UNIVERSITY OF EDUCATION, WINNEBA

AUTOMATED LOG BASED RECOVERY ALGORITHMS USED TO RECOVER LOST DATA IN CLOUD COMPUTING

PAUL AMOAH

MASTER OF SCIENCE DISSERTATION

2021

UNIVERSITY OF EDUCATION, WINNEBA

AUTOMATED LOG BASED RECOVERY ALGORITHMS USED TO RECOVER LOST DATA IN CLOUD COMPUTING

PAUL AMOAH

A dissertation in the Department of Information Technology Education, Faculty of Applied Sciences and Mathematics Education submitted to the School of Graduate Studies in Partial Fulfillment of the requirement for the award of the degree of Master of Science (Information Technology Education) in the University of Education, Winneba

May, 2021

DECLARATION

STUDENT DECLARATION

I Paul Amoah, declare that this dissertation, with the exception of the quotations and references contained in published works which have all been identified and duly acknowledged, is entirely my own original work, and it has not been submitted, either in part or whole, for another degree elsewhere.

SIGNATURE: ……..…

DATE: ….....……

SUPERVISOR'S DECLARATION

I hereby declare that the preparation and presentation of the work was supervised in accordance with guidelines and supervision of dissertation as laid down by the University of Education, Winneba

NAME: MR. WILLIAM ASIEDU

SIGNATURE: …….

DATE: …….

DEDICATION

This work is dedicated to my children, Frederick Boakye Amoah, Akua Benhene Kufuor-Amoah Cedric Kwadwo Amoah and Reniel Ayeyie Nyarko-Amoah

ACKNOWLEDGEMENT

May the Lord grant me the serenity to accept the things that I cannot change, the courage to change those that I can, and the wisdom to know the difference?

My respect and highest regards to my thesis supervisor, Mr. Williams Asiedu., whose help, patience, passion, guidance and ingenuity has made this thesis worth writing. I am also thanking my parents for their unending love, prayers and support.

To all who in diverse ways also supported me by way of prayers and encouragement, I ask for God's blessings.

Help me to commend my lovely wife Mrs. Benedicta Amoah – as the saying goes "Behind every successful man, there is a woman".

Thank you for your support.

TABLE OF CONTENTS

DECLARATION……iii

DEDICATION……..……..……iv

ACKNOWLEDGEMENT……...v

TABLE OF CONTENTS……vi

LIST OF TABLES……...x

LISTOF FIGURES……..……xi

ABSTRACT……..xii

CHAPTER ONE .....**Error! Bookmark not defined.**

INTRODUCTION .....**Error! Bookmark not defined.**

1.1 Introduction .....**Error! Bookmark not defined.**

1.2 Background of the Study.....**Error! Bookmark not defined.**

1.3 Statement of the Problem 4

1.4 Purpose of Studyt 6

1.5 Objectives..... 6

1.6 Scope of the Study..... 7

1.7 Significance of the Study 7

1.8 Organization of the Study 8

CHAPTER TWO 9

LITERATURE REVIEW 9

2.1 Introduction 9

2.2 Reviews on Software Development Processes and Technologies 9

2.3 Review of Technologies..... 11

2.4 Autonomous Computation 12

2.4.1 Self-Management 14

2.4.2 Self-Configuration..... 14

2.4.3 Self-Optimization 15
2.4.4 Self-Healing 15
2.4.5 Self-Protection 16
2.5 Cloud 17
2.5.1 What's a "Cloud"? 18
2.5.2 A cloud may be public or private 18
2.5.3 Amazon 19
2.5.4 The case of AWS 20
2.5.5 Disaster Recovery Solutions by Amazon 21
2.6 Great service Interruptions 21
2.7 Disaster Recovery 23
2.7.1 Advantages of Disaster Recovery with Cloud 25
2.7.2 Other Relevant Researches 25
2.8 Cloud Computing Service Issues 27
2.8.1 What Service Availability Means? 27
2.8.2 What Service Availability Means in Era of Cloud Computing 28
2.8.3 Measuring Availability 29
2.8.4 Cost of Unavailability 30
2.8.5 Costs of Lost Business 32
2.8.6 Cost of Lost Reputation 33
2.8.7 Cost of Penalties 33
2.8.8 Cost of Lost Worker Hours 33
2.8.9 Recovery Cost 33
CHAPTER THREE 34
METHODOLOGY/SYSTEM DESIGN 34
3.1 Introduction 34
3.2.1 Prototyping Model 34
3.3 The Existing System 34
3.4 The Proposed System 35
3.5 System Actors 36

3.5.1 Users of the System 36
3.6 Entity Relation Diagram 38
3.7 Database Schema 38
3.8 Data Flow Diagram 39
3.9 Implementation 41
3.10 Specification of the System 43
3.10.1 User Characteristics Specifications 44
3.10.2 Constraints Specifications 44
3.10.3 System Specification 44
3.10.4 Software Specification 45
3.11 Recommended System Requirements for the Proposed System 46
3.11.1 Hardware Requirement 46
3.11.2 Software Requirements 47
3.12 Design of the Proposed System 48
3.12.1 Physical Design of the Proposed System 49
3.13 Requirement Specification 49
3.13.1 Functional Requirements 49
3.13.2 Non-Functional Requirements 50
CHAPTER FOUR 51
THE RESULTS OF THE ANALYSIS OF THE ALGORITHMS 51
4.1 Introduction 51
4.2 Performance of the Recovery Techniques 52
4.2.1 Log Based Recovery Techniques 53
4.21.1Deferred Update Method 53
4.2.1.2Immediate Update Method 55
4.3Shadow Paging Techniques 57
4.4Comparison of Recovery Techniques 59
CHAPTER FIVE 63
SUMMARY,CONCLUSION AND RECOMMENDATIONS 63

5.1 Introduction 63
5.2 Summary 63
5.3 Conclusion 63
5.4 Recommendations 64
REFERENCES 65

LIST OF TABLES

Table 1: Availability Percentage and Downtime Allowed……31

Table 2: Downtime Statistics of Top Cloud Service Providers from 2012to 2018…....32

Table 3: Users of the System…….....36

Table 4: Admin…….....39

Table 5: Original Database……39

Table 6: Backup Database…….....39

Table 7: Using Deferred Update…… .54

Table 8: After Update……54

Table 9: Using Immediate Update…….55

Table 10: After Update…….…… 56

Table 11: Comparison of Recovery Technique……..60

Table 12: Statistical Representation of the Recovery Techniques……60

LIST OF FIGURES

Figure 1: Availability Percentage and Downtime Allowed 31

Figure 2: Use Case Diagram 37

Figure 3: Entity Relationship Diagram 38

Figure 4: Data Flow Diagram 40

Figure 5: Nature of System in case of Concurrent Transaction 57

Figure 6: Recovery through Shadow Paging 58

Figure 7: Recovery through Shadow Paging after Crash 59

Figure 8: Graphical Representation of the Recovery Techniques 61

ABSTRACT

Cloud computing is an emerging technological breakthrough that offers a long-dreamed concept of computing as a service. The development of this modern technology in the IT domain has influenced almost all of the private and public sector organizations. Though cloud is implementing the dynamic and cost-effective model of on-demand operation, pay as you go, and assigning resources, security is often the area of concern in terms of its adoption. Moreover, due to the varying nature of the operation and implementation process, traditional in-depth security framework for defense cannot be implemented directly on cloud-based platform. However to safeguard the cloud-based infrastructure, the same principle of can be used. Many entities have used the disaster recovery method to promote continuity of operations. Disaster recovery mechanisms, however, require extensive personal intervention to restore the failed system and this cannot mitigate service outages because it may take longer following a network breakdown to get backup system fully functional. This dissertation proposed an automatic data recovery system that aims to achieve a robust, efficient, adaptable and effective recovery mechanism on a cloud-based system to help and support in this regard.

First, a method was proposed to take advantage of cloud service and automatically restore from volatile network failure, without complicated manual operation. This improves service stability by offering a centralized self-organizing system, which can automatically recognize and restore failures.

Then a model is proposed and extensively explained on the basis of the mechanism. Ultimately a test was deployed based on the concept to show the performance advantages of using this approach in cloud-based data recovery.

CHAPTER ONE

INTRODUCTION

1.1 Introduction

The first chapter shows the dissertation inaugural point. It presents the study's context, description of issue, intent, importance, demarcations and limitations. Finally the chapter outlines the overall structure of the dissertation as well. This dissertation aims at developing an automated cloud based data recovery system.

1.2 Background of the Study

The Internet was designed primarily to be resilient; it was not designed to be secure. Any distributed application has a much greater attack surface than an application that is closely held on a Local Area Network. Cloud computing has all the vulnerabilities associated with Internet applications, and additional vulnerabilities arise from pooled, virtualized, and outsourced resources.

With the massive growth of cloud-based services, there are even more options of storing and sharing information on the Internet. Components including servers and network management solutions have been delivered as services by many cloud service providers. With the obvious benefits in flexibility, high availability and pay-as-you-use basis model, cloud infrastructures have been purchased by many companies as virtual resources to run their sites rather than to host them on their own servers. Over the last years, a large number of cloud service providers have been developed, such as Amazon EC2, Rackspace, and Google Compute Engine etc. Nowadays, thousands of customers of all sizes, ranging from individual users to large companies, are depending on cloud services to host their sites and store their business data.

Along with the accelerating development of cloud services and technologies, some critical issues have been raised, such as the unpredictable connectivity issues and unexpected outages. Even the most popular hosting providers, such as Amazon Web Services and Rackspace, still occasionally suffer from service interruptions or failures.

On 22 May 2013, a flood in parts of Eastern Norway has hit the media group Amedia's data center at Kjeller outside Lillestrøm. The IT system media group and its around 50 local newspapers are impacted by this outrage. On 7 January 2013, Rackspace hosting suffered a widespread of email and application outages (Coombe, 2009). On Christmas Eve 2012, the cloud computing outrage of Amazon Web Services took down many services, especially Netflix, for more than 23 hours [4]. There are even more reports of various downtimes of the cloud services caused by power outage, Distributed Denial of Service (DDoS) attacks, machine malfunction, human error etc. In addition to having caused services shutting down, some crash disasters even have destroyed many customers' data permanently (Zhang et.al, 2010).

System failures like these are bound to happen. The cloud services are already familiar to us, but failure recovery for cloud based services is relatively new. Although, in theory, the cloud model is more failure-tolerant by employing hardware redundancy and/or distributing applications in different data centers on the world, it is still far from total immunity of failures. To help the customers of cloud services to deal with such failures better, many cloud service providers, such as Amazon's cloud, have started to introduce new features that would deal with and minimize the impacts of disasters.

However, even with these new failure-handling features, to build up an operational failure-handling system on top of them still presents a great challenge to many cloud service

customers; for these features, although being nonetheless ample and powerful, have following drawbacks:

They often expose too many technical details. Therefore, to use them properly requires strong cloud related technical background and considerable amount of engineering effort. For example, many of these failure-handling features expect the managed services to be distributed over multiple regions. Such technical detail could be easily overlooked if the companies, which use cloud service, especially small companies, lack of deep understanding of how cloud works. Many companies, although using cloud service, never distributed new services beyond their first AWS data center (Spinola2009). Consequently, these distribution based failure-handling features would never work. On the other hand, improper usage of these features, which is prone to happen, could cause uncontrollable complexity, confusion, and errors, which, in turn, lead to more unplanned downtime.

They are provided not as a systematic solution but a collection of features. System administrator then need to develop tools, often single-purposed ones, based on them. Being limited by the single or a few features in the collection of which they are built on top, these single-purposed tools each usually could only deal with but one specific situation, not to mention their poor ability of handling works crossing operating-systems and infrastructures of datacenters.

The current single-purposed tool based solution, as mentioned in above, which is "De facto" mainstream approach that is made on top base. These new failure- handling features, is still vulnerable to the failures occurred simultaneously to both on-line and standby systems, for it still depends on redundant resources in predefined locations. And even more, it costs extra since

the customers need to pay for hot backups, which stay idle at most of the time in cloud (Zhang et.al, 2010).

Also, it does not provide a means to spontaneously handle failures, which means, the costumers have to maintain a team of engineers at standby to handle potential failures, or to hire a third party to do so. To aid in this situation, administrators need the comprehensive solution to help them automating the whole process of failure handling of cloud services. This project, while providing significant improvements, explores a mechanism to take advantage of cloud service and recover automatically from unforeseeable system failure, without complicated manually management. The design is to enhance the reliability of services by providing a distributed self-organizing structure, which can detect and heal failures spontaneously.

1.3 Statement of the problem

According to the AWS Service Health Dashboard, outages at the data centers are almost happening every day. They may only affect one or two companies for a short time but when there are severe outages on the cloud, so many companies are impacted.

In 24 December 2012, at 12:24 PM PST on Christmas Eve, Amazon Web services experienced an outrage at Amazon's cloud computing data centers in North Virginia that took many services down. It was the fourth outrage of the year in this data center. Netflix (one of most-prominent customer of Amazon) services in the US, as well as some in Canada and Latin America are suffering from this outage. The disruption lasted 23 hours and 41 minutes and coursed by human error as told (Boehm, Gray, & Seewald, 2013).

In 22 October 2012, an outrage started as a small issue affected data centers in North Virginia and gradually affected big area of other parts of Amazon Web Services in North

Virginia. Affected services including Reddit, Foursquare, Minecraft and Heroku, suffered outages. GitHub, imgur, Pocket, HipChat, Coursera and others are having problems. Amazon Web Services said on its status-of-service page that this outrage appears to be a network-related issue.

In June 29, 2012, Amazon's East-1 US datacenter have Knocked Offline by violent storms in Virginia. The outrage has shut down the service in Netflix, Pinterest and Instagram and took with it a large percentage of some of the most popular sites.

In June 14, 2012, some major sites such as Quora and HipChat are taken down. The data center in Virginia was unavailable which resulted in lots of errors and latencies in the US-EAST-1 Region. In addition, the outrage has impacted on Heroku too. Amazon's service indicated that there were power issues (Boehm, 2012).

In August 2011, Amazon and Microsoft cloud services hit by lightning strike. This disaster had damaged power supplies of Dublin data centers and led to server failures and caused downtime on a large number of EBS servers. Although Amazon have managed to bring service back after 48 hours but the effects on businesses could have been serious. In April 21, 2011, this service outrage that considered one of the worst cloud service outages, knocked down big customers such as Reddit, Foursquare, Quora and others for as many as several days. This claimed to be a server problem in the Amazon's northern Virginia datacenter.

Although Amazon is fixing the problems that caused its latest outage, this won't be the last. It is obvious that many common events could lead to failures, such as application failures, power failures, and system overload. It brings to the users unexpected loss of services, if these services are holding on critical applications, the failures on businesses process could be serious. There is a need to create effective DR plan and build reliable cross-region services on cloud platform,

which is also the main objective of this paper. *Nonetheless, none of these approaches suggested a system for an autonomous and complex recovery of disasters that is stable, adaptable and efficient as presented in the dissertation.*

The keyword for the afore-mentioned researches is always regional redundancy that focuses on establishing replicated storage locations in the cloud system that is geographically physically separated. The definition of spatial consistency is also used in this essay but it can provide less manual intervention and better precision with a complex position model of preference.

1.4 Purpose of Study

The aim of this dissertation is to investigate the Disaster recovery in cloud computing using appropriate recovery mechanism to ensure automated self-healing process.

1.5 Objectives

The following objectives were set to achieve the aims of the dissertation.

1. Appreciating cloud computing, what it is, how it operates, and the advantages this system offers.
2. Deploy a technique that recover data lost in cloud systems
3. To provide security, self-optimization and self-management in the cloud application

1.6 Scope of the study

This system is built for MiLife Insurance Company Limited. The core idea is to restore data automatically after system failure, maintain risk, investment and endowment policy transaction details, generate transaction reports and maintain client records.

It is aimed at suggesting an algorithm that are deployed to handle and manage the activities involved in online transactions in an efficient and reliable way. This of course will come with several advantages.

1.7 Significance of the Study

This dissertation is to investigate and propose an algorithm that recover data faster after failure. The system can facilitate MiLife Insurance in performing their daily tasks, improving efficiency, more secured and helping them to be more productive. This dissertation will provide a solution through which MiLife Insurance can easily manage, handle and generate all required information in their respective format when needed. It will help them to manage and protect their cloud data from any form of disaster.

The outcome of this study is significant since it will bring to the notice of cooperate institutions how significant to have a framework for automated system for cloud application which can perform data lost recovery. The new framework is aimed at promoting efficiency, transparency, simplicity and speeding up the recovering of valuation data lost due to system failures. This dissertation will act as a literature for analysis of future studies for individuals (other researchers) who want to explore the topic of designing cloud computing applications.

1.8 Organization of the Study

Study comprises five chapters. These are outlined as follows, the introduction is in chapter one. It deals with introduction, context of the analysis, problem statement, analysis intent and study significance, study restriction, study delimitation, as well as study organization.

Chapter two focuses on reviewing the literature related to it. It usually examines theories about the use of computerized systems in cloud computing, the importance of securing data store on cloud systems and the introduction of data lost recovery approaches.

Chapter three is the approach of the research. This chapter describes; software creation methods, design architecture, E-R diagrams, and Database schema.

Chapter four outlines the algorithms deploy in both the existing and the proposed system. Compare these algorithms and suggesting the best algorithm for used in recovery data faster after system failure. Graphical representation of the analysis have been included in this chapter.

Section five includes summaries of observations, conclusions, guidelines and proposals for further inquiries.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This dissertation seeks to investigate the Disaster recovery framework in cloud computing using appropriate recovery mechanism to ensure automated self-healing process. The discussions are focused on the following aspects:

1. Review of relevant literature on software development processes
2. Definition of automated data lost recovery framework in cloud computing
3. Reviewing the similar system
4. Describe the proposed system.

2.2 Reviews on Software Development Processes and Technologies

Prototyping Model was adopted to achieve this system. Literally reviewing the prototyping model, the researcher will deem it fit to find various expression of this model confirming that it is a technique for faster building a function but incomplete model of the management system. A software development project is assumed by the classic prototyping model and its variations, where the job phases can be explicitly defined prior to implementation. Several modern approaches to software development such as iterative modeling in an effort to solve the limitations of the conventional prototyping model (Basili and Turner, 1975), Suggested quick prototyping (Gomaa, 1983), progressive waterfalls, and incremental progression (Floyd, 1984). Prototyping software engineering methods have been acceptable because they have proved capable of adapting rapidly to changes in consumer needs (Floyd, 1984), reducing the rework needed and assisting in reducing the probability of missing specifications (Floyd 1984).

Researchers have observed that prototyping is cost-effective in splitting the production cycle into smaller, simpler phases to manage (Kaushaar and Shirland, 1985) (Gordon and Bieman, 1994; Boehm et al., 1984). Improves contact between productions workers (Alavi, 1985) helps to assess technological feasibility (Floyd, 1984), is a successful risk control strategy (Tate and Verner, 1990), and contributes to greater engagement of customers and interest in the development process (Naumannand Jenkins, 1982). Over the past two decades researchers have identified specific process models and classification systems for prototyping and prototyping.

The systematic study of 'prototyping' by Floyd (1984) was the basis for a lot of serious prototyping research and so we use his scheme to include a structural structure through which we will incorporate modifications and refinements. For certain areas, samples are used, for example, to create model designs and investigate and control complexity in the design of a model, or to investigate problems in the production process before the final large-scale manufacture of the product. This development methodology has been adopted by the software industry to build prototypes as templates, projections or as partial execution of systems and to use them for a variety of specific purposes, e.g. checking the viability of other functional aspects of a system or as simulation methods to assess consumer needs. In the other hand, prototyping can be interpreted as a 'program' (Floyd, 1984), which is either a well-defined step within the life cycle of software creation, or an 'approach' that influences the entire process (Budde et al., 1992). Prototyping approach can facilitate effective framework development by breaking a complicated and often unsolved problem into much more detailed yet smaller and simpler sections (Kaushaar and Shirland, 1985). A prototyping approach to technology can help build a product to address end-consumer or market needs and then develop them (Gomaa H., 1983).In this study, Gomaa H,

1983 explored what prototypes are, what the prototyping method is, and how the software engineering methods follow prototyping for discovery, innovation.

2.3 Review of Technologies

Passey et al. (1997) described a number of innovations with management consequences for the use of internet services in schools and homes. A specific outcome illustrated the need for organizations to analyze intent closely when engaging extensively in networking technology, as realistic usage did not fuse until consumers had certain pre-existing definitions about future applications. A lot of studies further reported that impact on management when laptop technologies were being used (Butterworth-Heinemann, 1999). Of special interest was the finding that parents are expected to use ICT increasingly for contact with schools in the future, and that that will entail supervision, just as any other aspect of essential partnership between homes and schools needs to be handled? More recently, the consequences for management were analyzed and documented in terms of staffing requirements, financial needs, and recruitment needs (Jossey Bass, 1995). When ICT use pervades to a degree, it is clear that the scope of roles and duties expands, and the importance of financial and procurement management is much an.

In the digital world today as we are trying to create a software program more acronyms fall into our heads. This is very difficult to determine the technologies to use to create an application that better fulfills the company objectives, taking into account certain considerations such as cost and speed of production. Although cost is a very important consideration, machine stability is the most important. A program that requires low cost of production but fails to achieve the market target will not be a reasonable option. Therefore the integrity of the network will be the first priority for any organization who wants to start a business on the World Wide

Web where success is just for the best. And the same thought was in my head for this web shop to improve the infrastructure by using those innovations which are the best to address the cost and reliability challenges. And PHP, MYSQL, HTML, and CSS are the software used by this web program.

Using PHP the general business logic was introduced. MYSQL was used to include the system's database capabilities and store customer data including account information purchase specifics and total buy history. The system's front end was developed using HTML and CSS. Those are the market-proven innovations for creating the worldwide appealing and secure web application. When using the CSS (cascading style sheets) all the front end template settings are contained in a single CSS file so if there is a need to make certain adjustments then we don't need to alter for each page then we just adjust the value in the CSS file so that update is made on the whole website. For instance, if we want to change the background color of our website we don't need to adjust the background color on each page we just edit the background color from CSS (cascading style sheet) file and modify the background color of all website updates by modifying the single value. So that is the downside of having these systems because they are so basic and have a lot of functionality that make it very convenient to incorporate the entire process.

2.4 Autonomous Computation

Autonomous Computations are gradually becoming confounded. In the development of the multifaceted nature of IT infrastructure supervision, interconnection and coordination in the arrangement of diverse programming, it has becoming more complex to keep the systems configured and run by human inputs alone.

On October 2001 IBM introduced the concept of autonomic computing to cope with this problem. In the manifesto released by IBM, it highlights a software complexity crisis as the main challenge for the future IT industry. The task of creating, implementing, and maintaining devices goes further than managing single software environments. Today's computer system is getting too large and too difficult to handle. And the difficulty, uncertainty, and machine failures that arise from insufficient setup and maintenance may lead to further unplanned device problems. This condition has forced increased administration-related workload, exceeding the limits of human resources (David & Douglas 1991).

The only residual option is autonomous computation. An autonomous computation system, is a computer system which is capable of controlling itself according to the goals of the administrator. It runs itself, senses and responds automatically to different situations without the assistance of manual manipulations.

Autonomous computation study is still on a timely era, it overpowers existing devices and systems capabilities. Over the last two years, autonomic computing has added more than 20 workshops and gatherings to this exploration, providing analysts with a wide range of software engineering, including programming building design and artificial intelligence.

The purpose of an autonomous computation system is creating a system that has a lifetime wealth. Components develop into the device just as the cell process develops itself within the human body. They has auto-configuration, self-optimization, self-healing, and self-protection properties (David & Douglas, 1991).

2.4.1 Self-Management

The goal of autonomous computing systems self-management is to relieve administrators from system configuration details, and let the system continually adjust and run on its own at its highest efficiency. To be autonomous, a system must know itself, know its context, which ensures that all elements involved in the system must have an identification with the system. The autonomous system has to track its own actions on an ongoing basis and control shifts in its surroundings and climate. When finding errors, the program must return to the previous version and try to eliminate the errors using its method for question identification (Gray, John, & Atwood, 1992).

As mentioned above, along with autonomic computing there are four aspects of self-management. The four things often quoted in the representation of IBM is self-design, self-optimization, self-healing and self-protection. These factors are usually taken into account separately in early autonomic systems. The normal approach is to handle failures that relate by means of specific solutions Nevertheless, these things can gradually be merged and incorporated into a general design with the advancement of the automation technologies.

The four elements are explained in the following way:

2.4.2 Self-Configuration

To accommodate different conditions, an autonomous computing system must customize itself and reconfigure itself. Autonomic machine architecture implements high-level rules that understand what is expected, rather about how that's going to be handled. When integrating a new function into a network it automatically adapts to the general network setup and then interfaces with the device itself. Via this method, some components can learn about the new one,

obtain the information for use, or change it to function for it (David & Douglas, 1991).

2.4.3 Self-Optimization

Currently the complexity of middleware is rapidly increasing, a database system that can have many tunable parameters. Consequently, it may be impossible for even the most skilled IT administrators configuring them correctly and ensuring optimal efficiency over the whole network. An autonomous system should constantly look for ways to track and maximize its efficiency or expense, rather than remaining for status quo. This approach is linked with the biological concept, in which the simulation of human body is autonomous computing. Just like the brain adjusts its structure through daily practice and practicing to function better. Autonomous systems need to learn how to spool their own dimensions correctly and attempt to upgrade their apps to meet customers' desires without human interference or limited interference.

2.4.4 Self-Healing

Because autonomous computing system normally works in dynamic environments that change mostly during routine operation. Continuously assessing the status of the system and, moreover, discovering malfunctions, diagnosing and responding to system disruption, and automatically applying appropriate corrections (Gray, John & Atwood, 1992). Components must track themselves constantly to maintain the integrity of the system. The performance of the individual component is translated into estimation using the problem-diagnosis method; the damaged devices are detected and separated. After the analysis point, device can recover from errors by repairing, removing, rebooting, or isolating the faulty part. The aim of the self-healing

program is to reduce program error impacts and to bring the fixed component back to the network without lack of service. The program can try to anticipate the problems, and so take action in advance to avoid failures.

2.4.5 Self-Protection

An autonomous computing system cannot escape in an isolated environment without external influence. In order for a system to be self-protecting, the device must be able to protect itself in order to prevent vulnerabilities triggered by such incidents as malware attacks, unwanted access and user disoperation. It monitor other aspect of the system according to security policies on an ongoing basis, and takes appropriate action to maintain component compliance. IBM's recent offering of security solutions is to provide effective rules for automating security compliance management. First, the device needs to deal with risks as they arise and defend itself from assaults from anywhere (Gray, John& Atwood, 1992) and report such activities.

Autonomous computer technology has many advantages of self-managing, overcoming the complexities of current computing system; it avoids the burden of system management and decreases administration costs. Even the task of exploring different facets of autonomous computation and implementing systematic autonomous behaviors remains open, the demands of new technologies and industry standards, the exploration of work into automated computing systems has just started.

2.5 What's a Cloud?

The cloud is a modern technology of digital technology which offers computation resource (i.e. software, hardware, networks, data, and cloud-based user interfaces). Recently the cloud service has become more popular with business applications (Gary, John, & Atwood, 1992). The cloud-based technology is supplied essentially as the applications, network, and computing facilities. Over the past decade, cloud computing have prompted much enterprise organizations to switch their data to data centres with shared computers and storage facilities, rather than hosting their servers centrally.

Amazon EC2, Rack space, Google Compute Engine, Windows Azure, HP etc. are examples of these leading cloud service providers.

The cloud infrastructure has functionality that set it apart from the conventional hosting network. It offers the service as "On Demand Self-Service," allowing the user to choose the CPU, memory, bandwidth, and network access based on their demands. Both computing services are shared across the network and available by multiple client computers (i.e. Computer, laptop, cell phone) (Kehoe, & Bougton, 2001).

The accelerated elasticity is another significant aspect of cloud computing. With little management effort, computer tools can be provisioned and published dynamically. Nearly unlimited computing capacity is provided to the users via cloud servers (Kehoe, & Boughton, 2001). The programs are available and distributed to customers anytime and wherever they can need them. In the cloud, customers scale up and down services in minutes, based on changes in specifications.

Additionally, users can conveniently leverage the pay-as-you-use base model of the cloud service. Leasing the cloud service network infrastructures promises to minimize operating

expenses (Kehoe & Boughton, 2001). You have the benefits of reducing the expense of running numerous infrastructure equipment, renting data center floor space, and the expense of computing in the cloud. The level of expense only represents the real use of cloud services. Cloud-based computing has improved the way businesses treat their clients entirely, with the advantages of cost reduction, flexibility, and elasticity.

2.5.1 A cloud may be private or public

Cloud providers are usually classified into three divisions, i.e. SaaS (Software as a Service), PaaS (Service as a Platform), and IaaS (Infrastructure as a Service).

Software as a Service (SaaS): The SaaS model, defined as "on-demand apps," offers links to business-relevant systems and numerous cloud-based technologies like virtual office, e-mail, CRM, high-performance computing, and gaming. Applications are preconfigured and are taken care of by cloud service provider. This abstracts all the infrastructures surrounding the app free. Customers do not rely on managing the equipment that devices run on or keeping devices going. That's what it wants to do on the internet to pay for it on a pay-per-use basis.

Cloud vendor sources delivering SaaS: Google Apps, Zoho, Microsoft Office 365, OnLive, and GT Nexus.

Platform as a Service (PaaS): The PaaS architecture includes the program delivery services. It offers a database infrastructure plus the black-box applications which can be built and administer on top of the cloud platform by application developers. These may include program creation, testing, and implementation tools. There's absolutely no need to waste time and money installing and handling the components with the PaaS app. Data service offers and develops virtual infrastructure, programming software, servers, and Web server.

Examples of PaaS providing service providers include: Google Web Computer, AWS Elastic, Windows Azure Compute, Engine Yard and Oranges cape (Gray, John, & Atwood, 1992).

Infrastructure as a Service (IaaS): The emphasis of the thesis is this. The IaaS is the simplest cloud computing model that typically provides a wide variety of hardware and virtual computers and other computer services such as processing, storage, and networking to customers. In this architecture, the vendor offers the uniform network, usually virtual servers, that the customer can execute arbitrary applications. Such of IaaS providers include: Amazon EC2, Rackspace, HP Cloud, Windows Azure Virtual Computers, Google Cloud Servers (Gray, John, & Atwood 1992).

2.5.2 Amazon

Amazon Web Services (abbreviated AWS) is among the major cloud service providers. A cloud providers are the companies that offer cloud computing platform infrastructures and services over the internet to other organizations or individuals.

AWS Officially launched in 2006, it began offering a range of electronic computing infrastructure such as data storage and web-based program delivery for enterprises. Amazon is one of the major providers of cloud storage with some of the biggest and well-known businesses such as Netflix, Pinterest and Dropbox. Amazon EC2 and Amazon S3 are the principal services. Amazon EC2 provides services, application platforms and operating system for customer application hosting in virtual computing environments. Amazon S3 supports cloud computing, which provides customers with an online interface to access their stored data anywhere on the

globe. Amazon Web Services today provides more popular modules like Amazon RDS, Amazon Basic DB and Amazon SQS etc. Cisco, Citrix, Google, Microsoft, Rackspace and Verizon / Terre mark are other cloud providers.

2.5.3 The case study of AWS

In explaining any feature of Amazon EC2 is outside the reach of this thesis; it will present the models of Amazon EC2 which are most important for Disaster Recovery (DR). You must sign up for an account to continue using AWS and open your phones. AWS Management Console facilitates and manages cloud infrastructure in a convenient way. The AWS gateway provides a variety of AWS software to help clients build their own cloud applications or leverage the different cloud services available from AWS. Application hosting is another of the key utilities: running under the Software-as-a-Service (SaaS) architecture, Amazon Web Services offers platforms such as Amazon EC2 and Amazon EBS to allow providers of applications or other users to host their current services. (Boehm, Seewald & Gray, 1992).

Backup and Backup: AWS offers data management facilities such as the Amazon S3. Customers will scale up and down AWS storage as they have to store records, data and archive or use it as an archive location for disaster-recovery solutions.

Content Delivery: Every consumer uses the Amazon Cloud Front program to Provide Web-based products and resources, including images, videos, applications or other media, to end-users with fast Internet data delivery capabilities.

Databases: AWS has offered various database options including Amazon RDS, Amazon Dynamo DB, Amazon Simple DB and a variety of relational databases that can be operated by customers themselves. The Relational Amis Amazon EC2 and Amazon EBS have total control

over instances.

E-Commerce Applications: Amazon Web Services offers a variety of tools to help consumers maintain a stable, reliable e-commerce platform, like Amazon FPS.

Maximum capacity computation: Potential of accessing massive volumes of processing power resources on the cloud, this technology primarily targets customers from sectors that need high computational efficiency and high broadband for handling their extremely resource-consuming tasks, such as storing very broad data sets through multiple virtual situations.

2.5.4 Amazon Disaster Recovery Solutions

- Different Locations-Place instances in various accessible regions and areas. In other cases, several locations may help to separate faults.
- Elastic IP addresses — Having the power to mask instance or zone compatibility faults by dynamically remapping the dynamic IP address in the same account to some other instance;
- Balancing elastic load - Detection within a database of unsafe incidents and automatic redirection of congestion to appropriate instances prior to retrieval of unhealthy incidents.
- Save and recover backups of patterns to current networks in ahead of time, and recover backup if a malfunction happens.
- Phase Pilot Recovered Key Replicated Light Set Data Collection as Standby Solution. Turn to crisis contingency system (Boehm, Gray & Seewald, 1992).

2.6 Great Service Interruptions:

According to the AWS Service Health Dashboard, there are almost daily outages at the

data centers. This will only affect one or two businesses for a limited period, but then there are significant outages on the internet, then other businesses will be impacted.

Christmas Eve around 12:24 PM PST, 24 December 2012, Amazon Web Servers suffered an earthquake at Amazon's North Virginia cloud data storage centers that disrupted most infrastructure. The data center was the year's fourth outrage. This disturbance impacts Netflix networks (one of Amazon's most common clients) in the U.S., and many other networks in Canada and Latin America.

The controversy exploded as a minor issue on 22 October 2012 hit data centers around North Virginia and subsequently corrupted large portions of the Amazon Web Infrastructure in North Virginia. Applications that have been affected have suffered outages, including Facebook, Foursquare, Minecraft and Heroku. Reddit, imgur, Facebook, HipChat, Coursera and others have issues with this.

On June 29, 2012, Amazon's East-1 US datacenter, Pulled down by extreme storms in Virginia The outrage dismantled Netflix, Pinterest and Instagram's service and took a large proportion of some of the most major sites with it. Several big pages like Quora and HipChat was shut offline in June 14, 2012. The Virginia data center in the US-EAST-1 area became inaccessible resulting in a number of errors and latencies. In fact, the outrage has influenced Heroku, too. There were control problems indicated by Amazon's web site (Boehm, 2012).

Cloud infrastructure from Amazon and Microsoft struck by lightning strike in August 2011. This accident had destroyed power supplies to the Dublin data center which resulted in system crashes and a vast number of EBS systems suffering downtime. While Amazon succeeded in getting back service after two days, it may have had significant consequences on companies. The internet disaster that was considered to be one of the worst cloud service outages

in April 21, 2011, took down major users including Pastebin, Foursquare, Quora and others for as many days as before. That seemed a server issue within the Northern Virginia datacenter of the Amazon.

While Amazon fixes the concerns that caused the latest shutdown, it isn't going to be the last. It's clear that several common accidents will lead to crashes, such as system failures, power failures and computer overload. It brings unintended service interruption to the customers, if such systems maintain essential programs, the business process errors may be severe. There is a need to establish an appropriate Disaster Recovery system and develop stable cross-regional infrastructure on the cloud network, which is also the main goal of this study.

Whereas the cloud dramatically extends solutions for disaster recovery, web-based Disaster Recovery technology work is still nascent. Available Disaster Recovery approaches either expensive, or concentrate only on better data replication efficiency or backup site response time. Today, businesses face increasing amounts of data to protect them. The continuity of service is a vital requirement for many businesses to operate smoothly. Therefore, Disaster Recovery (hereafter DR) program is used to mitigate the harm done by service disruption IT system failures.

2.6 Disaster Recovery frame work

The Disaster Recovery Mechanism (DR), which focuses on IT or infrastructure systems in this thesis, is a method and technique used by an organization to prepare and recover from a natural or man-induced catastrophe and to help mitigate or prevent damage. This may include recovery of program bugs, system deficiencies, networking challenges,

Today, companies face growing amounts of data to secure them. The quality of operation

is a critical necessity for many businesses to operate smoothly. Thus Disaster Recovery (hereafter, DR) program is used to mitigate the harm done by the network disruptions leading to interruptions of operation. Disaster Recovery plans come in various forms: typical Disaster Recovery plans are based on backups of resources examples are tape backups, disk backups and Wireless Area Network backups. If the main server is offline, the more recent mirrored data must take over the backup. And usually there are several ways to position the backup sites: (Gray, John, & Atwood, 1992)

- Onsite: the primary server is located in the same position as the backup network.
- Co-location: Alternate location of the backup device, isolated from the main location.
- Cloud: Put the backup system in the cloud.

The key idea for a good contingency plan is to often physically discern between the active sites and the reserve sites. A small number of disasters suffered is much localized (Gray, John, & Atwood, 1992), as indicated in a Symantec study.

However, conventional Disaster Recovery solutions entail extra maintenance costs, because they need not only an additional reserve station but also an efficient network link to it, ensuring that main and backup data are both coordinated. Since the backup location needs all the essential requirements, it typically costs the same. It means that the IT service costs are actually doubled in order to provide a backup system afterwards. Because of the very high financial expense, Therefore, some companies are restricted to a Disaster Recovery strategy only, and in addition, according to Eweek, over 40 per cent of small companies do not have a formal Disaster Recovery programme.

2.6.1 Advantages of Disaster Recovery with Cloud

Luckily, the great important merits for adopting Cloud is its cost-effective and very robust capacity for recovery from disasters. Second, the expense of setting up a Disaster Recovery system is minimized with multiple cloud providers adopting the pay-as-you-use model. Firstly, no ad-hoc network access must be enabled to Cloud, because the server is already attached to the Internet, of course. Again, hardware costs are often divided among Cloud users. Wood also provided the comprehensive cost study, demonstrating substantial cost savings relative to conventional Disaster Recovery plans for cloud-based DR plans (Gray, John & Atwood, 1992).

The advantage of having Disaster Recovery plan in Cloud is that time can be dramatically reduced to create a backup server. Thanks to virtualization technology, the "backup site" divided into a piece of data trunk, along with operating system, applications, data and system status information, and forwarded to the cloud. Since the time to set up a reserve site in a cloud is too limited, you can also set up the backup site on request and put it up until the tragedy is found. Although cloud-based Disaster Recovery software does show its tremendous impact for modern industry, a final recovery measures from a catastrophe often require some level of human intervention. An automated recovery process is always favored for the sake of efficiency.

2.6.2 Other Relevant Researches

Although the cloud widely expands options for disaster recovery, cloud-based Disaster Recovery approaches research is still nascent. Traditional Disaster Recovery solutions are either expensive, or concentrate on data replication effectively and backup site response rate. Some studies are currently aiming at minimizing service interruption during the fault recovery phase by offering

efficient data availability and failover capacity from the main module of data centers. The primary emphasis is to provide several replications of the main system and to spread them into various geographic areas as copies of the original program (Gray, John, & Atwood, 1992).

On the storage side, SAN has also introduced a cloud computing device that uses the Network Storage Area (SAN) with a range of high-performance networking resources that will provide the client with secure links to get reserve data. And because the SAN expects a huge number of routing facilities in maintaining the data transport network intact and, therefore, adequate power to achieve the optimal response time for the operation, the cost of hardware is comparatively expensive for start-ups enterprises.

Caraman, M's paper proposed an innovative approach for developing an end-to-end Disaster Tolerance (DT) solution that would support IaaS server. Disaster Tolerance (DT) refers to the ability for a device to recover after a catastrophe and return to operation within relatively short periods of time. Current research primarily focuses on Remus, a service HA method applied in the virtualization layer (Gray, John, & Atwood, 2000). Caraman, M's solution involves a seven-stage Disaster Tolerance algorithm covering all levels of the IaaS cloud, but it's also meets the conventional Disaster Recovery strategy involving double data centers, The primary executes and the replacement is idle. That approach includes a full disk replication process, thus by replicating the original disk state with the new disk improves failure resistance on the fly. A high-speed communication between the data centers is a strict requirement for this approach. Shriram Rajagopalan presents Second Site Implementation. This approach expands the high-availability framework Remus module by enabling the duplication of a number of data centers through large-area Internet links.

Other option for greater data replication efficiency is synchronous pipeline replication.

The solution tackles performance decrease caused by WAN replication latency. Multi-tier servers are used to monitor the replication. The programs log the effects of the disk modifications, and continue at a point of recovery.

Not a single solutions has proposed an autonomous and robust approach to disaster recovery, as captured in this thesis. The focal point of the above-mentioned researches is always regional redundancy, which focuses on establishing replicated storage locations in the cloud system that is geographically separated.

The definition of spatial consistency is also used in this thesis but it can provide less manual intervention and better precision with a complex position model of preference.

2.7 Application Challenges in Cloud Computing

As more firms move their corporate operations or information into the internet, the quality of operation becomes highly uncertain. Ensuring that the cloud services are reliably available remains a difficult problem for cloud platform providers. Several recent infrastructure disruptions have brought into question infrastructure system reliability for critical sensitive systems in operation. Amazon Web Services (AWS) went down for 49 minutes on January 13, 2013 and was officially losing almost US$ 5 million as a result. Click on Amazon.com, operating on the AWS cloud storage network, returned 503 errors indicating potential service attack denials. Many big cloud platform providers including Google and Microsoft Azure have experienced their share of outages.

2.7.1 What Service Availability Means?

In the thesis, the term quality of service was used several times with the presumption that

the significance is clear to the reader. This region is working to ensure that we share the same idea of what accessibility program plans for the customer. According to a review of the Service Consistency Website, service quality is: "an expression of high performance, in comparison with available resources irrespective of operating systems, application or system failure and impact;"

A crucial condition of simple mission systems is that they must sustain continuous operation even in the event of disappointments in equipment or programming, that is, they must be respectful of responsibility. To accomplish this requirement, SAF has established four main criteria. They are true to: (Gray, John, & Atwood, 1992):

- Redundancy.
- · Recovery from errors stateful and smooth
- Reducing Mean Time To Repair (MTTR)
- Surveillance.

These concepts require not only responses to failure but also constructive control under which action is taken before failure happens. These concepts refer to cloud infrastructure systems, which will be covered in the article afterwards.

2.7.2 What Service Availability Means in Modern of Cloud Computing

For every IT system, usability is a key aspect, because lack of connectivity means lack of development. In recent years, coverage of server outages has grabbed headlines and there are significant financial consequences in most instances. How does the quality of service in a cloud system correlate with the quality of service in conventional IT infrastructures, however? Failures in a program are possible, but why is cloud storage such a big deal? Has the lack of cloud

computing service quality been significantly underestimated given the established track record of cloud success? The solution lies in the way growing infrastructure delivers service. An overview of how services are obtained in the cloud and in conventional IT networks is shown in figure 2.1 and figure 2.2. Cloud application hosting resources include database, block storage, object storage, runtime, computing, network and more. Given that a cloud infrastructure platform such as AWS houses many million programs and more than two trillion computer items (Gray, John, & Atwood, 1992), a full termination of operation means denial of access to the host software and computer that could devastate the global IT field. In this way, a cloud network can be seen as one big single point of limitation.

2.7.3 Measuring Availability

To solve the problem of quality of service in distributed computing, would need an approach to measure accessibility. Usability generally refers to the amount of time a system is up, a 100 per cent usability platform is always up. Abbadi articulated in, the quality principle as:

Availability = MTBF/ (MTBF + MTTR) ∗ 100

MTBF refers to the mean time between failures, and MTTR refers to the mean time to repair.

For example, a scenario in which a cloud user suffered 24 hours of downtime in a month, here is how to quantify the availability for that month:

MTBF = 30 days * 24 hours * 60 minutes = 43200 minutes

MTTR = 24hours * 60 minutes = 1440 minutes

Recall,

Availability = MTBF/ (MTBF + MTTR) ∗ 100 Therefore,

Availability = (43200/43200 + 1440) ∗ 100 = 96.77%

Similarly, say a cloud service-e provider promises uptime of 99 percent a year, this deal seems fantastic, except it might mean 3 days downtime a year which is terrible for mission-critical applications. Here's how to measure acceptable downtime in one year:

Total uptime in a year = 12 months * 30 days * 24 hours * 60 minutes = 518400 minutes The Cloud Provider's MTBF in a year= 99% Allowable MTTR in a year (percentage) = 1%

Therefore, Allowable MTTR in a year = (518400 ∗ 1%)/100 = 3.65days

Table 2.1 lists the approximate amount of downtime allowed for a cloud service provider to achieve a certain percentage of availability.

The International Working Group on Cloud Computing Resiliency (IWGCR) provides a summary of downtime statistics of some top cloud service providers from 2013 - 2018 as shown in table 2.2 (Abbadi, 2008).

According to (Anna, 2008), approximately 165,000 websites went down for a period of one week due to NaviSite outage in 2007. A failed data center migration was cited as the cause.

2.7.4 Cost of Unavailability

Cloud services failure can lead to a decrease of profitability, sales and credibility. The cumulative effect of the failure is determined by accumulating all of the downtime damages sustained. Based on the programs delivered the following considerations should be weighed.

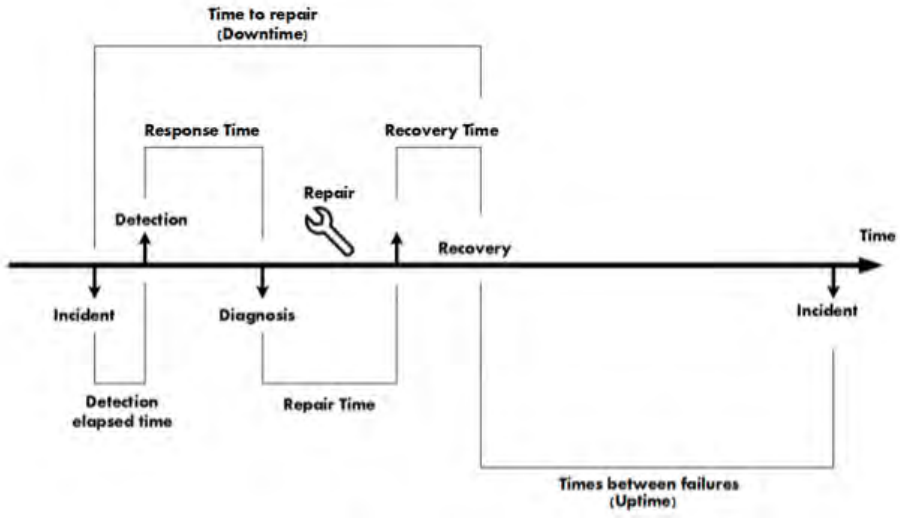


Figure 1: Measuring Availability

Table 1: Availability Percentage and Downtime Allowed

Uptime (%)	Downtime (%)	Downtime Per Year	Downtime Per Week
98	2	7.3 days	3 hrs., 22 mins
99	1	3.65 days	1 hr., 41 mins
99.8	0.2	17 hrs., 14 minutes	20 mins, 10 secs
99.9	0.1	8 hrs. 45 mins	10 mins, 5 secs
99.99	0.01	52.5 mins	1 mins
99.999	0.001	5.25 mins	6 secs
99.9999	0.0001	3.15 secs	0.6 secs

Table 2: Downtime Statistics of Top Cloud Service Providers from 2013-2018

	2013 (hr.)	2014 (hr.)	2015 (hr.)	2016 (hr.)	2017 (hr.)	2018 (hr.)	Total (hr.)
Amadeus	2			3	2		7
Facebook	1	2	2	1	1	3	10
Server Beach	4	1	2	1	2	2	12
PayPal	1		2	1	2	3	9
Google	1		4		1	2	8
Yahoo!		3		1	6		9
Twitter		7	2				79
Amazon	2		24				26

2.7.5 Costs of Lost Business

It applies that immediate financial debt resulting from a system failure. Take, for example, a trading network where an outage of a few minutes will turn into impacts of several million fraud. Global publisher of the Network Brandon Butler wrote in.' the recent earnings report from Amazon.com shows the company is making approximately $11.8 billion a year, or around $118 million a day, about $4.9 million an hour. "That average that Amazon.com will suffer a deficit of about $5 million with any hour of downtime (Harland, Lamming & Cousins, 2009).

2.7.6 Cost of Lost Reputation

Outages will harm the credibility of a cloud service provider in the long term. Credibility matters to customers because the choice of a service is usually based on a concept of trust dependent on credibility and proof of availability.

2.7.7 Cost of Penalties

Any cloud service providers and users pay termination payments in the case of an interruption. This is especially so when sensitive consumer data is lost.

2.7.8 Cost of Lost Worker Hours

Loss in profit in the long term can be expensive for cloud service providers and consumers because of missed working hours. In real instance permanent workers are paid irrespective of how they achieve their job targets.

2.7.9 Recovery Cost

This applies to the expenses borne to recover from an accident, such as paying workers for extra labour, and resources for disaster recovery.

CHAPTER THREE

METHODOLOGY/SYSTEM DESIGN

3.1 Introduction

This project seeks to investigate the Disaster recovery framework in cloud computing using appropriate recovery mechanism to ensure automated self-healing process. This third episode describes the method employed to attain the goals of the mission. This chapter again explains details requirement engineering relevant to the project.

3.2 Prototyping Model

This framework was developed using Prototyping Pattern. The Prototyping model is a methodology for quickly creating a management platform structure yet unfinished model.

This approach was adopted in order to benefit from the advantages of prototyping including;

- Reduction in development time
- Reduces production expenses
- Improved user participation
- To provide customer input that is measurable
- Makes program execution smoother, because consumers know what to expect
- Making clients more satisfied

3.3 The Existing System

According to the Information Technology administrators of MiLife Insurance Company limited, there are no automated data loss recovery system for most enterprises that have the cloud as their backbone. The current system is manual so whenever the cloud service providers are

being struck by disaster which could be as result of system failure, fire outbreak, flooding, power outages or human destructions. They use manual backup systems to back up their systems retrieve to the lost data depending on the time ranges specified. It is very difficult for Information Technology managers to track impromptu issues making the existing system more vulnerable to data loss. Backup systems are manual which is very risky.

3.4 The Proposed System

The automatic framework was introduced after evaluating the current system. It is the automated recovery framework for data loss; the system is developed using PHP and MYSQL, the interactive network technologies. Administrators can add new accounts, erase existing users or modify old ones. Administrators should be able to change user identities. The system automatically checks and updates itself if there is a file missing or corrupted or lost due to system failure.

The recovery framework is developed using the PHP and MYSQL hybrid network technologies. The system will be able to automate the following functions.

- Administrators will generate new accounts, erase existing users or change old ones.
- Administrators modify user accounts.
- The system will auto heal itself.
- System checks for data loss based on time specified

3.5 System Actors

System actor is anyone who must periodically collect, verify, join, react, store and share data and information using or being influenced by an information system. They are: Administrators and Customers.

3.5.1 Users of the System

Characteristics of users: This product's customer should be reasonably informed about the device use. He should understand how to store items, and should have knowledge of specific items to save them. A person who has no programming skills would find it hard to comprehend the program. Yet the idea should be very easy to do with a bit of experience.

Table 3: Users of the System

Users	**Activities**
System Administrator	Add clients info, edit and delete files
System	This application allows itself to auto heals itself from data loss, checks and recover if data are lost due system failure

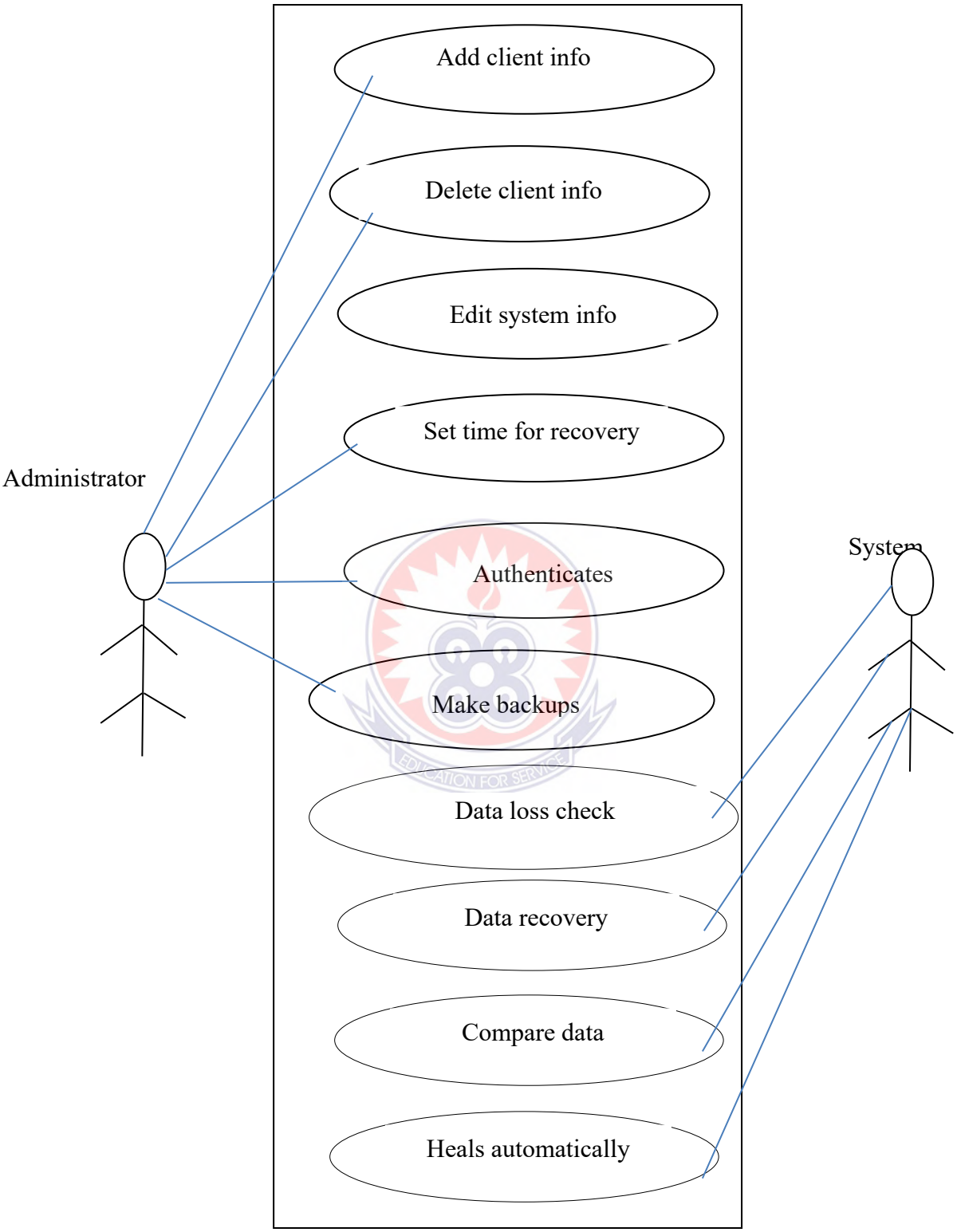


Figure 2: Use Case Diagram

3.6 Entity Relation Diagram

An entity-relationship (ER) diagram is a dynamic diagram which shows the interrelations within an entity data base. ER diagrams often use the symbols to describe three various data types. Boxes are widely used to describe people. Additionally, diamonds are used to depict partnerships, and characteristics are represented by ovals.

An entity-relation model (ERM) is an abstract and conceptual representation of data in information engineering. Entity-relationship modeling is a method of computing the configuration of the relational schema, used to generate a system's type of logical schema or semantic data model, typically a relational database, and its parameters top-down.

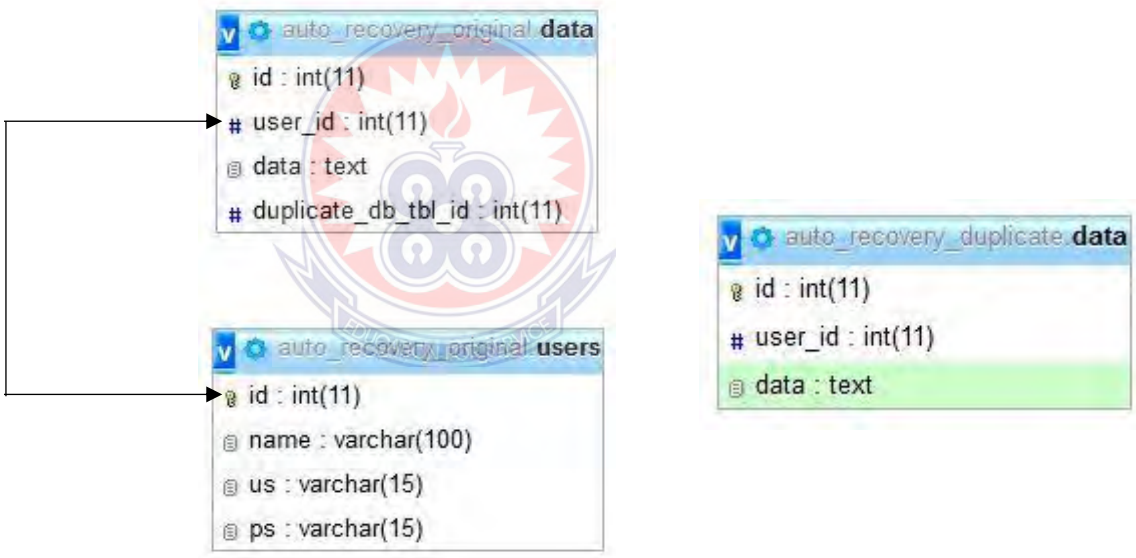


Figure 3: Entity Relationship Model

3.7 Database Schema

A database schema for a database is the framework specified in a structured language provided by the database management system (DBMS) which relates to data administration to construct a structure of how a database should be built (divided into database tables). Relatively the database

schema refers to the database architecture which indicates how the data is organized into tables or structures.

Table 4: Admin

Name	Id	Name	Us	Ps
Data type	Int (11)	Varchar (100)	Varchar (15)	Varchar (15)

Table 5: Original Database

Name	Id	User_id	Data
Data type	Int (11)	Int (11)	text

Table 6: Backup Database

Name	Id	User_id	Data
Data type	Int (11)	Int (11)	Text

3.8 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of an Information Network data "flow." A data flow model can also be used for simulation of the data analysis. A designer's common practice is to draw a DFD context-level first which shows the relationship between the system and external entities. Instead the context-level DFD is "exploded" to display in more depth the system being based on. A DFD stands for streaming data through a system. Information flow diagrams are commonly used during troubleshooting. It regards a machine as a

process which transforms the input into the desired output. A DFD shows data flows within the device across various transformations or processes.

Dataflow diagrams can be used to give the end user a concrete understanding of where the data they input actually influences the layout of the whole system and recreate how a dataflow diagram can be used and decide how every system is created. The relevant log was deposited in the database and managed by the appropriate authorities.

Data Flow Diagram Notation

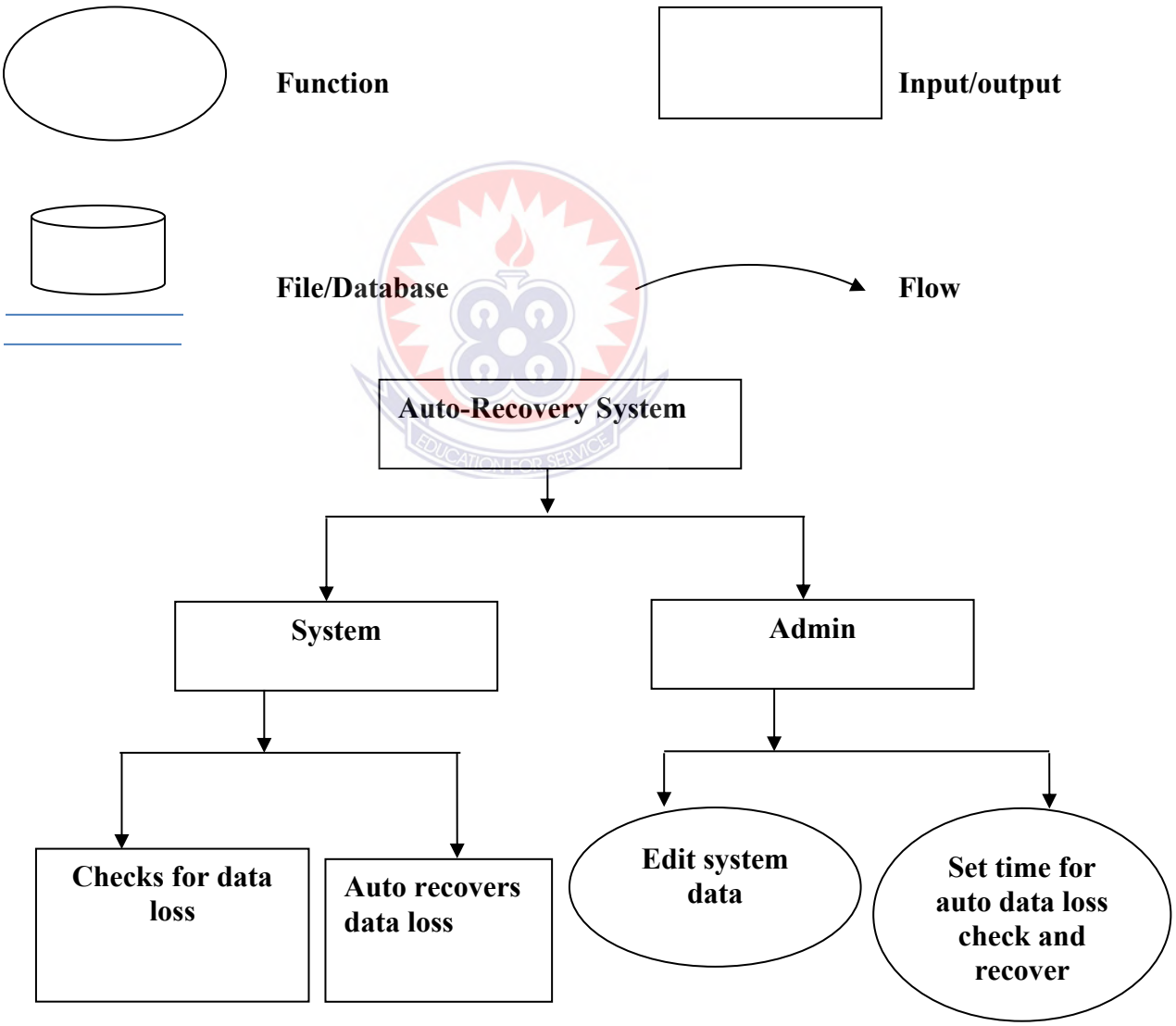


Figure 4: Data Flow Diagram

3.9 IMPLEMENTATION

During the deployment stage of The Automatic Data Loss Recovery System, the following procedure was followed.

1. Drawing of Needs Analysis/Strategy Plans
2. The use of Pilot Implementation Program
3. The Pilot Implementation Review
4. Finishing the Pilot Project
5. Monitor and acceptance process for the systems.

1. Needs Assessment.

When carrying out the Needs Assessment, the project teams confirm the enterprise' commitment to best appropriate practice. This was done to come up with evidence that there was really the need to implement the proposed system. This was done to confirm the priorities for the various data required to be collected about the data security and recovery. Once these decisions have been made, a detailed implementation program was then sure to be developed. It was then concluded that the sophistication of the online automated data loss recovery system will based on the numerical strength of the organization's data. Therefore the system had to be built on applications for project management to monitor the:

- Automated backup system
- Self-check system for data loss
- Auto healing system
- Outputs
- Search

2. Application of the Pilot Plan to Deploy.

This level includes the deployment of a detailed web automated data loss recovery system implementation plan to evaluate the online backup system on a small scale before the recovery system is implemented widely for all targeted enterprise data. The pilot stage covered:

a) Confirm data capture requirements and commence data storage

b) Information for pilot plan attributes to be obtained (confirm priority levels)

c) Build comprehensive pilot activity plan including:

d) Activities required

e) Required time allotted

f) Data collection and inputting

g) Quality assurance

h) Training

i) Product testing and handover

j) Acceptance of a contract signed

k) Track and study the contract-wide collection of data

3. Pilot Program Review

By reviewing the program at a 30% to 50% complete stage, we have the opportunity to:

- Assess the key choices taken during the implementation stage of the requirements analysis/strategy program,
- The outcomes obtained match the company's goals

The computing systems selected fulfill the planned purpose and assess the capacities of the internal personnel to better identify the data collection processes involved and the commitment or inputs needed

- Determine the most fitting way to complete the project based on the Pilot Plan evaluation
- Evaluation the overall scheme based on the review and agree on the rules of the online shop network to be implemented for plan consistency.

4. Complete the Pilot Program

Once the pilot plan was finalized, the preliminary final electronic data loss recovery framework pilot report had been further reviewed. If the pilot scheme evaluations were completed, a post-pilot brainstorming test meeting was held to discuss the details of online program malfunction, and how the participants in the pilot test should conduct the same analysis or otherwise depending on the acquired information. After evaluating the modifications to the software, the organization instead has to collect the system's required functionalities.

3.10 Specification of the System

A System Requirement Specification (SRS), a system requirement specification, is a description of a system's functionality to be created, which can provide a series of use cases that define the users' encounters with the program. Additionally it also contains non-functional specifications. Non-functional specifications enforce specification or implementation restrictions, such as criteria for performance engineering, quality standards, or architecture restrictions (Wikipedia 2015). All requirements are listed in brief below;

3.10.1 User Characteristics Specifications

There are basically two categories of actors for the Automated Data Loss Recovery System used with the PHP-MySQL management system: the system itself which performs some automated functions and the administrators who monitor the backend processes.

3.10.2 Constraints Specifications

The existing limitations on the project are due to the availability of hardware resources to develop and validate a high-performance Window-Based Operating System. At this stage, a Pentium- IV or Pentium V, with 1024 Mb RAM, serves as the minimum hardware required to run this application, with all the computers locally networked. For better performance analysis, a larger number of dedicated processor-based (Core Duo, i3, i5, i7) computer system would be beneficial, specifically a dedicated server system. For testing purposes, a simulating 64bit or 32bit server-based tool such as XAMPP, WAMPP and if on a Linux platform, LAMPP, needs to be used which may have to be installed and launched before viewing the contents of the PHP files or manipulating the MySQL database attached to this project.

3.10.3 System Specification

Components	Requirement
Storage requirements	• PC: 3.0 GB of available disk space Mac: 2.5 GB HFS+ hard disk format
System memory	• PC: 4 GB RAM (32-bit); 4 GB RAM (64-bit). Mac: 4 GB RAM
Input devices needed	• All features and functionality are always available by using a keyboard and mouse as accessible input device
Output devices to be used	• Printers, Display Monitors.

Computing/ processing power needed	• PC: 2.5 gigahertz (GHz) or faster x86-bit or x64-bit processor • Mac: Intel processor
Security and Backup systems	• How the passwords are handled and controlled, authentication methods, how backups are to be handled and so on.
Network configuration	• Local Area Network / Wide Area Network / Internet
System feeds	• Inter-linked with Microsoft Office Excel, staff and grade management system to keep track full student records.
Operating System	• PC: Windows XP, Windows Vista, Windows 7, Windows 8. Mac: Mac OS X 10.6 or later

3.10.4 Software Specification

Components	Requirement
Software applications to be used	XAMPP 5.1 or WAMPP 4.01 or LAMPP (Linux Kernel) SMSCaster E-Marketer GSM Enterprise 3.6 Plugin
Browsers	Best working in Google Chrome 15.0 or above Supporting browsers include: Opera Mini 15 or above Mozilla Firefox 15 or above Internet Explorer 5.0 or above
Export – Imports	Excel imports Comma Separated Value (.csv) files System Exports Comma Separated Value (.csv) file containing customer name and email as fields

3.11 Recommended System Requirements for the Proposed System

The researcher must include a set of specifications that are to run the program to the administrator. Those specifications are almost often significantly higher than the minimum specifications, which reflect the optimal condition for running the program. In general words, the following is a great checklist for any business that needs to operate the automated data loss recovery System:

3.11.1 Hardware Requirement

Physical computing resources, also called hardware, are the most basic set of specifications specified by any operating system or software program.

Architecture and Processing Power

Both computer operating systems are customized to specific computer design. Many computer programs are restricted to the same operating systems that operate on different architectures. A basic device specification for any program is the power of the central processing unit (CPU). The bulk of applications runs on x86 architecture describes computing power as the CPU's configuration and clock speed. The computer should at least have Intel Pentium IV 833MHz or above.

Memory

When used, all information remains in a computer's Random Access Memory (RAM). Memory specifications are specified after consideration of program demands, operating system, disk and

files support, and other running processes. To run this program the minimum RAM needed is 4 Gigabyte.

Secondary Storage

Specifications for hard drives differ on the basis of scale of the device installation, temporary files generated and managed even when the program is installing or running, and the use of swap space (if RAM is not enough). The minimum storage space available for running this program is 80 Gigabytes

Display Adapter or Video Card

Software needing a higher than average computer graphics monitor, such as graphics editors and high-end games, also describe the device specifications for high-end monitor adapters. But our program indicated includes the minimum device requirement.

3.11.2 Software Requirements

System specifications describe the concept of computing infrastructure criteria and preconditions that need to be implemented on a device to ensure that program works optimally. These specifications are usually not included in the installation plan of the program and must be configured separately before running the program.

Operating System

A network of program users defines any kind of computing system that enables the machine to operate. Typical systems provide a machine's processor, or programming languages on runtime, and their libraries. Operating framework is one of the criteria defined when setting device requirements. Software can't fit various implementations of the same set of operating systems, although certain backward compatibility metrics are also retained With this, our Proposed System, Automated data loss recovery System will only run effectively Windows 8.1 (6 -bit and 32-bit), Windows 8 (64-bit and 32-bit), Windows Vista, Windows XP.

Web browser

Most internet apps and devices, which are highly reliant on Web technology, use the default browser built on the device. Microsoft Internet Explorer is a popular choice of software running on Microsoft Windows and, given its vulnerabilities, uses ActiveX controls. Apart from this, the proposed software application can run only of the following computer browser; Mozilla Firefox, Netscape, Google Chrome and Opera Mini.

Other Specifications

Most technology has other efficiency criteria too. The proposed system should run on a network. The resolutions of the display screen are not less than 1366 x 768 (dpi).

3.12 Design of the Proposed System

Systems design is the process of specifying the system's design, elements, modules, interfaces, and data to satisfy defined needs. Systems architecture may be seen as applying system theory to

the creation of goods. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

3.12.1 Physical Design of the Proposed System

The formal architecture refers to the system's real input and output systems. That is laid out in terms of how data is put into a program, how it is checked, how it is interpreted and how it is displayed. In basic design; the following system specifications are determined; the planned system's physical component of system design will usually be broken down into two sub tasks:

• Data Design Interface

• User Design Interface

Data Design Interface

Data Architecture is about how the data is interpreted and preserved within the system. The proposed system is a web-based application system designed in simple modules with its caption as “Auto - recovery”. The system has got some main pages for data entry and other pages for display of information.

3.13 Requirement Specification

Requirements of systems can be Functional or Non-functional.

3.13.1 Functional Requirements

Functional standards specifies the collection of inputs, actions and outputs, measurements, technical information, storage and analysis of data, and other basic features that determine what a program is meant to do. R. T. Yeh (1982). In relation to this thesis, The Functional Criteria

Specification defines the processes and procedures to be carried out by an Automatic Data Loss Recovery Program. The core Device Functional Specifications include:

1. Entering and saving of enterprise data details in the system
2. Self-system backups
3. System auto heals from system failure
4. Self-system check
5. Update existing enterprise information
6. Delete enterprise data from a database

3.13.2 Non-Functional Requirements

The non-functional specifications define additional device property, rather than functionality, to be designed. A non-functional requirement defines the required reliability and availability of the system. Non-functional specifications can be grouped into categories such as user configuration criteria, specification requirements, functionality limitations, abstract database specifications and 'characteristics' that do not fall easily within any of the other categories.

From this framework, security criteria are essential considerations because confidential data would be contained in the database. User authentication should be performed during registration to ensure that the account is correct and that only his or her authorization data is available to the system. General users can only have access through the User Interface. The program should have standardized configuration templates and button sets for all the client modes, will have a form-based configuration for both data entry and presentation types, and will produce files that are formatted into a table that will appear like the current user-friendly manual report templates. The

program is quickly controlled by the creator or other qualified designated individual and is extremely sensitive to user request.

CHAPTER FOUR

THE RESULTS OF THE ANALYSIS OF THE ALGORITHMS

4.1 Introduction

This thesis aims to create a system that will represent auto recovering data loss due to system failure, power outages, flooding or intentional/unintentional deletes. With this system, Web-based companies can increase their services' efficiency. Automated data loss recovery system is one of the solutions for enhancing the protection of vital information to businesses. The system contains many of the functionality of an integrated online recovery system.

This chapter elaborates the testing of the system, I have described the performance of recovery techniques as deployed in the existing system and against the techniques deploy in the proposed system. Finally Requirement and Implementation analysis were also considered in this part of the chapter.

Recovery algorithms are techniques to ensure transaction atomicity and durability despite failures. The recovery subsystem, using recovery algorithm, ensures atomicity by undoing the actions of transactions that do not commit and durability by making sure that all actions of committed transactions survive even if failures occur.

The two main approaches in recovery process are:

I. Log-based recovery using WAL protocol and
II. Shadow-paging.

According to Joost Verhofstad, he presented the “seven techniques to recover the data in the Database as follows:

Salvation program: Run after a crash to attempt to restore the system to a valid state. No recovery data used. Used when all other techniques fail or were not used. Good for cases where buffers were lost in a crash and one wants to reconstruct what was lost.

Incremental dumping: Modified files copied to archive after job completed or at intervals.

Audit trail: Sequences of actions on files are recorded. Optimal for "backing out" of transactions. (Ideal if trail is written out before changes).

Differential files: Separate file is maintained to keep track of changes, periodically merged with the main file.

Backup/current version: Present files from the current version of the database. Files containing previous values form a consistent backup version.

Multiple copies: Multiple active copies of each file are maintained during normal operation of the database. In cases of failure, comparison between the versions can be used to find a consistent version.

Careful replacement: Nothing is updated in place, with the original only being after operation is complete.

4.2 Performance of Recovery Techniques

This section describes the performance of the recovery techniques as use to recover data in cloud computing. Here the focus is on the algorithms deploy in the existing system and the algorithm suggested in the propose system. The algorithms are Log Based Recovery Techniques as in the propose system and the Shadow Paging Recovery Techniques as used in the existing system.

4.2.1 Log Based Recovery Techniques

Logs are the sequel of records which are used to manage records of the actions during a transaction. Logs are written before the actual alteration and stored on a stable storage media. Log Based Recovery technique works in three different manners:

i. Deferred Update,

ii. Immediate Update,

iii. Checkpoint

4.2.1.1 Deferred Update Method

In this method a database is not physically updated on disk until after a transaction reaches its commit point. After it, the updates are stored persistently in the log and then written to the database. Before the commit point the transaction updates are managed in the local transaction workspace like buffers. If transaction fails before reaching commit point, it will not have changed the database; hence there is no need to UNDO. So it is necessary to REDO the effect of the operations of a committed transaction from the log, because the effect may not yet have been recorded. Hence it is also termed as “NO UNDO/REDO” Algorithm.

For an example:

Let there be a transaction t1 as follows: read(X), write (10, Y), write (20, Z),

Commit.

Table 7: Using the Deferred Update

TIME	ACTION	LOG
t1	START	-
t2	Read(X)	-
t3	Write(Y, 10)	Y=10
t4	Write(Z, 20)	Z=20
t5	COMMIT	COMMIT

From table 7, whenever any transaction is executed, the updates are not made immediately to the database as shown from the table. They are first recorded on the log file and then those changes are applied once commit is done. Once the rollback is done none of the changes are applied to the database and the changes in the log file are also discarded. If commit is done before crashing of the system, then after restarting of the system the changes that have been recorded in the log file are thus applied to the database.

Table 8: After update

DISK	Before			After		
	-	-	Y=6	-	-	Y=10
	X=5	Z=2	-	X=5	Z=20	-

The idea behind deferred update is to defer or postpone any actual updates to the database itself until the transaction completes its execution successfully and reaches its commit point. During

transaction execution, the updates are recorded only in the log and in the transaction workspace as display in table 8.

4.2.1.2 Immediate Update Method

In this method database may be updated by some operations of a transaction before the transaction compasses its commit point. These operations are consistently recorded in the log on disk effectively writing before adapted. If a transaction aborts after keeping record some changes to the database, but before commit point, the effect of its operations on the database must be undone. This method needs both undo and redo in recovery. Hence it is also termed as "UNDO/REDO" Algorithm to the database. Transaction T1 in the above example works as below:

Table 9: Using Immediate Update

TIME	ACTION	LOG
t1	Start	-
t2	Read(X)	-
t3	Write(10, Y)	Y = = 6 now 10
t4	Write(20, Z)	Z = = 2 now 20
t5	COMMIT	COMMIT

Table 10: After Updates

	Before			During			After		
DISK	-	-	Y=6	-	-	Y=10	Y	Y	Y=10
	X = 5	Z = 2	-	X=5	Z=2	-	X=5	Z=20	-

From table 10, whenever any transaction is executed, the updates are made directly to the database and the log file is also maintained which contains both old and new values. Once commit is done, all the changes get stored permanently into the database and records in log file are thus discarded .Once rollback is done the old values get restored in the database and all the changes made to the database are also discarded. If commit is done before crashing of the system, then after restarting of the system the changes are stored permanently in the database.

4.2.1.3 Checkpoint Method

When more than one transaction is being executed in parallel, the logs are interleaved. At the time of recovery, it would become hard for the recovery system to backtrack all logs, and then start recovering. During execution system manages the log, using one of the two techniques Deferred Update or Immediate Update. In addition, System repeatedly performs checkpoints. Checkpoint can be occur automatically and manually constantly whenever the database is shut down or when a redo log switch occurs. We use the following SQL command vigorously to use checkpoint:

i. Alter system switch log-file,

ii. Alter system checkpoint.

4.3 Shadow Paging Techniques

Shadow paging is a method used to attain atomic and durable transactions and provides the competence to manipulate pages in a database. In this method the database is partitioned into fixed-length blocks known as pages. For each database page, a page table with n entries is maintained. Each page contains a pointer to a page on disk. To recover from failure there are two page tables "current page table" and "shadow page table". At the starting of a transaction both tables are identical. During the transaction, shadow page table is never altered. When there is write operation during a transaction, the current page table may be changed. Current page table is used by all input and output operations to locate database pages on disk. Hence this method does not need UNDO/REDO algorithm. It recovers the data faster. But in this method data gets fragmented. If the transaction completes then all pages which contains old version of altered data need to be garbage collected. Because of these reasons this method is hard to extend to allow transaction to run simultaneously.

Figure 5: Nature of System in case of Concurrent Transaction

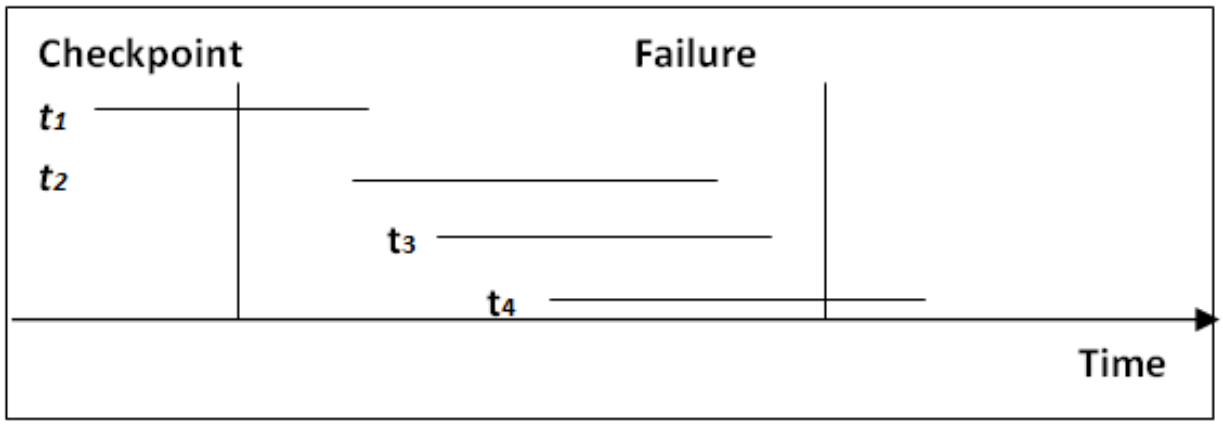
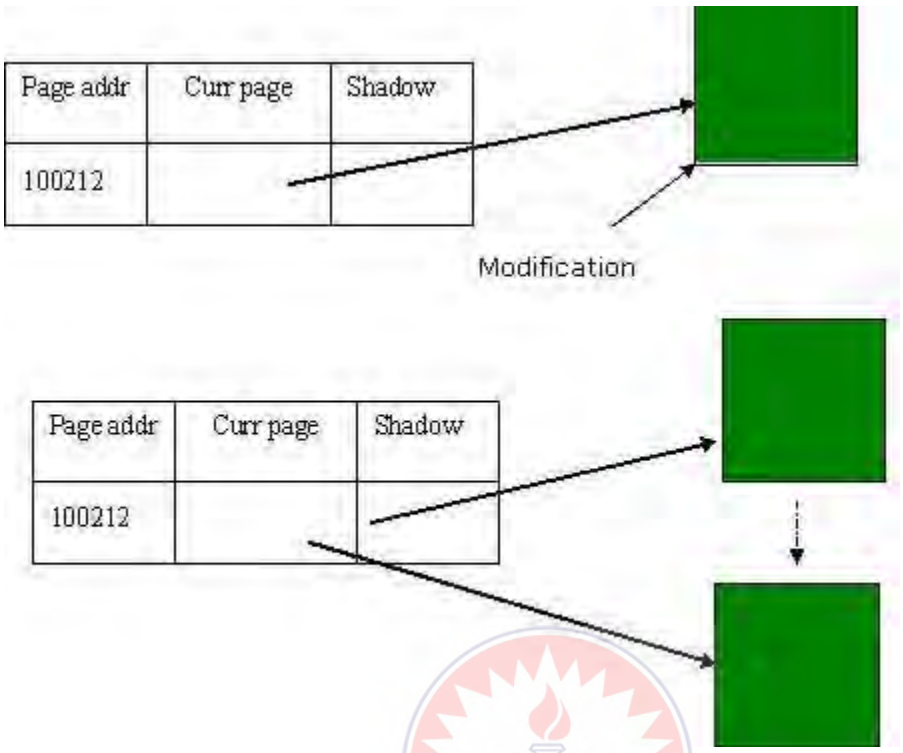
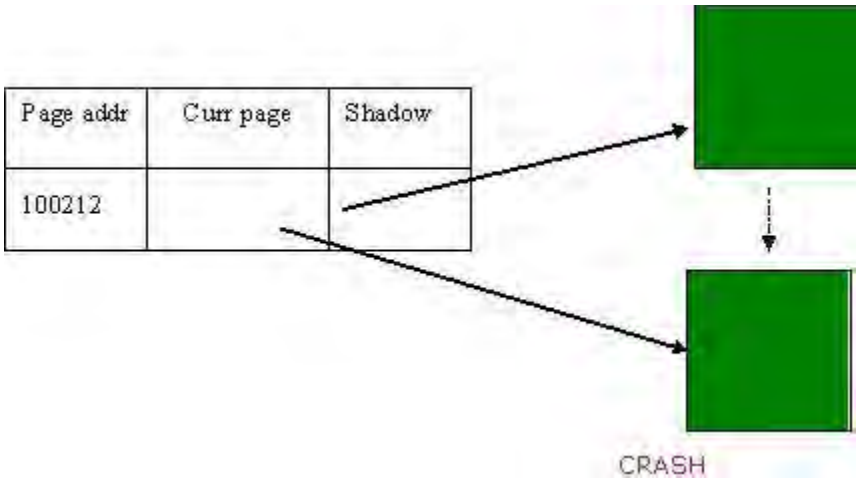


Figure 6: Recovery through shadow paging



The idea is to maintain two page tables during the life of a transaction: the Current page table and the shadow page table. When the transaction starts, both tables are identical. The shadow page is never changed during the life of the transaction. The current page is updated with each **write** operation. Each table entry points to a page on the disk. When the transaction is committed, the shadow page entry becomes a copy of the current page table entry and the disk block with the old data is released. As shown in figure 6

Figure-7: Recovery through shadow paging after crash



From figure 7, if the shadow is stored in nonvolatile memory and a system crash occurs, then the shadow page table is copied to the current page table. This guarantees that the shadow page table will point to the database pages corresponding to the state of the database prior to any transaction that was active at the time of the crash, making aborts automatic.

4.4 Comparison of Recovery Techniques

The table below depicts the comparisons of the techniques deployed in the existing and propose system as used in the data recovery in cloud computing.

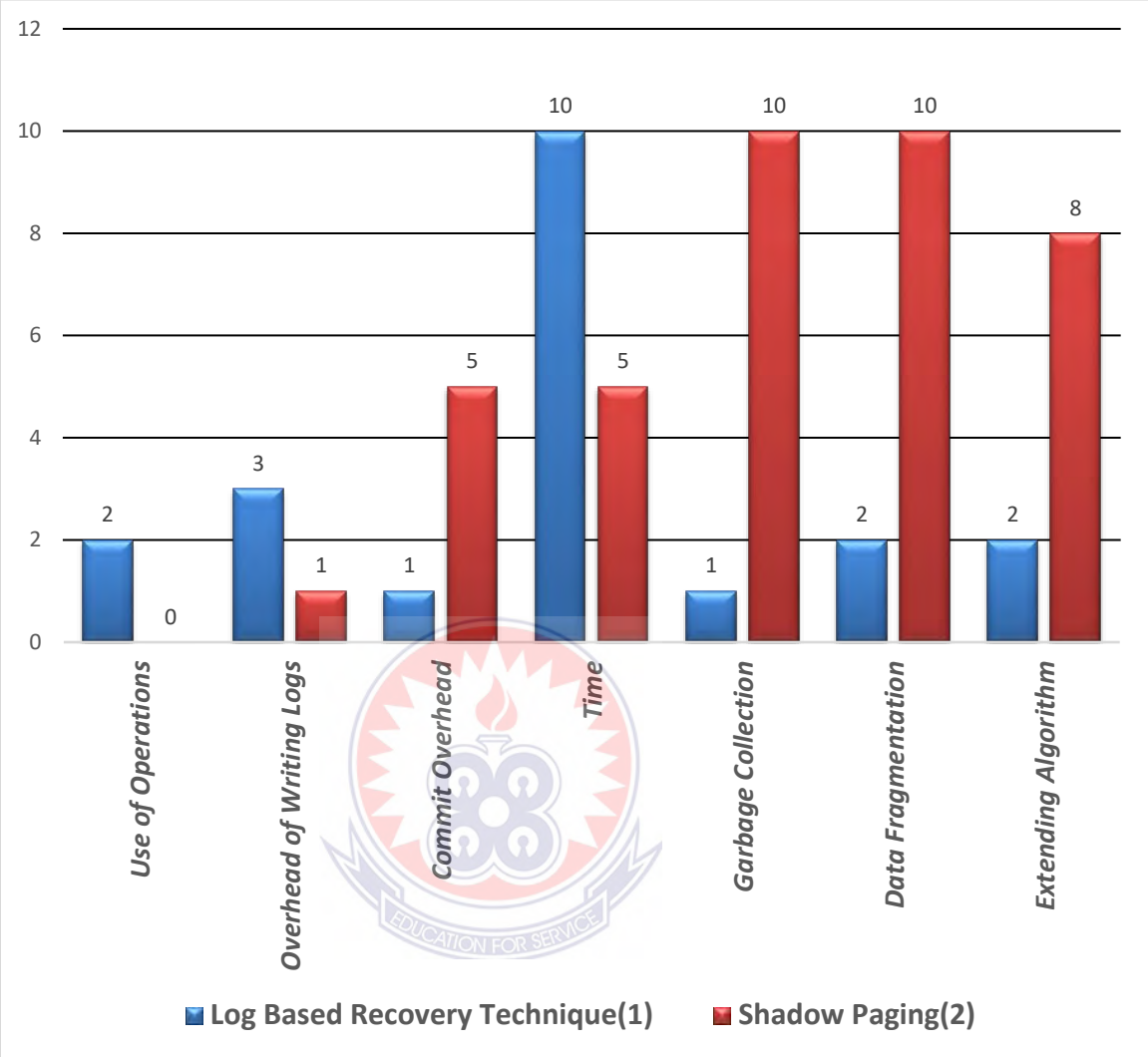
Table 11

Basis of comparison	**Log based Recovery**	**Shadow Paging**
Overhead of writing Logs	Logs required	No Require
Recovery	Not Trivial	Trivial
Data Fragmentation	No Fragmentation	Get Fragmentation
Need of Garbage Collection	No	Yes
Extension of Algorithm	Easier for concurrent transaction	Hard for concurrent transaction
Time in case of System failure	Increase to Recover	Decrease to Recover
Use of Operation	Undo(T) Redo(T)	No Undo(T) No Redo(T)

Table 12: Statistical Representation of the Recovery Technique

Factors	**Log Based Technique(T1)**	**Shadow Paging(T2)**
Use of Operation	2	0
Overhead of writing logs	3	1
Commit Overhead	1	5
Time	10	5
Garbage Collection	1	10
Data Fragmentation	2	10
Extending Algorithm	2	8

Figure 8: Graphical Representation of the Recovery Techniques



From the figure-8, it is clear that Log Based Recovery Technique use two operation (UNDO and REDO) but Shadow paging does not use them. In each transaction Log Based Technique uses Logs but Shadow Paging does not use which increases its performance. But Shadow Paging needs to be garbage collection of each new modified data in old table and also data gets fragmented which are the drawbacks of Shadow Paging. Shadow paging algorithm is also hard to extend to allow transaction to run concurrently but here, Log Based scheme is easier to extend.

Based on the analysis of the graphical representation as shown in figure 8, I vividly recommend that Log Based Recovery Technique should be deploy to recover lost data faster than the Shadow Paging Recovery Technique.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This thesis seeks to investigate the Disaster recovery in cloud computing using appropriate recovery mechanism to ensure automated self-healing process. Through implementing this program, organisations are able to make their programs more effective. Automated data loss recovery program is one of the solutions for enhancing the protection of sensitive information to businesses. A brief summary, achievements, conclusions and recommendation of the projects were discussed in this chapter.

5.2 Summary

Present disaster recovery techniques typically require manual action to restore the damaged system, which cannot avoid interruption of operation because it may take minutes following failure to run backup network. This thesis suggested a strategy for disaster recovery to assist in this situation, seeking to accomplish a stable, streamlined, adaptable and efficient recovery mechanism on a cloud-based network. Two significant goals were expense of program implementation and efficiency of the program. The system is compatible with any modern Web server, and no external hosting services are needed. The Log Based Recovery Algorithm is flexible enough to accommodate the deviations.

5.3 Conclusion

This thesis provided a paradigm for the creation of automatic data loss recovery system which can be used to model, evaluate and improve the data security of today's enterprises.

We have analyzed that in some cases Shadow Paging can be used to recover easily but it is restricted to use only in case of single user environment.

In case of multi-user environment, Shadow paging uses Logs for concurrency Control method. Hence we can say that Log Based Recovery should be used to recover data automatically after disaster in cloud computing. It also needs to be modified to perform trivial recovery as in Shadow Paging to decrease the time of execution of the process of Log Based Recovery Technique.

So in future our aim is to develop a new technique that solves this problem efficiently.

As an evolving platform for business connectivity, online platforms have the ability to transform the manner in which businesses exchange information and deliver services. Increased product knowledge sharing can offer benefits for both producers and consumers as suppliers can be more educated and buyers can expand their market presence. The aim of this thesis is to develop a cloud base automated recovery system to improve data recovery and data security.

5.4 Recommendations

From the figure-7 it is clear that Log Based Recovery Technique use two operation (UNDO and REDO) but Shadow paging does not use them. In each transaction Log Based Technique uses Logs but Shadow Paging does not use which increases its performance. But Shadow Paging needs to be garbage collection of each new modified data in old table and also data gets fragmented which are the drawbacks of Shadow Paging. Shadow paging algorithm is also hard to extend to allow transaction to run concurrently but here, Log Based scheme is easier to extend. Hence we recommend that Log Based Technique should be extending to recover fast after data loss.

REFERENCES

Akanshika (2013). Analysis of Rollback Recovery Techniques in Distributed Database Management System", International Journal of Modern Engineering Research *(IJMER)*, 3(3), 1353-1356 ISSN: 22496645.

Alavi, M. (1985). An Assessment of the Prototyping Approach to Systems, Communications of the ACM, 27, 6, 556-564.xv

Amit, R. & Zott, C. (2001). "Value creation in e-business", Strategic Management Journal, Vol. 22 Nos 6/7, pp. 493-520.

Anastasi H. (1961). Software Engineering a Practitioner's Approach Review III, 2008, pp.130-154.

Archer, N. & Yuan, Y. (2000). "Managing business-to-business relationships throughout E-commerce procurement life cycle", Internet Research, Vol. 10 No. 5, pp. 385-95.

Ashish Srivastava, Udai Shankar and Sanjay Kumar Tiwari (2012).Transaction Management in Homogenous Distributed Real-Time Replicated Database Systems. International Journal of Advanced Research in Computer Science and Software Engineering. 2(6), ISSN: 2277 128X.

Basili, V.R. & Turner, A.J. (1975). Iterative Enhancement: A Practical Technique for Software Development, IEEE Transactions on Software Engineering 1, 4, 390-396

Becta B. (2001). Promotion and integration of ICT in education, Millburn Hill Road, Canley, Coventry, UK in the University of Warwick Science Park

Bentley R. v. Derek B. (1998). Networked Knowledge - Placement Report, CCA 2516

Boddy, D., Macbeth, D.K., Charles, M. & Fraser-Kraus, H. (1998). "Success and failure in Implementing partnering", European Journal of Purchasing & Supply

Boehm, B.W., Gray, T.E. and Seewaldt, T. (1992) Prototyping versus Specifying: A Multiproject Experiment, IEEE Transactions on Software Engineering, SE-10, 4, 290 -402.

Budde, R., Kautz, K., Kuhlenkamp K., & Zullighoven, H. (1992). What is Prototyping?" Information Technology & People, 6, 2-4, 89-95.

Croom, S. (2005). "The impact of e-business on supply chain management: an empirical study of Key developments", International Journal of Operations Production Management, Vol. 25, No. 1, pp. 55-73.

Demand Media Inc. (1999-2014). HR-D Systems. Retrieved from; http://www.ehow.com/facts_6918359_human_realation_system.html)

David G. & Douglas R. (1991). Guidelines towards European standard for quality assurance of student placement.

Doherty, N.F. & Ellis-Chadwick, F.E. (2006). "New perspectives in internet retailing: a review and strategic critique of the field", International Journal of Retail & Distribution Management, Vol. 34 Nos 3/4, pp. 411-28.

Floyd, C. (1984). A Systematic Look at Prototyping, in: Budde, R., Kuhlenkamp, K., Mathiassen, L. and Zullighoven, H. (Eds.) Approaches to Prototyping, Springer-Verlag: Heidelberg, 1-17.

Frohlich, M.T. & Westbrook, R. (2001). "Arcs of integration: an empirical study of supply Chain strategies", Journal of Operations Management, Vol. 19 No. 2, pp. 185-200.

Gomaa, H. (1983). The Impact of Rapid Prototyping on Specifying User Requirements, ACM SIGSOFT Software Engineering Notes 8, 2, 17-28.

Gordon, V.S. & Bieman, J.M. (1994). Rapid Prototyping: Lessons Taught, IEEE Software, 12, 1, 85-95.

Gray, W. John, B. & Atwood, M. (1992). The precis of project Ernestine or An overview of a validation of GOMS. Proc. of the Annual ACM SIGCHI Conference: CHI '92, New York: ACM Press 307-312.

Guihe Wang Ligang Qu Limin Fan Tianbiao Yu Wanshan Wang, (2009)."Web-based system for industry using information and communication technologies", Kybernetes, Vol. 38 Iss 3/4 pp. 533 – 541.

Harland, C.M., Lamming, R.C. & Cousins, P. (1999). "Developing the concept of supply strategy", International Journal of Operations & Production Management, Vol. 19 No. 7, pp. 650-73.

Hawker, A. (2000). *Security and Controls in Information Systems*, London, Routledge.

Hoyt, J. & Huq, F. (2000). "From arms-length to collaborative relationships in the supply chain: an evolutionary process", International Journal of Physical Distribution & Logistics Management, Vol. 30 No. 9, pp. 750-64.

International Dictionary of Education (1998). Electronic Dictionary of Education, Crawley, W.A: UWA Publishing.

Kankam J. & Onivehu F.A. (2000). Building School System- Placements for the Benefit of Students. Athens 28 September (2000). Inc. Retrieved March 9, 2014, from http://www.eui-net.org/Project_documents/

Kaushaar, J.M. & Shirland, L.E. (1985). A Prototyping Method for Applications Development End Users and Information Systems Specialists, MIS Quarterly, 9, 4, 189 -197.

Kehoe, D.F. & Boughton, N.J. (2001). "New paradigms in planning and control across manufacturing supply chains", Vol. 21 Nos 5/6, pp. 582-93.

Microsoft Encarta. (1999). Encarta world English dictionary. New York: St. Martin's Press.

Nauman, J.D. & Jenkins, M. (1982) Prototyping: The New Paradigm for Systems Development, MIS Quarterly, 6, 3, 29-44.

North T.A, Rojoneo T. & Praya N. (2000). Training and Placement Department of Rajiv Gandhi Institute of Technology.

Passey & Benjamin, (1997). Nominet Trust, students and the use of the placements – current engagement and future needs.

R. T. Yeh (1982). "Requirements Analysis- A Management Perspective," Proc. COMPSAC '82, Nov. 1982, pp. 410–416.

Ramez Elmasri, Shamkant B. Nava the (2015). Fundamentals of Database System (5th Edition). ISBN-13:978-0321369574, ISBN-10: 0321369572.

Senge, Peter (1990). National Educational System of Australia, computerized databases in preference to paper-based systems. Retrieved from: http://www.wbdg.org/nesa/csms.php.

Somekh G. (2001). University-Industry cooperation in the Knowledge based society. Proceedings of the 2nd International EUI-Net Conference on: Teaching and Research Synergy", 4-6 May, Tallinn, pp.7-13.

Tamakloe, A., & Amedahe T. (1996). Student placements in schools: do they pay-off or shift tastes? 26, pp.125-144.

Tate, G. & Verner, J. (1990). Case Study of Risk Management, Incremental Development and Evolutionary Prototyping, Information and Software Technology 42 4, 207-214.

The free encyclopaedia (2004). System Requirements Wikipedia Inc. Retrieved March 10, 2014, from http://en.wikipedia.org/wiki/System_requirement.

Uduak A. Umoh, Enoch O. Nwachukwu, Augustine A. Umoh, & Etebong B. Isong. (2012). a Web-Based Database System: An Industrial Application. Research Journal in Engineering and Applied Sciences 1(3) 203-208.

Verhoef, P., Lemon, K., Parasuraman, A., Roggeveen, A., Tsiros, M. & Schlesinger, L. (2009). "Customer experience creation: determinants, dynamics and management strategies", Journal of Retailing, Vol. 85 No. 1, pp. 31-41.

Visscher, Wild, & Fung (2001). Evaluation of the Centralized Clinical Placement System (CCPS), Center for California Health Workforce Studies, San Francisco: University of California.

http://en.wikipedia.org/wiki/Software_requirements_specification (Retrieved April 26, 2014)

http://en.wikipedia.org/wiki/System_requirement
(Retrieved April 26, 2014)

WIKIPEDIA (2015) Data Recovery. https://en. wikipedia.org/wiki/ Data_recovery.04/07/2015.