

**IMPLEMENTING E-GOVERNANCE IN BANGLADESH  
USING CLOUD COMPUTING TECHNOLOGY**

Mahafuz Aziz Aveek  
Student ID: 10241005

Md. Sakibur Rahman  
Student ID: 05101024

**Department of Computer Science and Engineering**  
August 2011



**BRAC University, Dhaka, Bangladesh**

IMPLEMENTING E-GOVERNANCE IN BANGLADESH USING CLOUD  
COMPUTING TECHNOLOGY

A Thesis

Submitted to the Department of Computer Science and Engineering

of

BRAC University

By

Mahafuz Aziz Aveek

Student ID: 10241005

Md. Sakibur Rahman

Student ID: 05101024

In Partial Fulfillment of the

Requirements for the Degree

of

Bachelor of Science in Computer Science

August 2011

## DECLARATION

We, Mahafuz Aziz Aveek and Md. Sakibur Rahman, student of Computer Science and Engineering department, BRAC University represent my thesis work on “Implementing e-Governance in Bangladesh using Cloud Computing Technology” as requirement of completion of bachelor degree. This thesis research was performed under supervision of Farazul Haque Bhuiyan, Lecturer, BRAC University, and Dhaka, Bangladesh and co-supervised by Razib Imran, Software Architect, UNDP (access to information program), and Dhaka, Bangladesh.

This is to declare that the thesis work was done by me and it has not been submitted before. Help that was taken from internet and books was mentioned at references.

Signature of Supervisor

Signature of Authors

---

Farazul Haque Bhuiyan

---

Mahafuz Aziz Aveek

---

Md. Sakibur Rahman

## ACKNOWLEDGMENTS

We are grateful to my Almighty Lord for blessing me with the patience and knowledge and the opportunity to learn something new.

We are heartily thankful to my thesis supervisor Farazul Haque Bhuiyan for his belief in me and pushing me to do better.

I, Mahafuz Aziz am thankful to my Cousin, Sk. Muttabiur Raihan for his encouragement, guidance and support from the initial to the final level to enable me to develop my understanding and complete my thesis. He inspired me and gave solutions to problems I could not solve by myself.

We are thankful to all my teachers, especially to Abdur Rahman Adnan for giving me suggestions and advices.

We would also like to thank “jhorOTEK” for helping me by providing resources that I needed. And last but not the least I thank my family and all my friends (Specially Mohibuzzaman Zico, Imran Hossain Shaon) for supporting me in times of need.

## ABSTRACT

The evolution of cloud computing technology has brought a significant impact in our life. Having different constrains such as high volume query processing request, managing a central data center, data security, uncertainty of electric power, lack of domain knowledge expertise, we personally believe e-Governance implementation with the cloud computing technology can resolve the constrains as par discussed.

Our thesis will present a novel cloud computing model for implementing e-Governance system in Bangladesh and also the implementations of e-Governance applications in a cloud platform such as Google apps engine. This platform is based on distributed system, it has non-relational database such as BigTable. It can scale up to zeta bytes records and query and it has the power of fault tolerance and high availability, which are the basic requirement for the implementations of Cloud based e- Governance system.

## TABLE OF CONTENTS

Title	i
Declaration	ii
Acknowledgement	iii
Abstract	iv
Table of Contents	v
Chapter 1: Introduction	1
1.1: Cloud Computing	1
1.2: e-Governance	1
1.3: Thesis Objective	2
Chapter 2: e-Governance in Details	3
2.1: Importance of e-governance	3
2.2: Interactive relations between organizations	4
2.3: Challenges and Cloud benefits	6
Chapter 3: Our Approach to the Problem	9
3.1: Analysis of the existing platform	9
3.1.1 OpenNebula	9
3.1.2 OpenStack	10
3.1.1 Google Apps Engine	10
3.1.1 Comparison between other platform	11
3.2: Designing the solution using the chosen platform	12
3.2.1 Problem analysis	12
3.2.2 Problem optimization	13
Chapter 4: Implementation on Google Apps Engine	14
4.1: An Overview	14
4.2: System Requirement	15
4.3: System Design	15
4.4: Model Application implementation	16
4.4.1 Configure Eclipse	16
4.4.2 Developing Our Application	16

4.4.3 Few things to note first	16
4.4.4 PMF.java	17
4.4.5 EGovernmentCommissionCenter.java	18
4.4.6 EGovernmentCloudApplicationServlet.java	19
4.4.7 Using Database api	21
4.4.8 Database	23
4.4.9 View form cloud	25
4.4.10 Application Deploy	26
4.4.11 System Monitoring Tools	27
Chapter 5: Problems with setting up a private cloud using OpenNebula	29
5.1: OpenNebula	29
5.1.1: Network configuration error	30
Chapter 6: Power of Google Apps Engine (GAE) and our Recommendation	31
6.1: Resource Sharing	31
6.2: Openness	31
6.3: Concurrency	31
6.4: Scalability	31
6.5: Fault Tolerance	32
6.6: Transparency	32
Chapter 7: Future Work	33
Chapter 8: Conclusion	34
List of References	35
Publications	35
Internet	36
Disclaimer	36

## LIST OF FIGURES

Figure 2.2.2: Interactive relations between organizations in e-governance	5-6
Figure 3.1.4: A comparison between different cloud platforms	11
Figure 3.2.1: Traditional approach to solve the problem	12
Figure 3.2.2: Optimized Cloud based solution of the problem	13
Figure 4.3: Our designed class architecture	15
Figure 4.4.6: EGovernmentCloudApplicationServlet.Java running	19
Figure 4.4.8a: Database view form localhost	23
Figure 4.4.8b: Database view form localhost	23
Figure 4.4.8c: Database view form Google cloud	24
Figure 4.4.9: Service view from the local cloud	25
Figure 4.4.10: Service application view form cloud	26
Figure 4.4.11a: My application List to control apps	27
Figure 4.4.11b: Database entry management tools	27
Figure 4.4.11c: Application developer permission	28
Figure 5.1.1: OpneNebula host error	30

## LIST OF TABLE AND CHARTS

Table 4.4.4: PMF.java class	17
Table 4.4.5: EGovernmentCommisionCenter.java class	19
Table 4.4.6: EGovernmentCloudApplicationServlet.java code	20
Table 4.4.7.a: saveRecord() function code	22
Table 4.4.7.b: query string code	23
Table 4.4.9: doget() code to show the output	25
Table 5.1.a: table of network configuration	29
Table 5.1.b: table of network configuration	29



# CHAPTER I

## INTRODUCTION

### 1.1 Cloud Computing

Cloud computing can be defined from a general point of view as an abstracted pool of highly scalable and managed compute infrastructure capable of hosting end-user applications and are billed by consumption. It's architecture somewhat abolishes the concerns about the physical location, internal composition or ownership of its component parts.

Due to rapid development of internet and web based services over the last 10 years, the cost of storage and power consumed by hardware are increasing. At the same time, large enterprises have to study data source fully to support its business. As a result the data centers are sometimes failing to meet our needs and the traditional approaches cannot provide a solution to this problem. To cope up with this challenge a new solution had to be thought off, to allow maximum efficiency and utilization of resources and the same time to be economically viable. Cloud computing can be the answer to this.

### 1.2 e-Governance

Electronic Governance, known as e-governance in short, utilizes the facilities provided by Information and Communication Technologies to perform government processes e.g. digitizing government records, automating tax collection, getting feedback from community, information dissemination, data/information gathering, elections, administration etc. It automates the major state functions and capacities – Legislative, executive, judiciary, thus allowing optimal functioning and better interaction between the governments, its institutions and people interaction of institutions. E-governance brings effectiveness and transparency in operation. It also provides a framework for dialogue between the principal actors of development of the state,

private/business sectors, NGOs, civil society, political parties and local communities.

### **1.3 Thesis Objective**

In existing system in government organizations it is very difficult to exchange information between the different sub organizations. For instance data from election commission cannot be easily shared with the Police and Defense sections. When a situation arises, it has to go through levels of bureaucracy and authorization and a lot paperwork has to be done to finally get the required information which wastes huge amount of resources and time. The objective of the thesis is to provide an e-governance model that will allow this information sharing very easily, reliably and with appropriate security attached to it without any such hassle. For instance if a traffic police wants to know personal information of a driver, he will just provide registration number of the driver and his personal information will be forwarded from the BRTA and election commission to the police officer. Here, we have implementing and solving a specific model problem to implement effective e-governance. The objective is to achieve this with cloud computing methodology.

## CHAPTER II

### E-GOVERNANCE IN DETAILS

#### 2.1 Importance of e-governance

E-Governance provides the following services:

##### *Service Management System*

A service management system provides the visibility, control and automation needed for delivery in both public and private implementations.

##### *Easy access of Information:*

In this process all kind of people will be able to access the information easily. The General people can access the information. Provides improved informational services to citizens. E.g. A2I project (access to information)

##### *Simplified user interaction with IT:*

Its user friendly self service interface accelerates time to value. The service catalog enables standards which drive consistent service delivery and provides enhanced transparency and accountability.

##### *Increase system administrator productivity:*

The productivity increase is attributed from its move from management silos to a service management system

##### *Reduce Maintenance costs of Government project:*

Since most of the government systems operate manually with some extent of localized computer applications with no or a little utilization of internet facilities. This requires a lot of money and is highly inefficient. This cost can reduced by consolidating hardware and increasing server utilization. Server utilization can go up from 5-15% up to 80% based on workloads. Measurement of performance and availability of critical virtual resources, correlations of events can be achieved by e-governance with the cloud based solution.

##### *Helps to achieve the goal of digital Bangladesh:*

If an e-governance model is used to improve the existing systems it will be a major leap towards the government's goal of digitizing Bangladesh.

### *Reduce Corruption:*

E-Governance model provides a transparent way of operation which allows monitoring of work processes in all the layers hence corruption can be minimized.

### *Social Security:*

This model can increase social security. The automated provisioning and de-provisioning speeds service delivery. The provisioning of policies allows release and reuse of assets. Its centralized identity and access control policies provides fast and affordable adherence to security compliance.

### *E- Governance Application Promotes Growth in E- Business:*

Government introduction of e-Services will stimulate expansion of e-Business, The private sector benefits from lower costs of doing business with the government, such as online licensing and tax payments, e-Procurement makes government buying more transparent, reduces business transaction costs and lowers risks for corruption, and Government transactions online at all levels will promote greater use of IT across national economies.

### *Mobile E-Based E-Government Services Advance Development:*

There is a strong case for applying mobile communications to dramatically improve access to public services, including e-Government. Mobile services are quickly emerging as the new frontier in transforming government, making it even more accessible and citizen-centric, by extending the benefits of remote delivery of government services to those unable to access public services countries where mobile (M-Government) is expected to be the key method for reaching citizens.

## **2.2 Interactive relations between organizations**

E-Governance is a process of reform in the way and delivers services to external and internal clients for the benefit of both government and the clients that they serve. Governments have innumerable applications that can be automated. Government spending on IT world increases the productivity of the government and would help in decision making and policy enforcement etc. Applications in the government fall into the following broad categories:

### *Government to Government (G2G):*

Various functions of the government interact to fulfill the work. Majority of these applications are both vertical and horizontal. Vertical applications target a specific application of the government and horizontal make it. These applications have a high degree of message passing across departments.

*Government to Enterprise (G2E):*

Enterprises like Water Board, Electricity are controlled by the governments and should react quickly to government policies. Policy enforcements, security and auditing (for accountability) are the biggest challenges.

*Government to Business (G2B):*

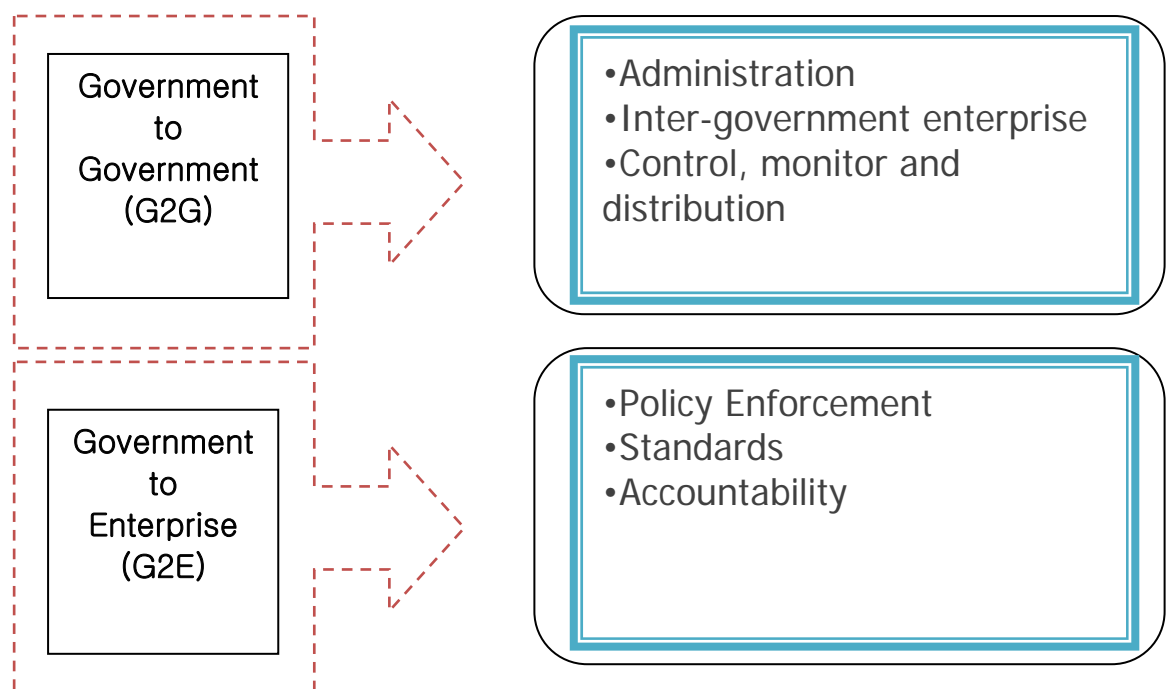
Government interacts with various business in terms of policy enforcement, collection of taxes, contract management etc. The biggest area that falls under government is Contract Management.

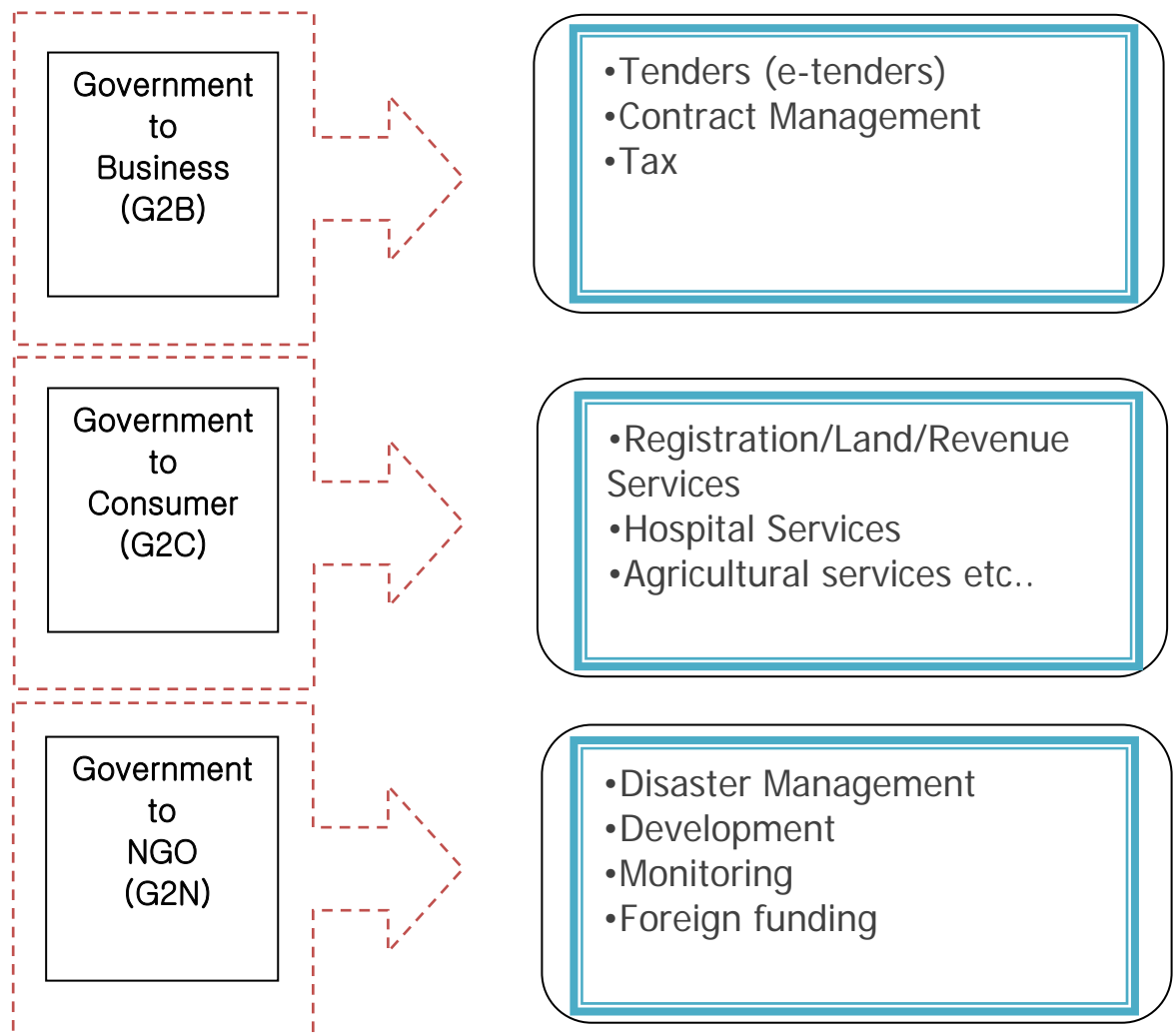
*Government to Consumer (G2C):*

Government provides numerous services to their citizens. Different departments offer various services that could scale from a simple request resolution to a starting workflow related scenarios.

*Government to NGO'S (G2N):*

Government provides numerous services to the NGOs who have been working for the rural development. The services are Monitoring, security services and so on. Beside this, The NGOs provide foreign funding, loan and disaster management policies.





**Fig 2.2.2: Interactive relations between organizations in e-governance**

### **2.3 Challenges and Cloud benefits**

#### *Data Scaling*

The databases should be scalable, to deal with large data over the years for E-Governance applications. Where relational databases ensure the integrity of data at the lowest level, cloud databases could be scaled and can be used for such type of applications. Cloud databases available for deployment offer unprecedented level of scaling without compromising on the performance. Cloud databases must be considered if the foremost concern is on-demand, high-end scalability – that is, large scale, distributed scalability, the kind that can't be achieved simply by scaling up.

#### *Auditing and logging*

Traceability to any changes to information content in E-Governance services is required. Corruption in government organizations can be controlled

by using Information Technology services, by keeping the providers of the services accountable. Process audits, security audits must be done periodically to ensure the security of the system. Cloud can help in analyzing huge volumes of data and detecting any fraud. It can help in building and placing defense mechanisms to enhance the security, thereby making the applications reliable and available.

#### *Rolling out new Instances, Replication and Migration*

Traditionally, applications in E-Governance work for department states and municipalities and hence take more time, effort, resources and budget. This happens for all the instances of these applications. Capabilities must exist to replicate these to include another municipality or e-court as part of E-Governance. Cloud architectures offer excellent features to create an instance of application for rolling out a new municipality. Cloud can reduce the time to deploy new application instances.

#### *Disaster Recovery*

Natural disasters like floods, earthquakes, wars and internal disturbances could cause the E-Governance applications not only loose data, but also make services unavailable. Multiple installations in geographically separated locations with complete backup and recovery solutions must exist. This could create huge problems. Disaster recovery procedures must be in place and practiced from time to time. Applications and data must be redundant and should be available on a short notice to switch from one data center to center. Cloud virtualization technologies allow backups and restoring. It offers application migration seamlessly compared to traditional data center. Cloud helps to increase the number of resources dynamically to maintain quality of service intact even at the times of high load, which generally happens in E-Governance.

#### *Performance and Scalability*

The architecture and technology adopted for the E-Governance initiatives should be scalable and common across delivery channels .It is required to meet growing numbers and demands of citizens. If implemented, the E-Governance portals could become the biggest users and beneficiaries of Information Technology. With cloud architectures, scalability is inbuilt. Typically, E-Governance applications can be scaled vertically by moving to a more powerful machine that can offer more memory, CPU, storage. A simpler solution is to cluster the applications and scale horizontally by adding resources.

#### *Reporting and Intelligence (Better governance)*

Data center usage (CPU, storage, network etc), peak loads, consumption levels, power usage along with time are some of the factors that needs to be monitored and reported for better utilization of resources. It

minimizes costs and plan well. Profiling data enables better visibility into various services provided by the government. Cloud offers better Business Intelligence infrastructure compared to traditional ones because of its sheer size and capabilities. Cloud computing offers seamless integration with frameworks like Map Reduce that fit well in cloud architectures. Applications can mine huge volumes of real time and historic data to make better decisions to offer better services.

### *Policy management*

E-Governance applications have to adhere and implement policies of the governments in terms of dealing with citizens. Along with the infrastructure and data center policies has to be enforced for day to day operations. Cloud architectures help a great deal in implementing policies in data center. Policies with respect to security, application deployment etc can be formalized and enforced in the data center.

### *Systems Integration and Legacy Software*

Not only the applications that are already deployed and providing services are to be moved to the cloud, but also integrate with applications deployed in the cloud. The power of Information Technology comes in co-relating the data across applications and pass messages across different systems to provide faster services to the end users. Cloud is built on SOA principles and can offer excellent solutions for integration of various applications. Also, applications can be seamlessly easily moved into cloud.

### *Obsolete Technologies and Migration to New Technologies*

Technology migration is the biggest challenge. Moving to different versions of software, applying application and security patches is the key to maintaining a secure data center for E Governance. With cloud, E-Governance applications can manage the policies well by providing security and adoptability. Various E-Governance applications can be integrated easily. Cloud architecture efficiently enables different versions and releases of the software at the same time. Once these applications are tested, they can be migrated into production with ease.

### *Going green*

More emphasis is laid out today in terms of data centers can create. The power usage, air electronic waste could create bio-hazard. This could be one of the reasons for moving to governance. Instead of duplicating these facilities, with cloud, one can offer centralized



## CHAPTER III

### OUR APPROACH TO THE PROBLEM

#### 3.1 Analysis of the existing platform

##### 3.1.1 OpenNebula

Key Features and Benefits for Integration to implement our system

Feature	Function
Infrastructure Abstraction	Seamless operation with any platform for authentication/authorization, virtualization, networking and storage, with a modular architecture to fit into any datacenter
Adaptability and Customization	Enable the deployment of any cloud architecture: private, public, hybrid and federated; customizable plug-ins to access virtualization, storage, information, authentication/authorization and remote cloud services; new plug-ins can be easily written in any language; configuration and tuning parameters to adjust behavior of the cloud management instance to the requirements of the environment and use cases; and hook mechanism to trigger administration scripts upon VM state change
Interoperability and Standards	Open standard-based architecture to avoid vendor lock-in and to enable interoperability; and implementation of standards
Openness	Open-source technology distributed under Apache license that is matured through an active and engaged community; and open internal and external interfaces
Programming Interfaces	Native cloud API in Ruby and JAVA and XMLRPC API to create new cloud interfaces and to access the core functionality

### 3.1.2 Open Stack

Key features of Open Stack to choose for our project later on

Feature	Function
Control and Flexibility	Open source platform means you're never locked to a proprietary vendor, and modular design can integrate with legacy or third-party technologies to meet your business needs. Hypervisor support for Microsoft Hyper-V, Citrix XenServer, Xen, KVM, VMWare ESX, LXC, QEMU, and UML
Industry Standard	More than 60 leading companies from over a dozen countries are participating in OpenStack, including Cisco, Citrix, Dell, Intel and Microsoft, and new OpenStack clouds are coming online across the globe.
Proven Software	Running the OpenStack cloud operating system means running the same software that today powers some of the largest public and private clouds in the world.
Compatible and Connected	Compatibility with public OpenStack clouds means enterprises are prepared for the future—making it easy to migrate data and applications to public clouds when conditions are right—based on security policies, economics, and other key business criteria.

### 3.1.3 Google apps Engine

Key features of Google apps Engine to choose finally for our project

Feature	Function
Easy to get Started	App Engine is a complete development stack, familiar technologies to build and host web applications. After writing application code, test it on local machine and upload it to Google. Once your application is uploaded we don't need to worry about system administration, bringing up new instances of your application, shearing your database or buying machines.
Free and Risk-free Development	Not only is creating an App Engine application easy, it's free! Application that people can use right away at no charge, and with no obligation. For more resources, we can enable billing and allocate your budget according to our needs.

Automatic Scalability

For the first time your applications can take advantage of the same scalable technologies that Google applications are built on, things like BigTable and GFS. Automatic scaling is built in with App Engine, all you have to do is write your application code and we'll do the rest. No matter how many users you have or how much data your application stores, App Engine can scale to meet your needs.

The reliability, performance and security

Google has a reputation for highly reliable, high performance infrastructure. With App Engine you can take advantage of the 10 years of knowledge Google has in running massively scalable, performance driven systems. The same security, privacy and data protection policies we have for Google's applications apply to all App Engine applications. We take security very seriously and have measures in place to protect your code and application data.

### 3.1.4 Comparison Between Other Platform

A simple comparison between different cloud platforms we have studied so far to choose the best optimized platform to approach our cloud based solution bellow -

	Platform ISF	VMware Vsphere	Eucalyptus	Nimbus	OpenNebula
Virtualization Management	VMware, Xen	VMware	Xen, KVM	Xen	Xen, KVM, VMware
Virtual Network Management	Yes	Yes	<b>No</b>	Yes	Yes
Image Management	Yes	Yes	Yes	Yes	Yes
Service Contextualization	<b>No</b>	<b>No</b>	<b>No</b>	Yes	Yes
Scheduling	Yes	Yes	<b>No</b>	<b>No</b>	Yes
Administration Interface	Yes	Yes	<b>No</b>	<b>No</b>	Yes
Hybrid Cloud Computing	<b>No</b>	<b>No</b>	<b>No</b>	<b>No</b>	Yes
Cloud Interfaces	<b>No</b>	vCloud	EC2	WSRF, EC2	EC2 Query
Flexibility and Extensibility	Yes	<b>No</b>	Yes	Yes	Yes
Open Source	<b>No</b>	<b>No</b>	GPL	Apache	Apache

Fig 3.1.4: A comparison between different cloud platforms

### 3.2 Designing the solution using the chosen platform

In this part, we carefully analyze the entire cloud platform to deploy our application. In this section, we discuss about the problem and its optimized solution with cloud computing technology.

Suppose A Traffic police wants to verify a licensee number including all other information along with car registration information and of a taxi driver. He sends a request to the police control center asking this information. The police control center will send a request to the BRTA. From the BRTA he can know about the licenses number and car registration number. From the Election commission database he is going to get other information about the taxi driver instantly.

#### 3.2.1 Problem analysis

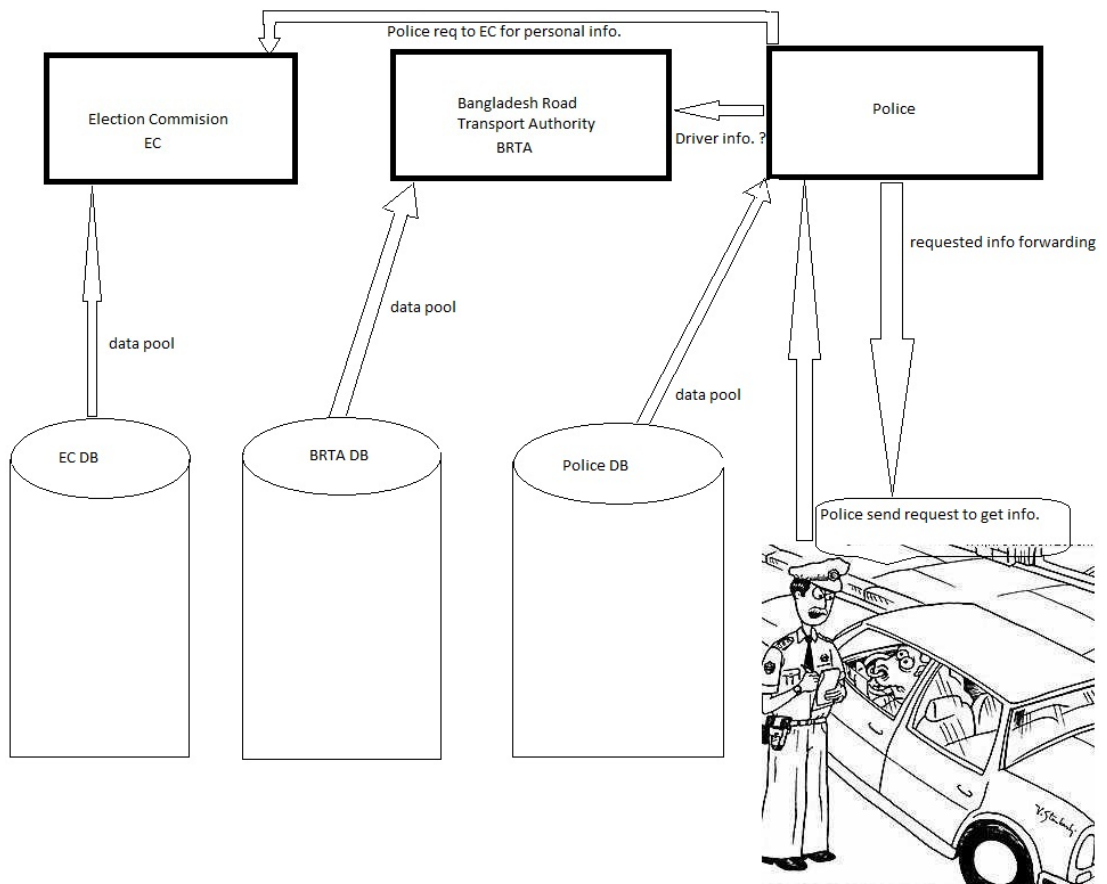


Fig: 3.2.1: Traditional approach to solve the problem

To solve this problem with traditionally web service methodology we have to design a service where we have to send three individual requests to three organizations. Every time we need to access to the Database to serve a service. Also we need individual server machine to deploy application and internet connectivity, bandwidth, maintenance and a system admin to manage the system. Which are a redundancy and a waste of our limited resource.

### 3.2.2 Problem optimization

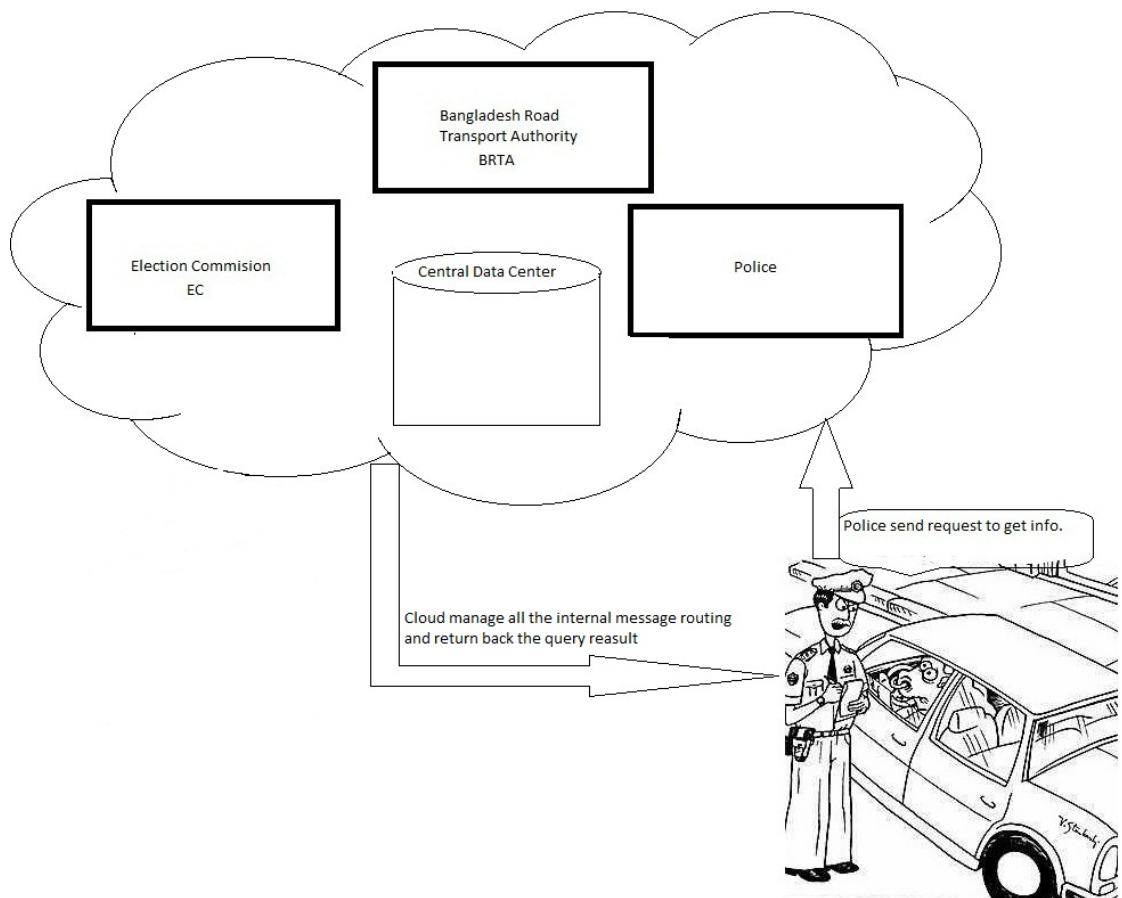


Fig. 3.2.2: Optimized Cloud based solution of the problem

Here, in this solution we don't need to be worried about all the management and other problems. Because inside of the cloud it maintains a central data center (in case of Google apps cloud is maintain a key value mapping database name BigTable). Where it store data centrally in a data center and update the data from the remote or local servers by maintaining scheduler. We don't even need a local server machine to deploy a database and service application.

## CHAPTER IV

### IMPLEMENTATION ON GOOGLE APPS ENGINE

#### 4.1 An Overview

We are using standard Java technologies and run them on Google's scalable web application infrastructure. The Java environment provides a Java 6 JVM, a Java Servlets interface, and support for standard interfaces to the App Engine scalable data store and services, such as JDO, JPA, JavaMail, and JCache.

App Engine runs Java applications using the Java 6 virtual machine (JVM). The App Engine SDK supports Java 5 and later, and the Java 6 JVM can use classes compiled with any version of the Java compiler up to Java 6.

App Engine uses the Java Servlet standard for web applications. App's servlet classes, JavaServer Pages (JSPs), static files and data files, along with the deployment descriptor (the web.xml file) and other configuration files, in a standard WAR directory structure. The JVM runs in a secured "sandbox" environment to isolate your application for service and security. App Engine provides scalable services that apps can use to store persistent data, access resources over the network, and perform other tasks like manipulating image data.

Apps can use the App Engine datastore for reliable, scalable persistent storage of data. The datastore supports two standard Java interfaces: Java Data Objects (JDO) 2.3 and Java Persistence API (JPA) 1.0. The App Engine Memcache provides fast, transient distributed storage for caching the results of datastore queries and calculations. The Java interface implements JCache (JSR 107). The service can handle CPU-intensive image processing tasks, leaving more resources available for the application server to handle web requests.

The App Engine Java SDK includes tools for testing application, uploading your application files, and downloading log data. The SDK also includes a component for Apache Ant to simplify tasks common to App Engine projects.

The development server runs your application on your local computer for development and testing. The development server can also generate configuration for datastore indexes based on the queries the app performs during testing.

## 4.2 System Requirement

- A complete Java 6 runtime environment in a secure sandbox environment.
- Based on common Java web technology standards, including servlets and WARs, JDO and JPA, java.net, JavaMail and JCache.
- A plugin for the Eclipse IDE makes project creation, testing and deployment a snap.
- Supports other languages that compile to the JVM or use JVM-based interpreters, such as JRuby, JavaScript (Rhino), and Scala.

## 4.3 System Design

This section is very important for the application developer; here strong OOP concept is a must to implement such a huge project like e-governance.

For our model e-governance project we are going to create a very simple class design to show service development in Google cloud.

We design our system with a very simple design to test how to run a simple application in cloud and manage the service. The system architecture is noted on bellow:



Fig4.3: Our designed class architecture

## 4.4 Model Application implementation

### 4.4.1 Configure Eclipse

Download Eclipse editor for windows 32 bit form [here](#).

Help menu > Install New Software... > Work with:

<http://dl.google.com/eclipse/plugin/3.6>

That how we basicly add the plugin libraries for GAE (Google apps Engine)

After Installation select all the Plugin and SDK's check box.

A simple easy configuration on development bed is rady now!

### 4.4.2 Developing Our Application

The first thing to do is to create a New Google Web Application Project. Follow these steps:

1. Either click on File → New → Other or press Ctrl-N to create a new project. Select Google and then Web Application project. Alternately you could also click on the New Web Application Project Toolbar icon as part of the Google Eclipse plugin.
2. In the New Web Application Project dialog, deselect the Use Google Web Toolkit and give a name to your project. I have named mine "EGovernmentCloudApplication".
3. Click on Finish.

This will generate the project and also create a sample Hello World Servlet for you. But we will be writing our own Servlet.

### 4.4.3 Few things to note first

Quite a few things are enabled for you by default as far as the database support is concerned. They are as follows:

1. Several JAR files are added to the CLASSPATH by default. Take a look and you will see several JARs \*jpa\*.jar, \*datanucleus\*.jar, etc.
2. In the src/META-INF folder, you will find a jdoconfig.xml file. There is a default Persistence Manager Factory class.



3. GAEJ uses the DataNucleus library to abstract the BigTable store. The DataNucleus library provides the JDO and JPA interfaces so that we do not have to deal with the underlying low level API. We will also find a logging.properties file present in war/WEB-INF folder. We will find several log levels mentioned for the DataNucleus classes. We can tweak them to lower levels like DEBUG/INFO to see more debug level statements of what happens when we are using these APIs. I have found it very helpful to set the debug levels to DEBUG/INFO especially when facing a problem.

#### 4.4.4 PMF.java

The first class that we shall write is a simple utility class that shall get us the underlying Persistence Manager factory instance. This class is important since all other methods like saving a record, querying records, etc will work on the instance of the PersistenceManagerFactory.

The code is shown below and wherever we need an instance of the class, we shall simply invoke the **get()** method below:

```
import javax.jdo.JDOHelper;
import javax.jdo.PersistenceManagerFactory;

public final class PMF {

    private static final PersistenceManagerFactory
    pmfInstance = JDOHelper
        .getPersistenceManagerFactory("transactions-
optional");

    private PMF() {
    }

    public static PersistenceManagerFactory get() {
        return pmfInstance;
    }
}
```

Table4.4.4: PMF.java class

#### 4.4.5 EGovernmentCommissionCenter.java

1. We need to have a constructor that contains all the fields except for the Key field.
2. All fields that need to be persisted are annotated with the @Persistent annotation.
3. The class is declared as being persistable via the @PersistenceCapable annotation and we are leaving the identity to the Application.
4. The Primary Key field i.e. Key is declared via the @PrimaryKey annotation and we are using an available Generator for the ID instead of rolling our own.

```
import javax.jdo.annotations.*;

@PersistenceCapable(identityType = IdentityType.APPLICATION)
public class EGovernmentCommissionCenter {

    //Instance variable
    @PrimaryKey
    @Persistent(valueStrategy = IdGeneratorStrategy.IDENTITY)
    public Long id ;
    @Persistent
    public String _name = null;
    @Persistent
    public String _nameOfMother = null;
    @Persistent
    public String _nameOfFather = null;
    @Persistent
    public String _maritalStatus = null;
    @Persistent
    public String _dateOfbirth = null;
    @Persistent
    public String _address = null;
    @Persistent
    public String _perAddress = null;
    @Persistent
    public String _genderType = null;
    @Persistent
    public String _nameOfSpous = null;
    @Persistent
    public String _passportNum = null;
    @Persistent
    public String _licenseNum = null;
    @Persistent
    public String _nID = null;

    public EGovernmentCommissionCenter(String name, String
nameOfMother, String nameOfFather, String maritalStatus, String
dateOfbirth ,
                                     String address, String perAddress , String
genderType , String nameOfSpous, String passportNum,
                                     String licenseNum , String nID) {
```

```

        this.setName(name);
        this.setNameOfMother(nameOfMother);
        this.setNameOfFather(nameOfFather);
        this.setMaritalStatus(maritalStatus);
        this.setDateOfBirth(dateOfBirth);
        this.setAddress(address);
        this.setPerAddress(perAddress);
        this.setGenderType(genderType);
        this.setNameOfSpous(nameOfSpous);
        this.setPassportNum(passportNum);
        this.setLicenseNum(licenseNum);
        this.setNID(nID);
    }
}

```

Table4.4.5: EGovernmentCommisionCenter.java class

#### 4.4.6 EGovernmentCloudApplicationServlet.java

We shall now look at how to persist the above EGovernmentCommissionCenter. Since we are not going to build a UI for it, we shall simply invoke a servlet (HTTP GET) with the required parameters. It would almost be like a FORM submitting these values to the Servlet. Before we write this Servlet code, let us look at how we will invoke it. Given below is a screenshot of the browser where I punch in the URL: <http://localhost:8888/egovernmentcloudapplication>

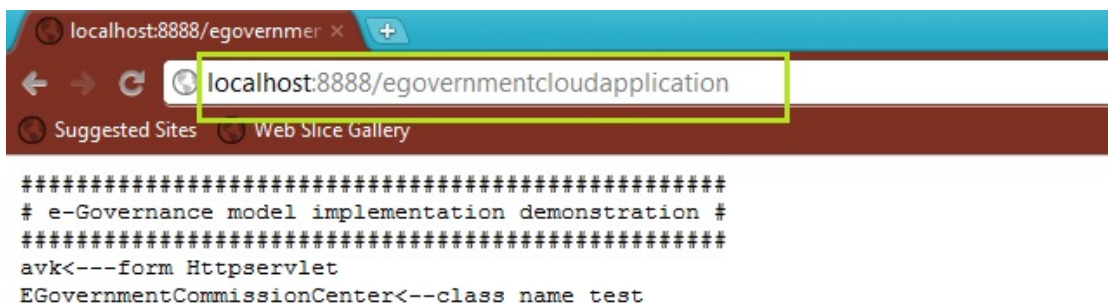


Fig4.4.6: EGovernmentCloudApplicationServlet. Java running

As you can see, I am running the application on my local development server and invoke the servlet (which we shall see in a while). These two parameters are two key fields of the HealthReport class that we saw above. The other fields like ReportDateTime and Status are determined automatically by the application. Similarly the Key value of the record in the underlying datastore will be generated by App Engine infrastructure itself.

Let us now look at the PostHealthIncidentServlet.java code shown below:

```
package com.brac.egovernment;  
  
import java.io.IOException;  
import java.util.Iterator;  
import java.util.List;  
  
import javax.jdo.PersistenceManager;  
import javax.jdo.Query;  
  
import javax.servlet.http.*;  
  
@SuppressWarnings("serial")  
public class EGovernmentCloudApplicationServlet extends HttpServlet {  
  
    public void doGet(HttpServletRequest req, HttpServletResponse resp)  
        throws IOException {  
        resp.setContentType("text/plain");  
  
        resp.getWriter().println("#####");  
        resp.getWriter().println("# e-Governance model implementation  
demonstration #");  
  
        resp.getWriter().println("#####");  
resp.getWriter().println(person1.getName() + "<---form Httpservlet");  
  
        resp.getWriter().println(EGovernmentCommissionCenter.class.getSimpleName()+"<--  
class name test" );  
    }  
    public void doPost(HttpServletRequest req, HttpServletResponse resp)  
        throws IOException {  
        resp.setContentType("text/plain");  
  
        //post handler  
        resp.sendRedirect("index.html");  
    }  
}
```

Table4.4.6: EGovernmentCloudApplicationServlet.java code

## 4.4.7 Using Database api

Key Points are :

1. The **saveRecord()** method first gets the instance of the PersistenceManager through the **PMF.java** class that we wrote earlier.
2. It simply invoke the **makePersistent()** method on it. The makePersistent() method will take as a parameter the object that you want to persist. In our case it is the HealthReport.java class instance that we have created in the servlet. This method will persist the record and in the process also assign it a unique key.
3. Finally, we need to close the PersistenceManager instance by invoking the **close()** method.

The entire code listing is shown below:

```
public String saveRecord(){

        EGovernmentCommissionCenter person1 = new
EGovernmentCommissionCenter("avk", "ma", "baba", "not yet", "04th
november", "Dhanmondi", "satkhira", "male", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");
        EGovernmentCommissionCenter person2 = new
EGovernmentCommissionCenter("amk", "ma", "baba", "not yet", "30th
December", "Dhanmondi", "satkhira", "male", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");
        EGovernmentCommissionCenter person3 = new
EGovernmentCommissionCenter("ank", "ma", "baba", "not yet", "29th
november", "satkhira", "satkhira", "male", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");
        EGovernmentCommissionCenter person4 = new
EGovernmentCommissionCenter("sagor", "ama", "ababa", "not yet",
"4th jan", "canada", "satkhira", "male", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");
        EGovernmentCommissionCenter person5 = new
EGovernmentCommissionCenter("faraz", "ama", "ababa", "not yet",
"4th feb", "belyroad", "b.barria", "male", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");
        EGovernmentCommissionCenter person6 = new
EGovernmentCommissionCenter("sakib", "ma", "baba", "not yet", "4th
march", "mohammadpur", "b.barria", "male", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");
        EGovernmentCommissionCenter person7 = new
EGovernmentCommissionCenter("shaon", "ama", "ababa", "not yet",
"4th april", "BRA", "khulna", "female", null, "B123456DHK",
"D12345CHT", "N12345DHNMONDI");

        PersistenceManager pm = null;
        pm = PMF.get().getPersistenceManager();

        pm.makePersistent(person1); // store
        pm.makePersistent(person2);
        pm.makePersistent(person3);
```

```

        pm.makePersistent(person4);
        pm.makePersistent(person5);
        pm.makePersistent(person6);
        pm.makePersistent(person7);

        Query query = null;
        query = pm.newQuery(EGovernmentCommissionCenter.class,
        "_address != paramAddress && _genderType == paramGender");
        query.declareParameters("String paramAddress, String
        paramGender");

        String return1 = null, return2 = null;

        @SuppressWarnings("unchecked")
        List<EGovernmentCommissionCenter> result =
        (List<EGovernmentCommissionCenter>)
        query.executeWithArray("Dhanmondi", "male");

        Iterator<EGovernmentCommissionCenter> it =
        result.iterator();
        while(it.hasNext()){
            EGovernmentCommissionCenter a = it.next();

            return1 = a.id + " #Name: " + a.getName()+
        #Father_Name: " + a.getnameOfFather()+ " #Address:
        "+a.getaddress()+ " #lisence no: "+a.getlicenseNum() + "\n";
        }
        // Second genetation query

        query = pm.newQuery(EGovernmentCommissionCenter.class,
        "_address == param");
        query.declareParameters("String param");

        @SuppressWarnings("unchecked")
        List<EGovernmentCommissionCenter> result1 =
        (List<EGovernmentCommissionCenter>)
        query.executeWithArray("Dhanmondi");

        Iterator<EGovernmentCommissionCenter> it1 =
        result1.iterator();
        while(it1.hasNext()){
            EGovernmentCommissionCenter a = it1.next();

            return2 = a.id + " **name: " + a.getName()+
        **passport no. - " + a.getpassportNum() + "\n";
        }

        pm.close(); //database close

        return return1+return2;
    }

```

Table4.4.7.a: saveRecord() function code

The query we are using ther is:

```
SELECT * FROM EGovernmentCloudApplicationServlet WHERE _address != paramAddress && _genderType == paramGender
```

```
SELECT * FROM EGovernmentCloudApplicationServlet WHERE "_address == param
```

Table4.4.7.b: query string code

### 4.4.8 Database

The code above generate Database bellow though we are not creating the database manually. The database image shown bellow:

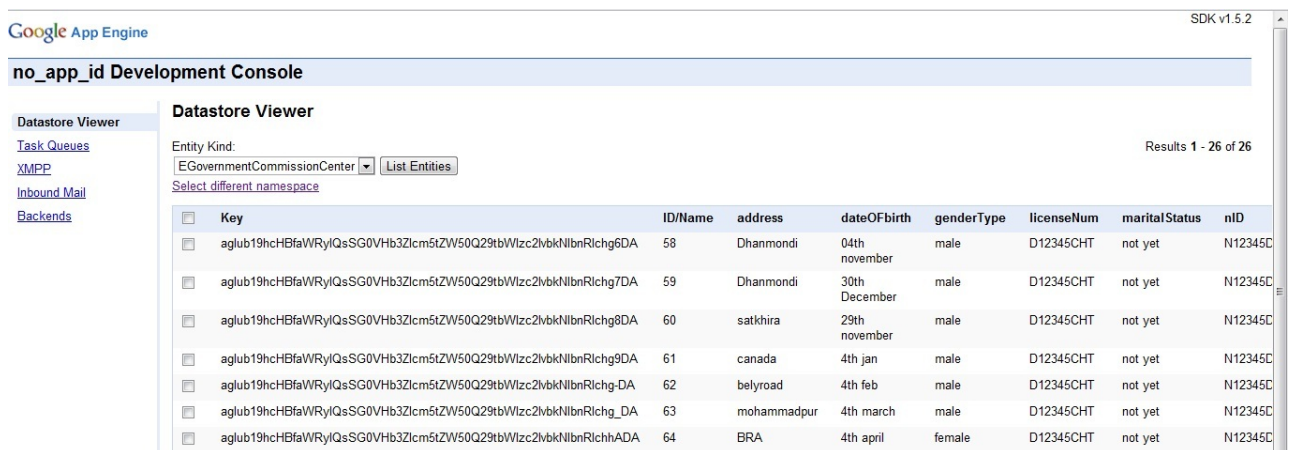


Fig4.4.8a: Database view form localhost

ID/Name	address	dateOfBirth	genderType	licenseNum	maritalStatus	nID	name	nameOfFather	nameOfMother	nameOfSpous	passportNum	perAddress
58	Dhanmondi	04th november	male	D12345CHT	not yet	N12345DHNMONDI	avk	baba	ma		B123456DHK	satkhira
59	Dhanmondi	30th December	male	D12345CHT	not yet	N12345DHNMONDI	amk	baba	ma		B123456DHK	satkhira
60	satkhira	29th november	male	D12345CHT	not yet	N12345DHNMONDI	ank	baba	ma		B123456DHK	satkhira
61	canada	4th jan	male	D12345CHT	not yet	N12345DHNMONDI	sagor	ababa	ama		B123456DHK	satkhira
62	belyroad	4th feb	male	D12345CHT	not yet	N12345DHNMONDI	faraz	ababa	ama		B123456DHK	b.baria
63	mohammadpur	4th march	male	D12345CHT	not yet	N12345DHNMONDI	sakib	baba	ma		B123456DHK	b.baria
64	BRA	4th april	female	D12345CHT	not yet	N12345DHNMONDI	shaon	ababa	ama		B123456DHK	khulna
65	Dhanmondi	04th november	male	D12345CHT	not yet	N12345DHNMONDI	avk	baba	ma		B123456DHK	satkhira
66	Dhanmondi	30th December	male	D12345CHT	not yet	N12345DHNMONDI	amk	baba	ma		B123456DHK	satkhira
67	satkhira	29th november	male	D12345CHT	not yet	N12345DHNMONDI	ank	baba	ma		B123456DHK	satkhira

Fig4.4.8b: Database view form localhost

Firefox - Data Viewer - egovTest

google.com https://appengine.google.com/datastore/explorer?&app\_id=s-egovmodel&version\_id=1.352504026544139763

Google app engine aveekbu@gmail.com | My Account | Help | Sign out

Application: egovmodel [High Replication] 1 My Applications

Query Create

By kind: EGovernmentCommissionCenter kinds as of 0:00:00 ago

Options

**EGovernmentCommissionCenter Entities**

Prev 20 1.7 Next 20

ID/Name	address	dateOfbirth	genderType	licenseNum	maritalStatus	nID	name	nameOfFather	nameOfMother	nameOfSpous	passportNum	perAddress
id=1	Dhanmondi	04th november	male	D12345CHT	not yet	N12345DHNMONDI	awk	baba	ma	<null>	B123456DHK	satkhira
id=2	BRA	4th april	female	D12345CHT	not yet	N12345DHNMONDI	shaon	ababa	ama	<null>	B123456DHK	khulna
id=1001	Dhanmondi	30th December	male	D12345CHT	not yet	N12345DHNMONDI	amk	baba	ma	<null>	B123456DHK	satkhira
id=2001	satkhira	29th november	male	D12345CHT	not yet	N12345DHNMONDI	ank	baba	ma	<null>	B123456DHK	satkhira
id=3001	canada	4th jan	male	D12345CHT	not yet	N12345DHNMONDI	sagor	ababa	ama	<null>	B123456DHK	satkhira
id=4001	belyroad	4th feb	male	D12345CHT	not yet	N12345DHNMONDI	faraz	ababa	ama	<null>	B123456DHK	b baria
id=5001	mohammadpur	4th march	male	D12345CHT	not yet	N12345DHNMONDI	sakib	baba	ma	<null>	B123456DHK	b baria

Delete Prev 20 1.7 Next 20

- Main
- [Dashboard](#)
  - [Instances](#)
  - [Logs](#)
  - [Versions](#)
  - [Backends](#)
  - [Cron Jobs](#)
  - [Task Queues](#)
  - [Quota Details](#)
- Data
- [Datastore Indexes](#)
  - [Datastore Viewer](#)
  - [Datastore Statistics](#)
  - [Blob Viewer](#)
  - [Prospective Search](#)
- Administration
- [Application Settings](#)

Fig4.4.8.c: Database view form Google cloud



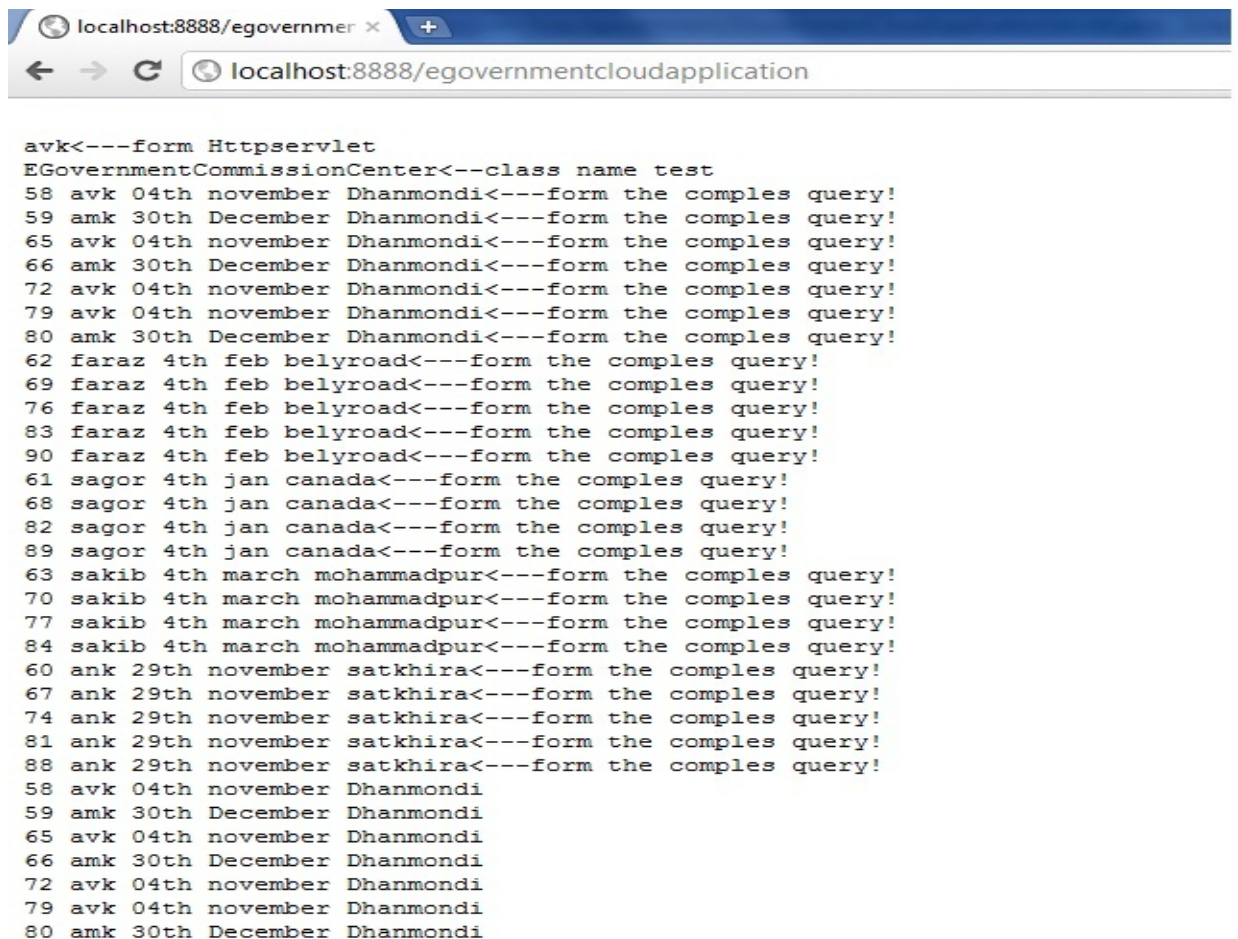
#### 4.4.9 View form cloud

The “EGovernmentCloudApplicationServlet” class extended from “HttpServlet” helps the view from the cloud using the function below:

```
public void doGet(HttpServletRequest req, HttpServletResponse
resp)
    throws IOException {
    resp.setContentType("text/plain");
    resp.getWriter().println();
}
```

Table4.4.9: doGet() code to show the output

The final images bellow:



```
avk<---form Httpservlet
EGovernmentCommissionCenter<---class name test
58 avk 04th november Dhanmondi<---form the comples query!
59 amk 30th December Dhanmondi<---form the comples query!
65 avk 04th november Dhanmondi<---form the comples query!
66 amk 30th December Dhanmondi<---form the comples query!
72 avk 04th november Dhanmondi<---form the comples query!
79 avk 04th november Dhanmondi<---form the comples query!
80 amk 30th December Dhanmondi<---form the comples query!
62 faraz 4th feb belyroad<---form the comples query!
69 faraz 4th feb belyroad<---form the comples query!
76 faraz 4th feb belyroad<---form the comples query!
83 faraz 4th feb belyroad<---form the comples query!
90 faraz 4th feb belyroad<---form the comples query!
61 sagor 4th jan canada<---form the comples query!
68 sagor 4th jan canada<---form the comples query!
82 sagor 4th jan canada<---form the comples query!
89 sagor 4th jan canada<---form the comples query!
63 sakib 4th march mohammadpur<---form the comples query!
70 sakib 4th march mohammadpur<---form the comples query!
77 sakib 4th march mohammadpur<---form the comples query!
84 sakib 4th march mohammadpur<---form the comples query!
60 ank 29th november satkhira<---form the comples query!
67 ank 29th november satkhira<---form the comples query!
74 ank 29th november satkhira<---form the comples query!
81 ank 29th november satkhira<---form the comples query!
88 ank 29th november satkhira<---form the comples query!
58 avk 04th november Dhanmondi
59 amk 30th December Dhanmondi
65 avk 04th november Dhanmondi
66 amk 30th December Dhanmondi
72 avk 04th november Dhanmondi
79 avk 04th november Dhanmondi
80 amk 30th December Dhanmondi
```

Fig4.4.9: service view from the local cloud

#### 4.4.10 Application Deploy

It is very simple to deploy the application (In my case model e-governance application) to Google cloud.

1. Upload app to appspot.com:

2. Click Google App Engine icon   
Enter Email, Password, App Engine project settings... > Application ID:  
<name of app>

3. Click Deploy

4. Open URL in browser  
`http://<name of app>.appspot.com`

In my case our application shows bellow:

<http://egovmodel.appspot.com/egovernmentcloudapplication>

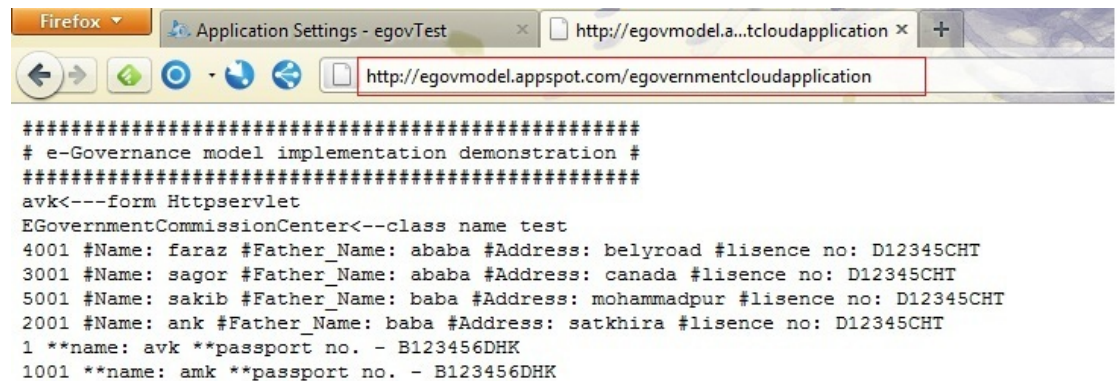


Fig4.4.10: Service application view form cloud

## 4.4.11 System Monitoring Tools

We use some GAE monitoring tools to monitor our application and manage the application through cloud. The best part is we have the full control over GAE application. I can modify and change my VM as I want to.

Some of the tools are shown below:

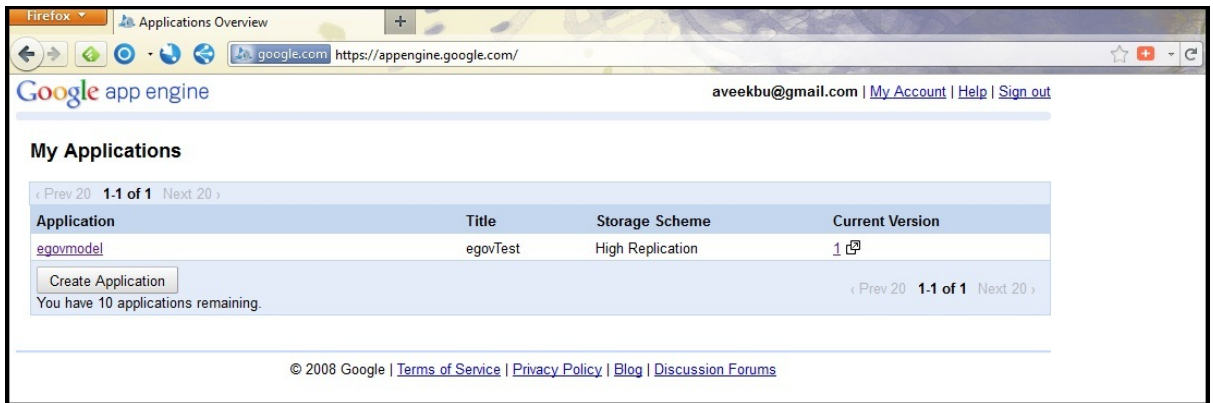


Fig4.4.11.a: My application List to control apps

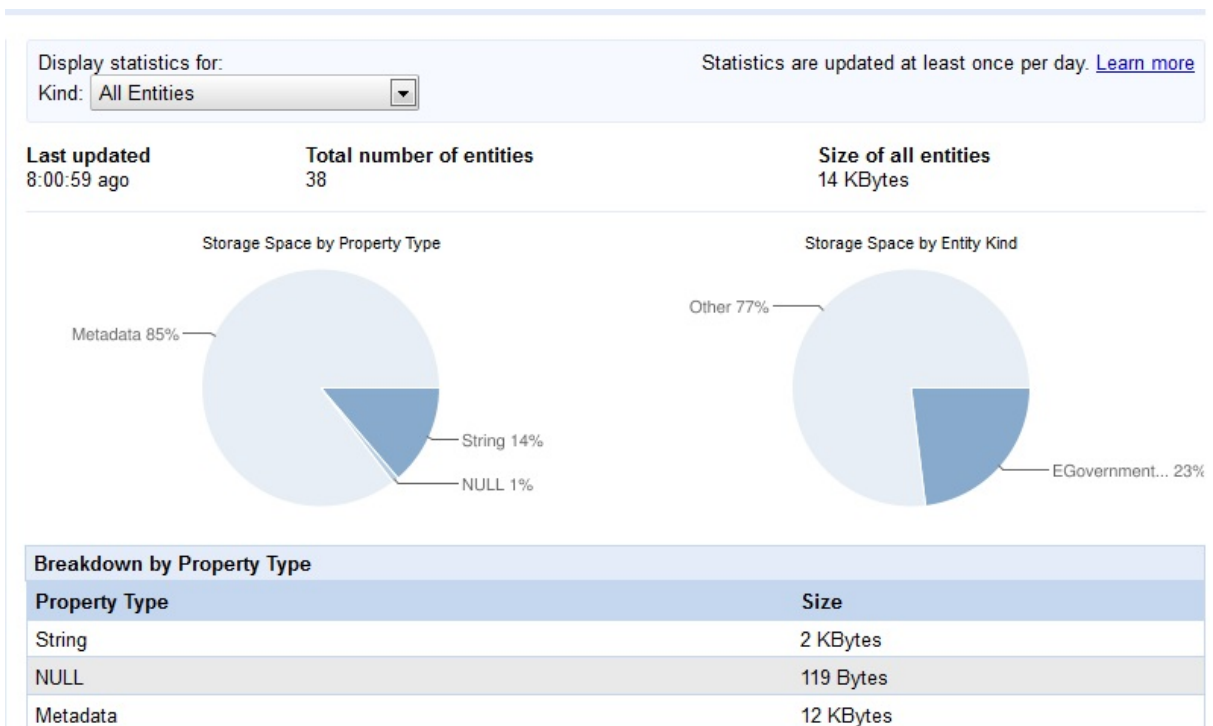


Fig 4.4.11.b: Database entry management tools

An email was sent to ameeek.aziz@gmail.com for verification.

Google Account	Role	Status	Remove Access
aveekbu@gmail.com — you	Owner	Active	Remove <small>The only owner cannot be removed.</small>
raihan99sk@gmail.com	Developer	Active	Remove
ameek.aziz@gmail.com	Developer	Pending	Remove

Admins can use the appcfg [appcfg](#) download\_app command to download your application's code. If you do not want any admin to be able to download code, you can [permanently prohibit code downloads](#). New!

**Invite a user to collaborate on this application**

Email:   
Enter a complete email address.

Role:

**Three developer can working together on a project.**

Fig 4.4.11.c: application developer permission

## CHAPTER V

# PROBLEMS WITH SETTING UP A PRIVATE CLOUD USING OpenNebula

## 5.1 OpenNebula

In our four month of study we understand and learn OpenNebula very good. We follow <http://blog.opennebula.org/?author=32> blog to set our private cloud. But there are some constrains we couldn't overcome within this short term period of learning. The constrains are:

1. Virtual Private Network management
2. Virtual machine management
3. Image repository setup Including Central Data Storage
4. Structural mismatch in database because of new version

### 5.1.1 Network configuration error

We configure our cloud controller network configuration like:

```
/etc/hosts
172.16.0.250 cloud-cc.lan.local cloud-cc
172.16.0.251 cloud-cc01.lan.local
172.16.0.252 cloud-cc02.lan.local
172.17.0.1 cloud-01.san.local
172.17.0.2 cloud-02.san.local
172.17.0.3 cloud-03.san.local
172.17.0.250 cloud-cc.san.local
172.17.0.251 cloud-cc01.san.local cloud-cc01
172.17.0.252 cloud-cc02.san.local cloud-cc02
```

Table 5.1.a: table of network configuration

And configure LAN:

```
/etc/network/interfaces
auto bond0
iface bond0 inet static
bond_miimon 100
bond_mode balance-rr
address 172.17.0.251 # 172.17.0.251 on server 2
netmask 255.255.255.0
up /sbin/ifenslave bond0 eth0 eth1
down /sbin/ifenslave -d bond0 eth0 eth1
auto eth2
iface eth2 inet static
address 172.16.0.251 # 172.16.0.252 on server 2
netmask 255.255.255.0
```

Table5.1.b: table of network configuration

## Network configuration Error:

Screen shot of host drop in Opennebula cloud, Error occurred because of network conflict. Our opennebula error listing:

```
root@espak-pc:/home/espak# onehost list


| HID | NAME      | RVM | TCPU | FCPU | ACPU | TMEM | FMEM | STAT |
|-----|-----------|-----|------|------|------|------|------|------|
| 0   | cluster   | 0   |      | 0    | 100  |      |      | err  |
| 1   | cluster   | 0   |      | 0    | 100  |      |      | err  |
| 2   | cluster02 | 0   |      | 0    | 100  |      |      | err  |
| 3   | host01    | 0   |      | 0    | 100  |      |      | on   |
| 4   | host06    | 0   |      | 0    | 100  |      |      | err  |
| 5   | host05    | 0   |      | 0    | 100  |      |      | err  |


root@espak-pc:/home/espak# onehost enable host02
Host named host02 not found.
root@espak-pc:/home/espak# onehost enable host06
root@espak-pc:/home/espak# onehost enable host05
root@espak-pc:/home/espak# onehost list


| HID | NAME      | RVM | TCPU | FCPU | ACPU | TMEM | FMEM | STAT |
|-----|-----------|-----|------|------|------|------|------|------|
| 0   | cluster   | 0   |      | 0    | 100  |      |      | err  |
| 1   | cluster   | 0   |      | 0    | 100  |      |      | err  |
| 2   | cluster02 | 0   |      | 0    | 100  |      |      | err  |
| 3   | host01    | 0   |      | 0    | 100  |      |      | err  |
| 4   | host06    | 0   |      | 0    | 100  |      |      | err  |
| 5   | host05    | 0   |      | 0    | 100  |      |      | err  |


root@espak-pc:/home/espak# onehost enable all
Host named all not found.
root@espak-pc:/home/espak# onehost enable host.[01,05,06]
Host named host.[01,05,06] not found.
root@espak-pc:/home/espak# onehost enable host05
root@espak-pc:/home/espak# onehost list


| HID | NAME      | RVM | TCPU | FCPU | ACPU | TMEM | FMEM | STAT |
|-----|-----------|-----|------|------|------|------|------|------|
| 0   | cluster   | 0   |      | 0    | 100  |      |      | err  |
| 1   | cluster   | 0   |      | 0    | 100  |      |      | err  |
| 2   | cluster02 | 0   |      | 0    | 100  |      |      | err  |
| 3   | host01    | 0   |      | 0    | 100  |      |      | err  |
| 4   | host06    | 0   |      | 0    | 100  |      |      | err  |
| 5   | host05    | 0   |      | 0    | 100  |      |      | on   |


root@espak-pc:/home/espak# onehost show host05
HID = 5
HOSTNAME = host05
IM MAD = im_Kvm
VMM MAD = vmm_Kvm
TM MAD = tm_nfs
MANAGED = 1
ATTRIBUTES
HOST SHARES
HID = 5
ENDPOINT =
MAX_CPU = 0
MAX_MEMORY = 0
MAX_DISK = 0
CPU_USAGE = 0
MEMORY_USAGE = 0
DISK_USAGE = 0
RUNNING_VMS = 0
```

Fig5.1.1: OpneNebula host error

## CHAPTER VI

### POWER OF GOOGLE APPS ENGINE (GAE) AND OUR RECOMMENDATION

We come up with a decision that we could use Google apps engine as our platform to implement our e-governance services in Bangladesh.

The reason why we decided to pick Google cloud as our platform:

#### 6.1 Resource Sharing

GAE has resource sharing capacity it has implemented BigTable/ MegaStore database. For our implementation we've used their database to implement our own database. So one of the great advantage is, we don't need to design and implement database like MySQL or Oracle database externally.

#### 6.2 Openness

Election commission applications have been implemented based on java programming language. GAE provides great service called openness, means it has capacity to integrate any programming environment into the Google cloud. This was one of the biggest requirements to implement to e-governance in Bangladesh.

#### 6.3 Concurrency

GAE can have multiple application form different vendor and they can run at the same time. In our perspective if we want to run multiple services form our implemented e-governance platform. We can easily do that GAE virtualization.

#### 6.4 Scalability

GAE has extensive power of scalability. To develop a service like e-governance it is like a blessing. We can scale our service as we want to and that is what we need. To integrate our service we just have to use the class instant and implement/ extend the class in java. And this GAE support the entire Device. So we can call it extensive sealable cloud.

## 6.5 Fault Tolerance

Probably the best feature is fault tolerance and reliability. Google provides the best data center in the world. GAE have several data center thorough out the world.

For any reason like power disaster, a natural calamity etc if some of our application server from multiple regions goes down. We could still serve our client/ citizen form rest of the server.

It is almost an impossible case for our government to make such a sustainable network like Google and also it's very expensive too.

## 6.6 Transparency

GAE ensure Transparency to their customer. Both ways Google ensure transparency like:

1. Software architectural point of view
2. Accountable to government

Distributed Transparency means Hide differences in data representation and how a resource is accessed, Hide where a resource is located, Hide Migration resource, Hide the failure and recovery of a resource, Hide whether a (software) resource is in memory or on disk.

So the end user will never know what is going on inside and will not feel any interruption.

You will have some idea how Google is accountable to governments form the link bellow:

<http://www.google.com/transparencyreport/governmentrequests/>



## **CHAPTER VII**

### **FUTURE WORKS**

#### **Develop a complete Election commission**

We would like to implement a fully distributed election commission's application that can be deployed is any cloud platform.

#### **Build awareness about Distributed system and cloud programming**

We will make developers interested to develop their application in cloud platform. Aware the students and programmers in Bangladesh about distributed application development.

#### **Develop a cloud Infrastructure as a service**

Develop a cloud Infrastructure like GAE or Amazon in Bangladesh with the help of open source cloud platform like OpenStack in near future. Build an Infrastructure without any electric power supply and in fully distributed design.

## **CHAPTER VIII**

### **CONCLUSION**

This is as far as I did in my thesis semester. Within a very short time I managed to get these outputs which show a very good opportunity for this system in future. If we can redesign and can continue our work on cloud platform and aware programmers more and more to develop on cloud platform, we will build our e-governance in near future. For a country like Bangladesh e-governance can make a milestone to run government efficiently and people would able to get digital info service form government easily.

Getting assistance from a cloud engineer would be more helpful.

## LIST OF REFERENCES

[Publication]

- [1] Shuai Zhang, Shufen Zhang, Xuebin Chen, Xiuzhen Huo; "Cloud Computing Research and Development Trend". 2010 Second International Conference on Future Networks.
- [2] Janakiram MSV, Cloud Computing Strategist; "Demystifying the Cloud, An introduction to Cloud Computing", Version 1.1 – August 2010
- [3] Junjie Peng, Xuejun Zhang, Zhou Lei, Bofeng Zhang, Wu Zhang, Qing Li ; "Comparison of Several Cloud Computing Platforms". Second International Symposium on Information Science and Engineering.
- [4] Wei-Tek Tsai\*, Xin Sun, Janaka Balasooriya; "Service-Oriented Cloud Computing Architecture". 2010 Seventh International Conference on Information Technology.
- [5] Olav Vahtras, "Getting started on Swegrid", June 4, 2009
- [6] Grobauer, Walloschek, Stöcker: "Understanding Cloud Computing Vulnerabilities"
- [7] "The OpenNebula, Virtual Infrastructure Engine"; Distributed Systems Architecture Research Group, Universidad Complutense de Madrid.
- [8] "Benefits of Cloud Computing", An Intalio White Paper; Ismael Chang Ghalimi, CEO — May 2010
- [9] Johnson D; Kiran Murari; Murthy Raju; Suseendran RB; Yogesh Girikumar; "Eucalyptus Beginner's Guide - UEC Edition", (Ubuntu Server 10.04 - Lucid Lynx), v1.0, 25 May 2010
- [10] Judith Hurwitz, Robin Bloor, Marcia Kaufman, Fern Halper, "Cloud Computing Dummies", in part. 3 1st ed. John Wiley & Sons, Inc November 16, 2009
- [11] Vishanta Rayamajhi, International IT Expert, UNDP Bhutan, "Overall Architecture - Low level e-paltform model", Locatization of E-Governance Project, March 30, 2009

[Internet Resources]

- [1] <http://code.google.com/appengine/docs/java/overview.html>
- [2] <https://sites.google.com/a/systemsplanet.com/google-app-engine-java-links/>
- [3] <http://www.rominirani.com/category/cloud-computing/>
- [4] <http://opennebula.org/documentation:features>
- [5] <http://opennebula.org/documentation:rel2.2:intro>

[Disclaimer]

We like to acknowledge all of my mentors and resources (website,books,other learning materials) I have used to complete my thesis. Due to some constrains we are not enable to mention all those names.