks may be broken down into more detailed levels, whereas some tasks may be skipped, depending on specific characteristics of each project. The cost factors influenced these tasks are identified and discussed in the following section.

4.3 Migration Influential Cost Factors

Based on our experiment and study, cost factors influencing the additional effort and cost on migration tasks are identified as follows. Some factors are similar to traditional software development cost factors [13, 10], some are specific for migration to Cloud.

1. Project team's capabilities: If the project team's development knowledge and skills are sufficient, training process can be picked up fast and less effort is required.

2. Application's complexity: If the application's complexity is high, it requires more effort to study and modify (if any) the application. 3. Existing knowledge and experience on Cloud providers and technologies: If the project team possesses some levels of prior knowledge and experiences of Cloud services and available tools, the learning curve can be improved significantly, and hence less effort is required. As discussed in the previous section, learning curve is a one time task, but requires significant effort.

4. Selecting the correct Cloud platforms and services (IaaS or PaaS): greatly affects the effort and cost required for the rest of migration activities; however, this practice itself is not a trivial task. If the selected Cloud platform is highly similar to the application's environment in the local server, less effort is required for modification.

5. Compatibility issues: This factor is also affected by the similarity of Cloud platforms and local servers. If the similarity is high, compatibility issues can be eliminated. Effort spent on resolving these issues varies from case to case.

6. Library dependency: When an application relies on a library to function in local server, it requires a similar library in the cloud platform. If there exists such a library for cloud, it can be reused with some minor efforts; otherwise, more efforts would be required to rewrite that library. For example, PetStore Java uses JDBC driver to connect to its JavaDB database and it also uses JPA, which depends heavily on advanced features of JDBC drivers. If we migrate PetStore's database to SimpleDB in cloud, we have to implement a full-featured JDBC driver for SimpleDB; otherwise, PetStore's data access layer must be rewritten.

7. Database features: Migrate from a Relational Database to Amazon RDS or Azure SQL requires less efforts than to a NoSQL database like SimpleDB, because NoSQL database does not support full relational features, such as Join operation. In the latter case, efforts are required to implement Join operations or rewrite custom code for the application so that it would not require Join features.

8. Connection issues: In some cloud migration cases, when only some components of the system are migrated to cloud while the rest is kept in house for various reasons (e.g. enterprises may wish to keep their sensitive data in house), the connection between two parts of the system - one in house and the other one in cloud - may face different issues such as security, latency, etc...

Some of these influential cost factors are specific for migration to cloud, because they are not applicable for a conventional migration project from one platform to another. For example, a migration project from Java to .Net is a complete rewrite and it would not have compatibility, datatabase, connection, or possibly library dependency issues. A migration project from an old version to a newer version of platform or environment would not have networking issues, library dependency or database feature issues as discussed above, and so on...

5. DISCUSSION

Using Cloud computing platforms, enterprises pay for the Total Cost of Ownership (TOC) which is calculated from the pricing models of Cloud providers. TCO covers software development cost, integration and customization cost, subscription cost (for SaaS clouds), hardware and middleware cost (for IaaS clouds), computing resource cost, storage cost, data IO transfer cost, etc... However, the benefits observed by having the system in Cloud can possibly outweigh cloud costs, especially with the advantages in quality attributes, such as scalability, 99% availability guaranteed as well as reduced maintenance cost. Besides its strengths, there are weaknesses of Cloud computing that also attract much attention and interesting work from the community, specifically in security and privacy issues.

Migrating applications to Cloud platforms requires extra efforts as demonstrated in the previous sections. Application migration from local servers to Cloud platforms is a one-time task and may seem straightforward at first. However, our experience showed that this process is not automatic and efforts spent on migration could be not trivial.

The taxonomy was derived mainly from our experience of migrating PetShop .Net to Windows Azure, a PaaS type of Cloud. We also considered the case of migrating Java PetStore to Amazon EC2, an IaaS type of Cloud, in an attempt to add more richness to the taxonomy. Due to the differences of PaaS and IaaS types of Cloud, efforts required for each migration task are also different. The following shows a side by side comparison of how different efforts are required for migrating to PaaS and IaaS clouds.

1. Training or Learning Curve - Both PaaS and IaaS clouds require significant learning efforts because: clouds offer latest technologies across various platforms; new offerings and services are rapidly created; and clouds have

a broad community who contributes with numerous third-party tools.

2. Installation and Configuration - Creating an environment in IaaS clouds with similar settings to the applications' local environment require significant efforts compared to PaaS clouds. In PaaS clouds, environments and platforms are provided and handled by providers.

3. Code Modification - IaaS clouds provide a more flexible environment to deploy and manage applications; therefore, no major code modification is required if the environment in IaaS clouds was installed and configured similarly to the local servers. There is no such flexibility in PaaS clouds; hence, custom code is required to fit into the provided environment.

4. Migration - When all those preparation steps above are completed, migration itself in both PaaS and IaaS is relatively minor.

5. Testing - Testing is unavoidable when changes are performed in the application, either code changes or configuration changes. The migration team may wish to carry out a full testing on the application in clouds, both IaaS and PaaS, to make sure it functions properly. This effort depends on the complexity of the application.

In this study, the assumption was that the Cloud target has been already selected and it is outside the scope of a migration project. However, our experiences show that efforts required for deciding Cloud providers and services are also major. Besides, for security reasons, large enterprises tend to keep sensitive data and applications in their local data centers, and migrate only some components to Cloud platforms. Therefore, enterprises may as well encounter challenging post-migrating tasks to ensure the entire system function seemlessly.

Efforts required for a migration project to a cloud platform, either PaaS or IaaS type of cloud, depend on various factors as illustrated above. This work enables us to understand these influential aspects and forms a background for us in our next step of quantifying this cloud migration effort. Our future direction based on this work is to build a metrics to estimate how much effort required for a given migration project to cloud. It will then contribute towards our plan to build a cost-benefit analysis framework on deploying applications in clouds.

6. CONCLUSION

In this paper, we experimented with the development and migration of a software application, and successfully deployed into Cloud platforms. Experience learned allows us to identify important influential cost factors for migrating to Cloud, which provides the basis for building a cost estimation model specifically tailored to Cloud computing. A taxonomy of migration tasks has been developed in the study, and applied to various projects we have been carrying out to determine how efforts are required for different types of Cloud as well as to identify any missing tasks.

The taxonomy and cost factors of migration activities identified in this paper contribute towards our future work in building an estimation model of cost overhead for migration to Cloud platforms. A cost-benefit trade-off analysis framework is to be built afterwards to provide Cloud users with a fair comparison of migrating applications into Cloud platforms with its alternatives of using other Cloud services or keeping the systems in house.

7. REFERENCES

[1] Google App Engine. http://code.google.com/.

[2] Amazon Elastic Compute Cloud. http://aws.amazon.com/ec2/, 2010.

[3] A. Albrecht and J. Gaffney. Software function, source lines of code, and development effort prediction: A software science validation. *IEEE Transactions on Software Engineering*, 9:639–648, 1983.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A berkeley view of cloud computing. Technical report, Electrical Engineering and Computer Sciences, University of California at Berkeley, 2009.

[5] M. D. de Assuncao, A. di Costanzo, and R. Buyya. Evaluating the cost-benefit of using cloud computing to extend the capacity of clusters. In *HPDC '09: Proceedings of the 18th ACM international symposium on High performance distributed computing*, pages 141–150, New York, NY, USA, 2009. ACM.

[6] IFPUG. Function point counting practices manual.

[7] M. Klems, J. Nimis, and S. Tai. Do clouds compute? a framework for estimating the value of cloud computing. *Designing E-Business Systems. Markets, Services, and Networks*, 22:110–123, 2009.

[8] G. Leake. Microsoft .net pet shop 4: Migrating an asp.net 1.1 application to 2.0, 2006.

[9] W.-S. Li, W.-P. Hsiung, O. Po, K. Hino, K. S. Candan, and D. Agrawal. Challenges and practices in deploying web acceleration solutions for distributed enterprise systems. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 297–308, New York, NY, USA, 2004. ACM.

[10] R. Madachy. Heuristic risk assessment using cost factors. *Software, IEEE*, 14(3):51 –59, may. 1997.

[11] J. C. Mudge. CLOUD COMPUTING: opportunities and challenges for australia. Technical report, The australian academy of Technological sciences and engineering, Melbourne, Victoria, Sept. 2010.

[12] M. R. Palankar, A. Iamnitchi, M. Ripeanu, and S. Garfinkel. Amazon s3 for science grids: a viable solution? pages 55–64, 2008.

[13] M. Ruhe, R. Jeffery, and I. Wieczorek. Cost estimation for web applications. In *ICSE '03: Proceedings of the 25th International Conference on Software Engineering*, pages 285–294, Washington, DC, USA, 2003. IEEE Computer Society.

[14] I. Singh, B. Stearns, and M. Johnson. *Designing enterprise applications with the J2EE platform.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[15] I. Sommerville. *Software Engineering.* Pearson Education, 8th edition, 2006.

[16] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A break in the clouds: towards a cloud definition. *SIGCOMM Computer Communication Review*, 39(1):50–55, 2009.

[17] C. Yuan, Y. Chen, and Z. Zhang. Evaluation of edge caching/offloading for dynamic content delivery. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 461–471, New York, NY, USA, 2003. ACM.