University of Nairobi

Real Time Multi Agent Based Fraud Detection tool for Banking institutions

SOMBA STEPHEN KORU

A research report submitted in partial fulfillment for the requirement of the Degree in Master of Science in Information Systems of the University of Nairobi.

APRIL 2014

DECLARATION

The project in this report is my original work and has not been presented for any other university award

Signature ______________________ Date ______________________

Somba Stephen Koru (P56/P/8407/2003)

This research report has been submitted in partial fulfillment for the requirement of the Degree in Master of Science in Information Systems of the University of Nairobi with my approval as the university supervisor

Signature ______________________ Date ______________________

Dr. Elisha Opiyo
Senior Lecturer
School of Computing and Informatics

Table of contents i
Abstract ii
Dedication iii
Acknowledgement iv
Abbreviations v
Definations vi
List of Tables vii
List of Fiqures viii
CHAPTER ONE: INTRODUCTION 1
1.1 Background of the study 1
1.2 Problem Statement 2
1.3 Assumptions and limitations of the research 3
1.4 Objective of the Study 4
1.5 Research questions/objectives/hypotheses 5
1.6 Proposed Solution 6
1.7 Scope of the Study 7
1.8 Significance of Study 8
CHAPTER TWO: LITERATURE REVIEW 9
2.1 Introduction 9
2.2 Multi Agent Concept and Approach 10
2.3 Review and evaluation of Available Anti Fraud Detection systems 11
2.4 Review of the past studies 12
CHAPTER THREE: RESEARCH METHODOLOGY 14
3.1 Data Collection method 14
3.1.2 Source of Data .... **Error! Bookmark not defined.**15
3.1.3 Data collection tools 16
3.2 Data analysis method 17
3.3 Multi Agent Methodology and Design 18
CHAPTER FOUR: ANALYSIS AND DESIGN 20
4.1 SWOT System Analysis 20
4.2 Real Time fraud detection requirements 21
4.3 System Specifications 22
4.3.1 Functional Requirements 23

4.4 Architectural Design 24

4.4.1 Agent system overview 25

4.4.2 Agent message communication 26

4.5 Agent Detailed Design 27

4.5.1 Agent Based System Overview Diagram 28

4.6 Agent Internal Process 29

4.6.1 Event Descriptors 30

4.6.2 Agents Plan Descriptors 31

4.6.3 Algorithms 32

4.6.4 Database Design 33

CHAPTER FIVE: SYSTEM IMPLEMENTATION AND RESULTS………………………….....34

5.1 Implementation of the System 34

5.2 System Testing 35

5.3 Disscusion of Results 36

5.3.1 Challenges Facing the system 37

5.3.2 system Results 38

CHAPTER SIX: CONCLUSION AND FURURE WORK ………………...……………………40

6.1 Achievements 40

6.2 Research Contributions 41

6.3 Future Work 42

6.4 Assumptions and limitations 43

ABSTRACT

To design and implement a proactive real time fraud detection system tool for banks, it is key to consider the use of a multi agents system in order to achieve this. In this paper after having a chance to meet several system users, bank auditors and experts in charge of diverse risk management activities inside several commercial and community banks, I will introduce a multi agent based system which integrates the knowledge and the opinions collected during the meetings, to designed a prototype of pure reactive agents intended to detect fraud using rules set to determine suspicious activities and transactions in the banking system. The prototype is intended for the management of fraud detection situations where system users and auditors are collaborating according to a three-phase detection process. In the first phase users and auditors, respectively, an effective environment to explicit their knowledge, select the most likely fraud attack components. In the second phase, the components are structured into rules that act as thresholds used for monitoring each account that is transacted into in the core system. In the third phase an alarm is raised for each threshold reached .The success to a solution of this kind depends to a large extent on a proper definition of rules that determine a suspicious event.

DEDICATION

I dedicated this Masters of Research Project to my wife Mary Lyambiko and daughter Serah wanjuki Koru ,who supported me unconditionally throughout the course of my studies , to my lifelong friend the late Moshe Ole Keiwa who encouraged me in time of doubt and despair , to my siblings Benjamin ,John and Cecilia who assured me they will follow my footsteps once am through with the masters degree course and finally to my parents Mr and Mrs Kivalya Somba, who have made innumerable sacrifices so that my siblings and I are able to have an opportunity to pursue our dreams.

Only GOD can truly know the depth of my gratitude to all.

ACKNOWLEDGEMENT

A Master of Science research process requires a lot of effort and support from other people. First and foremost I would like to thank my supervisor, Dr. Elisha Opiyo. I would like to express my deepest gratitude to my lecturers to mention but a few Dr. Elisha Opiyo, Prof Peter Waiganjo, Dr Christopher Moturi, and Prof Odongo William Okelo.I would like to thank all my colleagues at the school of informatics university of Nairobi for their support and cooperation towards my project. Finally I thank my family and the Almighty God for all the opportunities, strength and support offered in the course of the research process

LIST OF ACRONYMS

ATM: Automated Teller Machine.

DTMs: Deposit taking micro-finance institutions

RTGS: Real time Gross Settlement

SACCO: Savings and Credit Cooperative Organization

ICT: Information and Communication Technology

MFI: Microfinance Institution

STP: Straight through Process

MAS: Multi Agent System

FIPA: Foundation for Intelligent Physical Agents

JADE: Java Agent Development Environment

DEFINITION OF THE THEORETICAL TERMS USED

The following terms are operational used and defined in respect of this study:

Fraud: In law, a person is guilty of fraud if they are in breach of any of the following:

- False representation
- Failing to disclose information
- Abuse of position

Fraud occurs where a person unlawfully obtains money or other property belonging to another person or organization by knowingly giving false information or omitting to declare information. It may include stealing, forgery and falsification of records.

Financial Sector: Investopedia(2013) defines 'Financial sector' as

A category of stocks containing firms that provides financial services to commercial and retail customers. This sector includes banks, investment funds, insurance companies and real estate.

Financial Institutions: wikipidia(2013) defines a financial institution as an institution that provides financial services for its clients or members. Probably the most important financial service provided by financial institutions is acting as financial intermediaries.

Commercial banks: Investopedia(2013) defines commercial bank as a financial institution that provides services, such as accepting deposits, giving business loans and auto loans, mortgage lending, and basic investment products like savings accounts and certificates of deposit. The traditional commercial bank is a brick and mortar institution with tellers, safe deposit boxes, vaults and ATMs. However, some commercial banks do not have any physical branches and require consumers to complete all transactions by phone or Internet. In exchange, they generally pay higher interest rates on investments and deposits, and charge lower fees.

Microfinance Institutions: Omino (2005) define MFIs as organizations that
Provide savings and/or credit facilities to micro and small scale business people. MFIs
Provide financial services to poor people who have experienced difficulties obtaining these
Services from most formal financial institutions because their businesses, saving levels and
Credit needs are all small.

LIST OF FIGURES

Figure 1……………………………………..…………….…….Phases of Prometheus Methodology

Figure 2 ……………………………………………………………….MAS architecture and Design

Figure 3 …………………………………..……………………….……..System Use Case Diagram

Figure 4 ……………………………..…….…………………………...Agents Overview Diagram

Figure 5 ………………………………………………………………...………..…Database Diagram

Figure 6…………………………………………………………………..………. Agent communication

Figure 7………………………………………………………………………..……..….System Log file

List of Tables

Table 1……………………………………………..Agents types, category and responsibility

Table 2……………………………………………………………………….System Results

CHAPTER ONE

INTRODUCTION

If fraud was a virus then everyone would be slightly ill

kroll Global Fraud Report: Annual Edition 2010/11

1.1 Background of the study

Most of the fraud surveys published in the last five years by leading international consulting companies have shown that system user data manipulation play a central role, and therefore the biggest problem to be addressed by auditors and inspectors is to analyze and summarize the information coming from several, sometimes conflicting, sources. Fraud in the financial sector is increasing steadily as technology in these sector advances and most of the cases reported involve data manipulation with the assistance of bank staff that is working with external fraudsters. Banks and other financial institutions have lost billions of dollars to fraudsters every year as a result to such schemes where there is false representation of client information, disclose of client information and abuse of position by institution staff .It is with great concern to note that in the recent past there are more and more cases of fraud in financial institutions where employees are involved. This leaves the institutions vulnerable from the inside as most fear the external attack and forget the internal vulnerability

Fraud in the financial sector is increasing steadily as technology in these sector advances and most of the cases reported involve data manipulation with the assistance of bank staff that is working with external fraudsters. Banks and other financial institutions have lost billions of dollars to fraudsters every year as a result to such schemes where there is false representation of client information, disclose of client information and abuse of position by institution staff

1.2. Problem Statement and Purpose

Fraud is an increasing phenomenon as shown in many surveys carried out by leading international consulting companies in the past years. Despite the evolution of electronic payments and hacking techniques there is still a strong human component in fraud schemes. Conflict of interest in particular is the main contributing factor to the success of internal fraud. In such cases anomaly detection tools are not always the best instruments, since the fraud schemes are based on faking documents in a context dominated by lack of controls, and the perpetrators are those ones who should control possible irregularities. In most cases an employee of the bank would adjust the existing records of an account so that when the transaction is done the details of the account or user seem in order for the transaction to pass as a normal entry or is forced to go through since the person supervising the transaction is aware of the fraudulent entry and is part of the crime which can be termed as abuse of position.

In the banking sector audit team experts can count only on their experience, whistle blowing and the reports sent by their inspectors (Alessandro Buoni, 2012) It is with great concern to note that in the recent past there are more and more cases of fraud in financial institutions where employees are involved. This leaves the institutions vulnerable from the inside as most fear the external attack and forget the internal vulnerability

1.3 Assumptions and limitations of the research

Unfortunately most institutions prefer not to go publicly and report the incidences as they fear losing business if the fraud cases are known to the public, due to this fact it is hard get direct reports from the institutions so it is hard to come up with a clear indication of how rampant the problem is, but I have tried to cover as many reports as I could from the Central Bank Reports, newspaper articles and a few publications on the issue to prove that the problem exists and is actually growing steadily. The following are a few examples:

1. Kenyan banks were victims of more than half the Sh4.1 billion ($48.3 million) fraud that hit East African banks last year as technology made the crime easier. A single bank in the economic bloc lost Sh2.72 billion ($32.1 million) to account for a third of the total fraud through data manipulation.
2. In 2011 commercial banks agreed to combine efforts in weeding out dishonest employees from the banking industry following shocking incidents of banking fraud. Under the arrangements commercial banks agreed to discuss and share information on fraudulent staffs implicated in financial scams with a view of tracking and blacklisting them. In some incidences, bank tellers or clerks collude with outsiders and even with their supervisors to defraud the banks they worked for.
3. "Of all the risks, reputational risk is the worst. Most banks would rather keep cases of attempted fraud or actual fraud under wraps to avoid the damage disclosure would do to their reputation. Most of these cases involve bank employees,"
4. NAIROBI, Kenya, Aug 6 – Four Standard Chartered Bank employees who were charged with stealing a whooping Sh328.6 million from the bank, have been released on a Sh10 million cash

bail or Sh20 million surety each. The bank is yet to recover the stolen money. This is not the first time Kenyan banks have lost money to theft, which is mostly orchestrated by techno savvy bank employees. Last July a report by Deloitte showed that Kenyan banks lost about Sh3 billion to fraud in the year 2011 alone. In 2012, about Sh1.5 billion was lost with majority of the money stolen from customer accounts.

1.4 Objective of the Study

i) To analyze possible internal attacks in financial institutions

ii) To provide various ways of preventing internal attacks in financial systems

iii) To design and implement a MAS prototype system that can detect or prevent these attacks.

1.5 Research questions/hypotheses

- Can a multi agent based system be used to monitor user posting activities in the system?
- Can agents be used to check any unusual entries in the core banking system, so that in case of any unusual transaction initiated by a user the agents can alert management before the transaction is completed?
- Can agents be able to monitor and analyze data in the historical files in the system and any attempt to change result to an alert the management of such attempts?

1.6 Proposed Solution

A multi agent system can be used to monitor user activities in the core banking system, some of the areas that the agents can check in the system would include, keeping track of those entries that have odd frequency ,for example too many debits in same account in one day or recent days , user printing account statements without customers notification, regular altering of accounts information .etc. the intention here would be that chances are the employees with such odd behavior in the system would eventually be involved in fraud or attempted fraud, currently several financial institutions have bought or come up with in house developed anti-fraud systems ,unfortunately non have been keen to check for internal vulnerability, some banks are implementing measures in an attempt to combat fraud; the following are some of the examples:

- Postbank organized a series of anti-money laundering workshops to train its staff on detecting cases of bank related fraud and on desisting from committing fraud themselves
- Equity Bank signed a deal with Belgian EMV for a card management system that speeds up transaction times in order to curb fraud

- Consolidated Bank has instituted multi-level access points where staff have different access rights in the system, password management and a rigid recruitment process
- Imperial Bank selected Oracle Financial Services Software to drive their quest for improved corporate governance and greater risk management. Of particular importance was the selection of Oracle Mantas Anti Money Laundering Oracle Mantas Fraud products
- Some of the banks are now sending the card and pin separately instead of together while others are allowing customers to set their own pin rather than issuing them one
- The implementation of the Credit Reference Bureau system will hopefully curb bank fraud as banks share information on previous banking records
- The Anti-Money Laundering Act should also assist in the fight against bank crime as all financial institutions are required to submit suspicious transactions and Cash Threshold Reports to the Financial Reporting Centre which will be used to facilitate the consolidation of financial intelligence for analysis

1.7 Scope of The study

Due to the vast number of available types of financial institution and their numerous differences and similarities, it is not at the moment possible to address all financial institutions in general, thus the study will narrow down to a particular type of financial institution which is the bank, since most of the existing types borrow most of their core business idea from the banking sector. Also due to the broad nature of financial fraud the study will only deal with fraud on data that is stored on the core banking system data base .Fraud outside the database e exists but will be beyond the scope of this research

1.8 Significance of Study

The process of searching for the fraud is lengthy due to the amount of data involved and also in most cases auditors unknowingly get the information they need from the involved employees whom deliberately mislead them and waste their time. With a multi agent fraud detection system in place to check for unusual transactions, the work load is distributed among the agents thus a search is faster and also blocking a transaction can be faster since the different agents communicate and carry out the verifications otherwise done manually thus can be able to be done on run time before committing a transaction.

Without an effective system to check against internal attacks, Management of these financial institutions rely on auditors both internal and external to trace the fraud, If they know that one has taken place. The problem is that some fraud can go undetected or by the time they figure out that fraud has occurred it’s either too late and they fraud stars have disappeared or they have had enough time to cover their tracks and the trail goes cold

CHAPTER TWO
LITERATURE REVIEW

2.1 INTRODUCTION

Fraud is an increasing phenomenon as shown in many surveys carried out by leading international consulting companies in the last years. Despite the evolution of electronic payments and hacking techniques there is still a strong human component in fraud schemes. Conflict of interest in particular is the main contributing factor to the success of internal fraud. In such cases anomaly detection tools are not always the best instruments, since the fraud schemes are based on faking documents in a context dominated by lack of controls, and the perpetrators are those ones who should control possible irregularities. In the banking sector audit team experts can count only on their experience, whistle blowing and the reports sent by their inspectors (AlessandroBuoni, 2012)

According to (Leung, Wai Sze, 2010) fraud detection systems must not only contend with the creativity of fraudsters, but should also be acutely aware of when day-to-day processes have changed due to recent innovations or technological advancements in the domain. Existing fraud detection methodologies may therefore need to be updated frequently in order to remain sufficiently informed of current developments. An agent-based fraud detection system can be developed were number of multi-agent systems, each incorporated to add a particular aspect of the criminal justice process in investigating incidences of potential crime. By having agents emulate the various tasks that are involved in dealing with a crime, it is anticipated that the resulting fraud detection system will be able to achieve similar successes from applying the same procedure. In order to successfully develop the fraud detection model

One method for detecting fraud is to check for suspicious changes in user behavior. Through automatic design of user profiling methods for the purpose of fraud detection, using a series of data mining techniques. Specifically, using a rule-learning program to uncover indicators of fraudulent behavior from a large database of customer transactions. Then the indicators are used to create a set of monitors, which profile legitimate customer behavior and indicate anomalies. Finally, the outputs of the monitors are used as features in a system that learns to combine evidence to generate high-confidence alarms (Fawcett, T., & Provost, F. (1997).)

Criminal elements in today's technology-driven society are using every means available at their disposal to launder the proceeds from their illegal activities. While many anti-money laundering solutions have been in place for some time within the financial community, they cannot adapt to the ever-changing risk and methods in relation to money laundering. In order for a more adaptive, intelligent and flexible solution for anti-money laundering, the intelligent agent technology is applied in this research. Intelligent agents with their properties of autonomy, reactivity and pro activity are well suited for money laundering prevention controls. Several types of agents are proposed and a novel and open multi-agent architecture is presented for anti-money laundering (Gao, S., Xu, D., Wang, H., & Wang, Y. (2006))

2.1.1 Agent Based Platform

The concept of agent Based anti-fraud system involves software in which agents are developed and designed to achieve specific goals. Several agents development platforms exist including, Agent Builder (Acronymic, inc, 2006), Aglet (IBM Research, 2002) , JADE (Telecom Italia Group, 2007) JACK (Agent Oriented Software pty.Ltd, 2006) and FIPA for agent communication .JADE has been developed by the Telecom Italia Lab and the Agent and the Agent and Object Technology Lab at the University of parma and is selected based on two criteria. JADE is well-proven and scalable and it provides complete control to the agent framework (Bellifemine et al., 2007) JADE also simplifies the implementation of multi agent system through a middle-ware that compiles with the Foundation for intelligent Physical Agents (FIPA) specifications and through a set of graphical tool that support the debugging and deployment phases

2.1.2 Multi Agent Concept

The emergence of multi agent technology is radically transforming software development, design and implementation. The agent based system for real time anti-fraud system is considered effective due to the multi agent capabilities. The desired optimal solution should be proactive and independent .Our desire is to demonstrate an alert notification to the key system users on any suspicious transactions on run time

2.1.3 Reviewing and Evaluation of Available Anti fraud systems

As Published in the (Management and Service Science (MASS), 2010 International Conference on 24-26 Aug. 2010) fraud represents a serious economical problem, which has been studied in different ways due to the fact that fraudsters are benefiting from the fast development of ICT and are developing their techniques. An example is the multi-agent system, called FIDES, which integrates the computational power of data mining tools and attack trees with experts' judgments negotiated through a Delphi-based system. Two scenarios are described: in the first one FIDES, supported by cause-effect diagrams, is used to classify alarms generated by the system to help the experts to focus on the real dangerous ones; in the second one FIDES is used in a proactive way in order to block or prevent human based frauds. The system combines Think-map, Delphi method and Attack trees and it has been built around audit team experts and their needs. The output of FIDES is an attack tree, a tree-based diagram to “systematically categorize the different ways in which a system can be attacked”. Once the attack tree is built, auditors can choose the path they perceive as more suitable and decide whether or not to start the investigation. The system is meant for use in the future to retrieve old cases in order to match them with new ones and find similarities. The retrieving features of the system will be useful to simplify the risk management phase, since similar countermeasures adopted for past cases might be useful for present ones. Even though FIDES has been built with the banking sector in mind, it can be applied in all those organizations, like insurance companies or public organizations, where anti-fraud activity is based on a central anti-fraud unit and a reporting system.

(John Gavan, 2003) The layered system includes a core infrastructure and a configurable, domain-specific implementation. The detection layer employs one or more detection engines, such as, for example, a rules-basedthreshholding engine and a profiling engine. The detection layer can include an AI-basedpatternrecognition engine for analyzing data records, for detecting new and interesting patterns and for updating the detection engines to insure that the detection engines can detect the new patterns. In one embodiment, the present invention is implemented as a telecommunications frauddetection system. When fraud is detected, the detection layer generates alarms which are sent to the analysis layer. The analysis layer filters and consolidates the alarms to generate fraud cases. The analysis layer preferably generates a probability offraudfor each fraud case. The expert systems layer receives fraud cases and automatically initiates actions for certain fraud cases. The presentation layer also receives fraud cases for presentation to human analysts. The presentation layer permits the human analysts to initiate additional actions. Existing methods of detecting fraud are based primarily on setting predetermined thresholds and then monitoring service records to detect when a threshold has been exceeded. Parameters for such thresholds include total number of calls in a day, number of calls less than one minute in duration, number of calls more than 1 hour in duration, calls to specific telephone numbers, calls to specific countries, calls originating from specific telephone numbers, etc. Many parameters can be used to tailor a particular thresh holding system for certain customers or services.

Pattern recognition is a process whereby event records are analyzed to learn and to identify normal and potentially fraudulent patterns of use in a system. If an interesting pattern is detected, pattern analysis processor determines whether it is a fraudulent or non-fraudulent pattern. To accomplish this, pattern analysis processor uses artificial intelligence technology to train itself in identifying fraudulent patterns. By analyzing volumes of events from history, an AI-basedpattern analysis processor first determines normal patterns and then looks for deviations that can be identified as fraudulent. Processor then detects emerging patterns of such deviations and identifies them as fraudulent patterns.

There are various AI systems available for such a purpose. Examples include tree-based algorithms that obtain discrete outputs, neural networks, and statistical-based algorithms that use iterative numerical processing to estimate parameters. Such systems are widely used forpatternrecognition. By utilizing an AI system forpattern analysis, both normal and fraudulent patterns can be identified from the volumes of data stored in the history database.

The processes of threshold detection, profiling and pattern recognition are described as being performed substantially in parallel primarily to reduce the processing time. The processes can, however, be performed one after another or as some combination of parallel and non-parallel processing.

CHAPTER THREE
RESEARCH METHODOLOGY

The research was conducted in two phases where phase one was Data collection through literature review to show area that banks feel the need to check to detect fraud and phase two involved system development

3.1 Data Collection

3.1.1 Sources of data

Both secondary and primary data was used to get facts on the subject where primary data was data collected from actual institutions and secondary data was collected from literature review that included understanding and observing available fraud detecting systems, other sources included books journals and the internet

3.1.2 Data Collection Tools

Due to the sensitive nature of the study, the methods used for primary data collection where limited the person(s) involved where reluctant to have any written document from them the result where the following three methods

- **Personal Interviews**

This was carried out by interviewing Bank key employees from their personal experience on areas on the core banking system that where prone to misuse by users or area already that had been misused by users. The key employees included Branch managers, internal auditors and credit officers

- **Telephone Interviews**

This was carried out by interviewing Bank key employees from their personal experience on areas on the core banking system that where prone to misuse by users or area already that had been misused by users. The key employees included Branch managers, internal auditors and credit officers

- **Prototype system**

This method proved to be very useful. Even though the bank employees were reluctant to give information on the subject, when provided with a prototype system and asked to contribute on adding on checks that could be put in the system to detect possible fraud they fully collaborated thus most of the data collected was through this method

3.2 Data Analysis Method

The research based its findings though both qualitative and quantitative research methods. During data collection the choice of the methods was constantly modified based on the process of the system development .This methods helped to find and build on the hypothesis of whether Banks require a real time fraud detection system or not and also how many of the suspicious transactions was the system able to detect.

3.3 Multi Agent Methodologies

The proposed design is to have several interacting, intelligent agents pursue some set of goals and/or perform some set of set tasks; The Methodology implemented to support the solution proposal is a closed MAS, where the architecture design is static, with all the agents and functionalities pre-defined. In this closed MAS the agents communicate using a common language, each agent is developed as an expert in his functionality and they all work and cooperate together in order to achieve a main goal.

The coordination of the MAS is cooperative the agents do not compete they cooperate in order to achieve a main objective. The organization is flat each agent is an expert in an area, there is no agent commanding other agents and all agents have the same importance. The communication between the agents is direct there is no agent or middleware between two agents communicating with each other, they communicate directly.

CHAPTER FOUR
ANALYSIS AND DESIGN

4.1 SWOT analysis

The SWOT analysis is one of the most popular tools to analyze strengths and weaknesses of a system and to offer valid suggestions to improve it. In this section a SWOT analysis will be carried out

4.1.1 Strengths

A strong success element of the system is that it has been built around the needs auditors and key system users have collected and expressed; The system was developed according to their suggestions and based on a continuous process of verification moving between the literature and the bank world. The system can be used as an anti-fraud system, but also in a proactive way to prevent future potential fraud; once suspect behaviors are reported to the audit team similar schemes can be detected .A third strength of the system is flexibility, since it can be used in all the contexts where an interaction between external inspectors and a central risk management unit plays a fundamental role.

4.1.2 Weaknesses (or Limitations)

There are some limitations due to its complexity. One difficulty will be in its implementation as different tools with specialized features are problematic to integrate with each other. Another limitation is the validation of the system. Since the frequency of internal fraud is very low and sometimes up to 20years may be needed to collect a few dozen relevant cases, a validation process in the traditional positivisticsense is not possible.

4.1.3 Opportunities

A system like this one would permit a bank to shorten the inspection and reporting system timesaving is a key factor in fraud detection, which allows auditors to develop a timely counter-strategy to block potential fraudulent activities. There are substantial possibilities for improvement in the attack tree building procedure in the system. This can greatly improve the detection process and consequently save time and money; thus the system has a great future potential. Auditors could use it a tool to study old cases stored in the system and create an automatic detection system to help their work

4.1.4 Threats

The decision to adopt the system by banks (or other organizations) could be an issue for the auditors since users are nowadays unwilling to use many different tools to complete a single (even complex) task.

4.2 System Design

The Design has several interacting agents pursuing set of common goals and/or perform a set of preset tasks for each of the agent .Coordination of the MAS is cooperative the agents do not compete they cooperate in order to achieve the main objective. The organization is flat, each agent is an expert in an area, there is no agent commanding other agents and all agents have the same importance. The communication between the agents is direct there is no agent or middleware between two agents communicating with each other, they communicate directly.

The Prometheus Methodology the selected methodology used to implement the proposed solution .This is because of the following reasons:

- The Methodology is Detailed and complete in the sense of covering all activities required in developing MAS
- The methodology makes it possible to develop closed MAS, where the architecture design is static, with all the agents and functionalities pre-defined. In this closed MAS the agents communicate using a common language, each agent is developed as an expert in his functionality and they all work and cooperate together in order to achieve a main goal
- Real time Fraud detection involves monitoring and recoding activities that take place in an institutions core banking system during transactions .Most Fraud detection and prevention systems use historical data that has been already committed to the database and fraud detection is reactive and not proactive. The desire is to have a system that detects suspicious entries at run time that is immediately the entry is posted the account involved is checked against a number of set rules to that define a suspicious account or amount The Design of the system is guided by the Prometheus Methodology .The three phases of Prometheus methodology followed are outlined below.

i) **System specification -** where the focus is on identifying the basic functionality of the system along with inputs (percepts), outputs (Action) and any important shared data.

ii) **Architectural design** – where by the outputs from the previous phase is used by the next agent level in the system and show how they interact

iii) **Detailed design** – Involves looking at the internals of each agent and how it accomplishes its tasks within the overall system

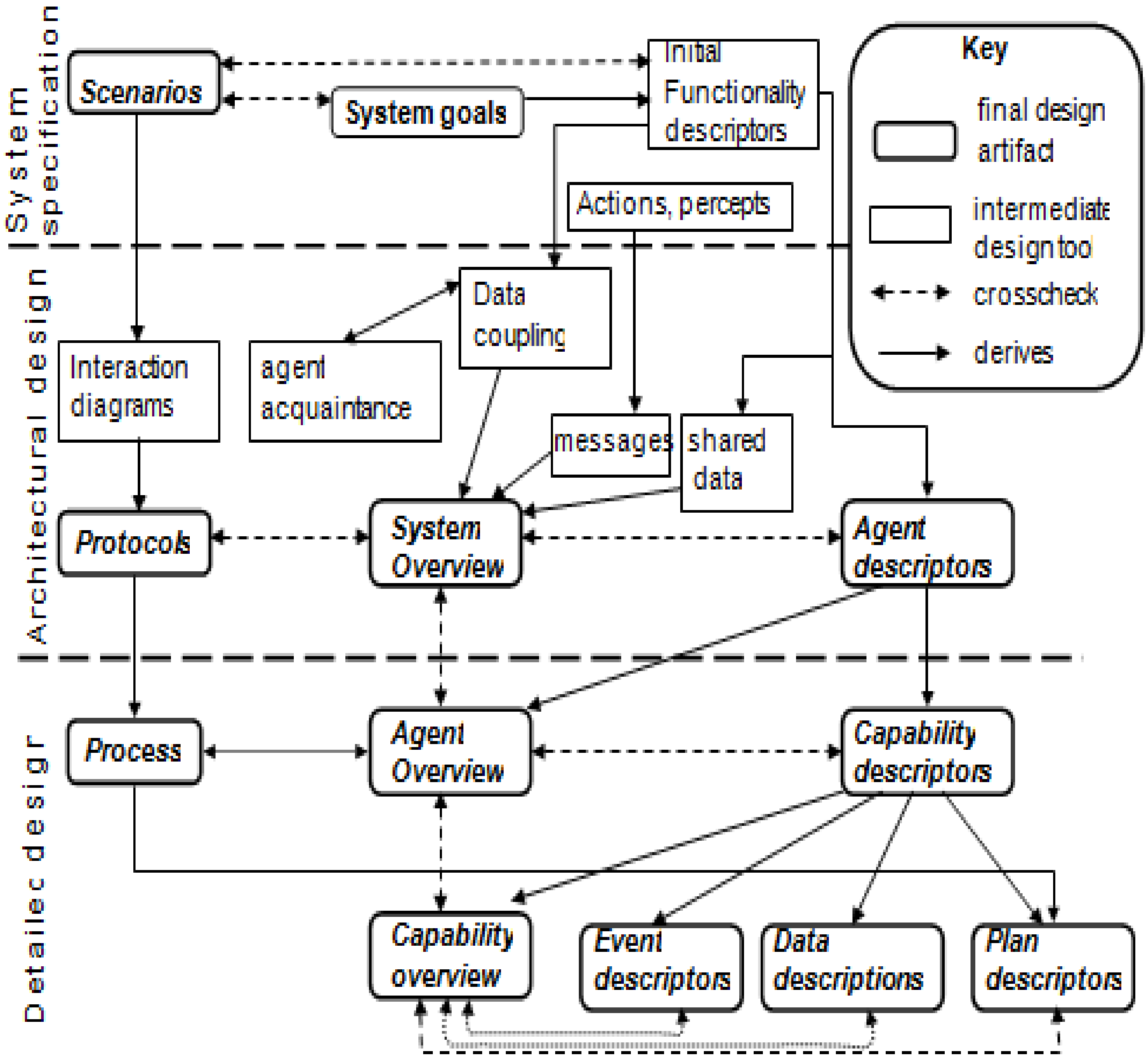


Figure 1: the Phases of Prometheus Methodology

4.3 System specification

System specification begins with a rough idea of the system description and processes in order to define specific requirements of the system in terms of:

4.3.1 Functional Requirements

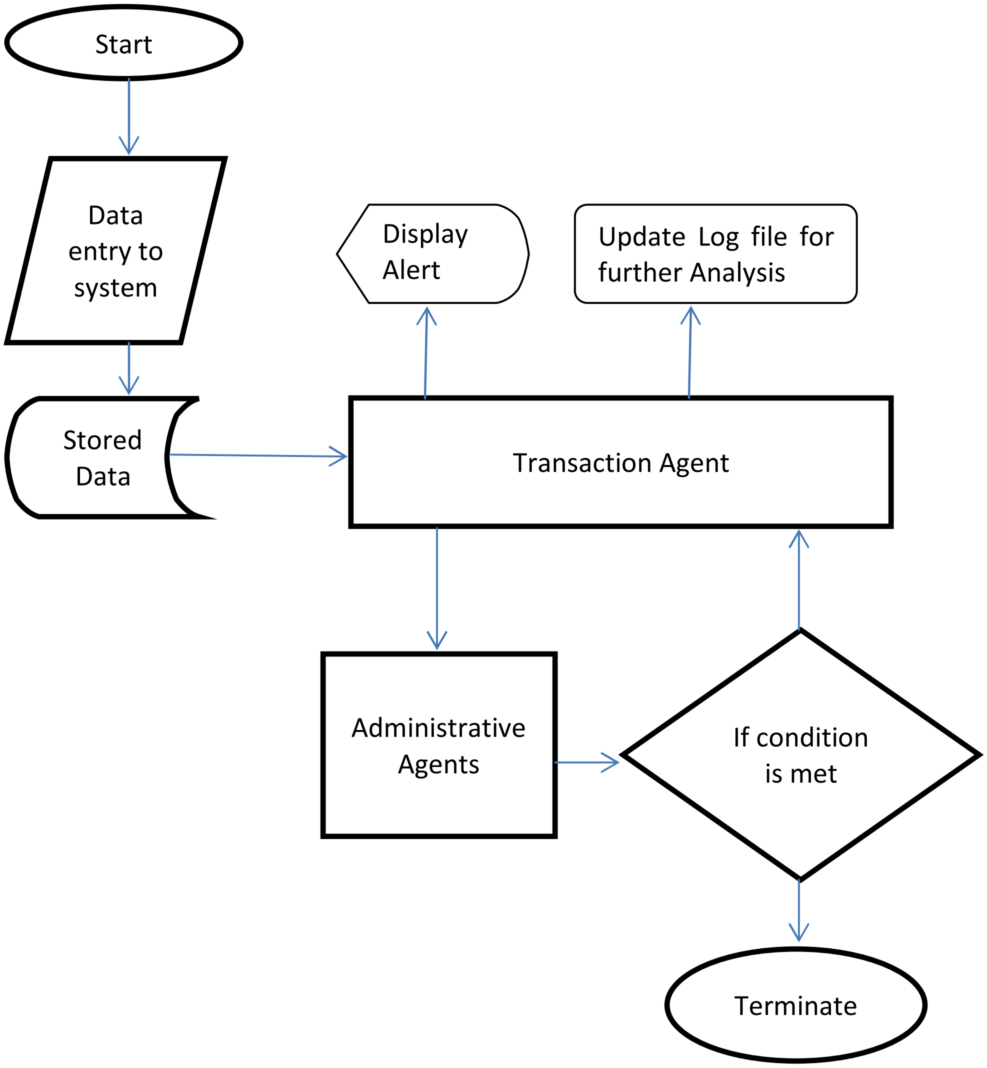
The Desired system should be able to perform the following tasks:-

1. Capture Account number that user is accessing on core banking system
2. Pass the number to agents
3. Check each account number against set rules
4. Report back any set of rules that are broken
5. Display an alert for every rule broken

4.4 Architectural design

The architectural design describes the overall system view communication protocols and the relationships between the agents through descriptors.

Figure 2 illustrates data flow between Agents the external environment



4.3.1 Agents System Overview in relation to the Bank

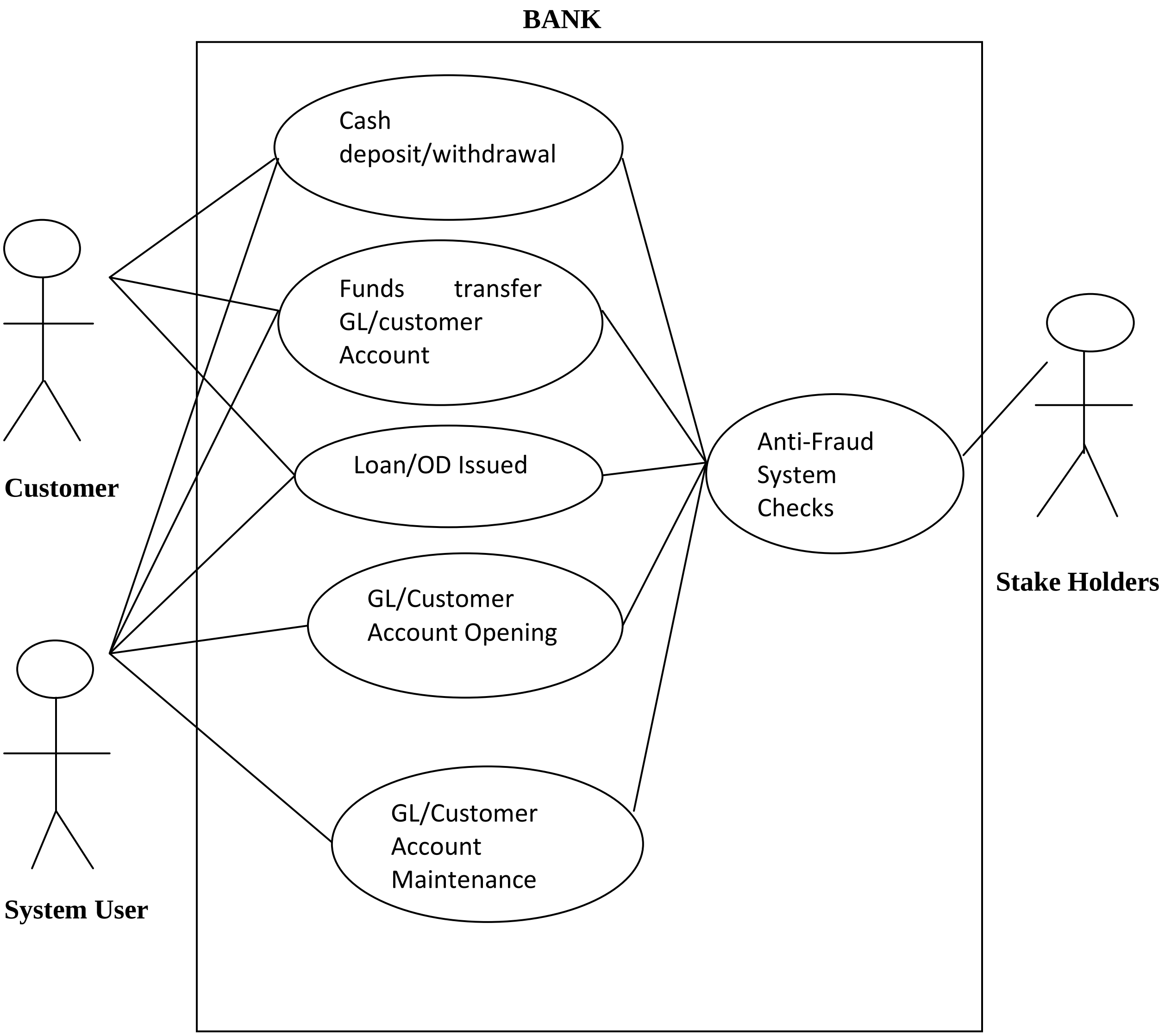


Figure 3 System Use Case Diagram

4.4 Detailed Design

The architecture of the MAS is composed by two agent's categories:

Type And Level	**Agent category**	**Responsibilities**
Level 0	Transaction agent	1.Capture account number transacted into by the users 2.passing the number to the administrative Agents 3.Give out an alert in response from the administrative agents 4. Write response from administrative agents into log file
Level 1	Administrative Agents	1.Checks specified condition to check for every account number received from the transaction agent 1.Reports back to the transaction agent

Table 1: Agents types, category and responsibility

i) **Transaction agent**

This agent has two main roles

1. Capture account number transacted into by the users
   [Select distinct (Account ID) from transaction table]
2. passing the number to the administrative Agents
3. Give out an alert in response from the administrative agents
4. Write response from administrative agents into log file

ii) **Administrative Agents**

Each agent is given a particular condition to check and does so for every account number received from the transaction agent and if the condition to be check is met form any of the account, the agent reports back to the transaction agent for an alert to be raised the following are the available administrative agents

- **Dormant account agent** [checks for transaction into account that has been activated and transacted into on the same day]

  -[If exists(select Account ID from Dormant Account table inner join Transaction Table on Dormant Account table.AccountID= transaction table.AccountID where Dormant Account table .Activatedon= transaction.Date)]

- **Photo and signature Agent**[checks for transaction into account that has either no signature or photo or either of the two is modified]

  -[If exists(select Account ID from Account Details table inner join Transaction Table on Account Details table.AccountID= transaction table.AccountID where Account Details table .photoID= NULL]

- **Average Debit Transaction Agent** [checks for Transaction amount that is larger than the average debit]

  - [SELECT AccountID from Transaction table where amount > (select sum (Amount)/COUNT (*) from Accounttransactions]

- **Transaction Frequency Agent** [checks for transaction frequency of the account above average for the day]

- [SELECT Account ID from Transaction table where COUNT AccountID) > 4**]**

- **Freeze Amount Agent** [checks for transaction into account that amount was frozen then released and debited on the same day]

  -[If exists(select Account ID from Freeze Account table inner join Transaction Table on Freeze Account table.AccountID= transaction table.AccountID where Freeze Account table .Activatedon= transaction.Date)]

- **Transaction Supervisor Agent**[checks for transaction that the supervision is done by an unusual supervisor]

  - [If exists(select OperatorID from Account transaction table inner join Transaction Table on Account transaction table.operatorID= transaction table.OperatorID where transaction Account table .opertaorID<>transaction.OperatorID)]

4.4 Agents Internal Process

The agent's compatibility overview is through event descriptors and plan descriptors

4.4.1 Event Descriptors

1.

Main Agent [Transaction Account Capture]

- User Interface
- Pass Account Number to Administrative Agents

2.

Administrative Agents

-Receive Account number from Transaction Agent
-Perform specified check against the account number
-Send back feedback to transaction Agent

3.

Transaction Agent

- Write received response from Administrative Agent to text file of User display form

4.2.2 Agents plan Descriptors

Real time fraud detection process requires the following plan of action to be taken

Plan 1: on run time capture the account number being transacted upon by user on the core system

Plan 2: Distribute the account number to the various available Administrative agents

Plan 3: Each Administrative agent using a rule based setup check the pertaining rule against the account number received to check is the rule applies

Plan 4: If rule doesn't apply in plan 3 the administrative agent reports back to the transaction agent

Plan 5: The transaction Agent stores the received message/Alert from the administrative agent and displays the alert to the User using GUI interface

4.2.3 Algorithm

- User posts into core system and transaction is stored into the daily transactions table
- Transaction Agent captures the Account Number being posted into as soon as transaction is stored in the table
- Transaction Agent passes the number to administrative agents
- Administrative agents check on rule set against the Account number received
- Administrative agents report back to transaction agent if any rule is broken
- Transaction agent stores the alert received
- Transaction Agent supervised by manager or rollback the transaction before being committed to database table

4.5.4 Database Design

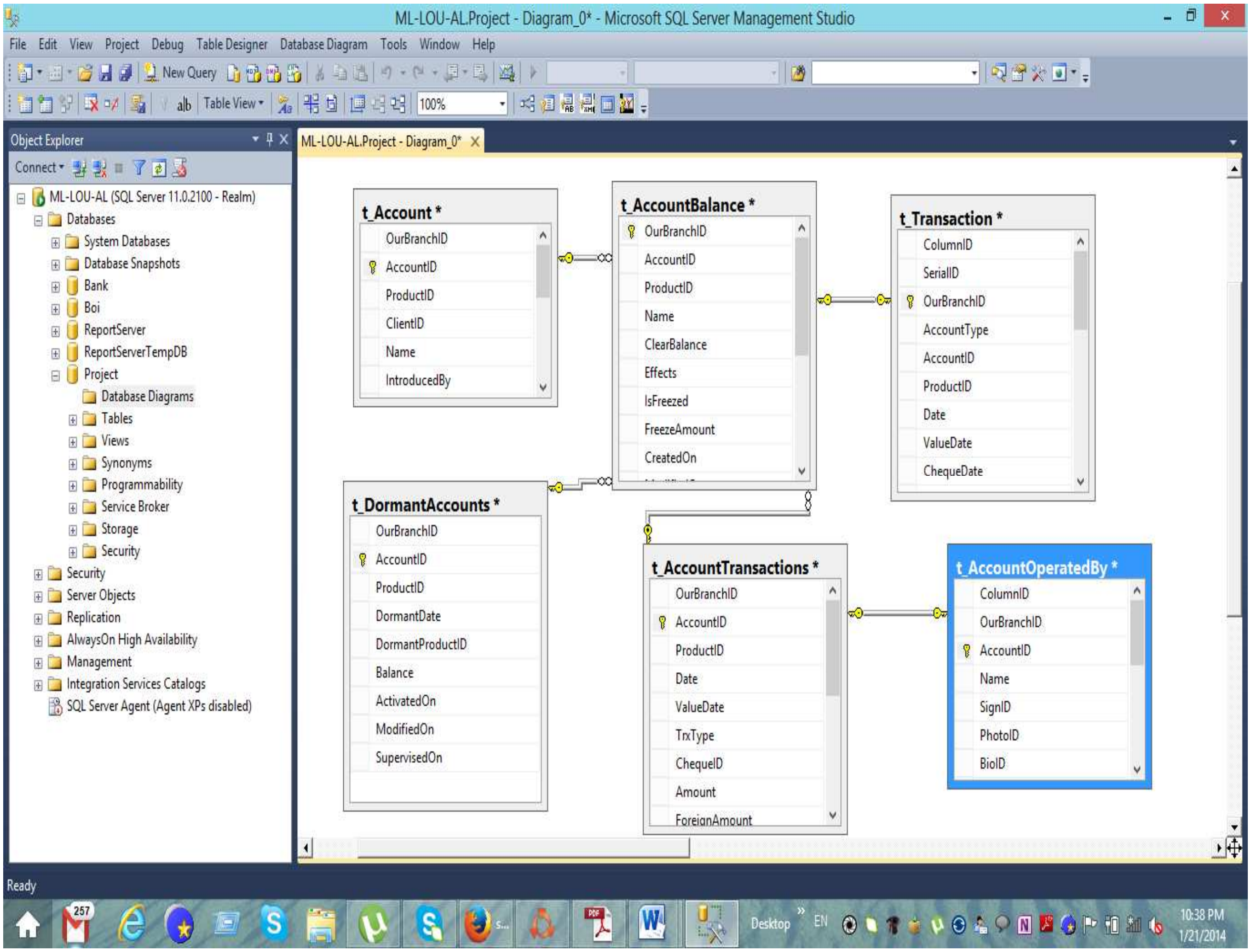


Figure 5 Database Diagram of the sub set of the core banking system

CHAPTER FIVE
SYSTEM IMPLIMENTATION AND RESULTS

5.1 Implementation of the system

The system has been implemented using a combination of frameworks. The Multi –agent component has been implemented using JADE [Java Agent Development Environment] to capture the transaction account on run time while the user interface is implemented using Java Eclipse. The core banking system in stored in a Microsoft SQL server and queries to the database were developed in SQL

5.2 System Testing

The system testing was done in three different banks that were willing to participate in the trial of the system ,these three are Mwanga Community Bank in Mwanga ,Tanzania, Uchumi Commercial Bank in Moshi,Tanzania and Kilimanjaro Co-operative Bank in Moshi,Tanzania . During the User Acceptance Test [UAT] process, a list of different cases was presented as a short description and possible consequences (typically a loss from a bank account) was provided to the users. Users were asked to give a short feedback highlighting positive and negative aspects about what they observed and suggest further developments. The first reaction was a positive evaluation of the improvement and addition of the existing rules on suspicious entries

The system was tested using a system prototype where a sub set of the core banking system database was simulated, then users where given an opportunity to suggest various rules for system to check ,which were incorporated into the system as shown by the various Administrative agents indicated in chapter four system analysis and design [The final set of rules or administrative agents was a combination of the common check points from the four different Banks used as test sets during Data collection phase].

The Real time fraud detection system was connected to the simulated database and then users were asked to post transactions against accounts that had predefined conditions set in the simulateddatabase, these predefined conditions included the following:

1. Dormant account [several accounts where set to have dormancy removed on same date as core system transaction date]
2. Photo and signature [several accounts had there photo and signature image removed or modified]
3. Average Debit Transaction[Posing a large amount into several accounts]
4. Transaction Frequency [doing more than 5 withdrawals for the day on one account]
5. Freeze Amount[several accounts with frozen amount release on same date as core banking system transaction date]

6. Transaction Supervisor Agent[Once transaction is done a supervisor that had not supervised the particular user transaction does the supervision for that day]
7. Login [user with several failed login for the day does a transaction in the system]

The Real Time Fraud Detection system was checked to see if it detected these transactions as suspicious when users posted into the selected accounts and if it raises an alert and kept a log of the same.

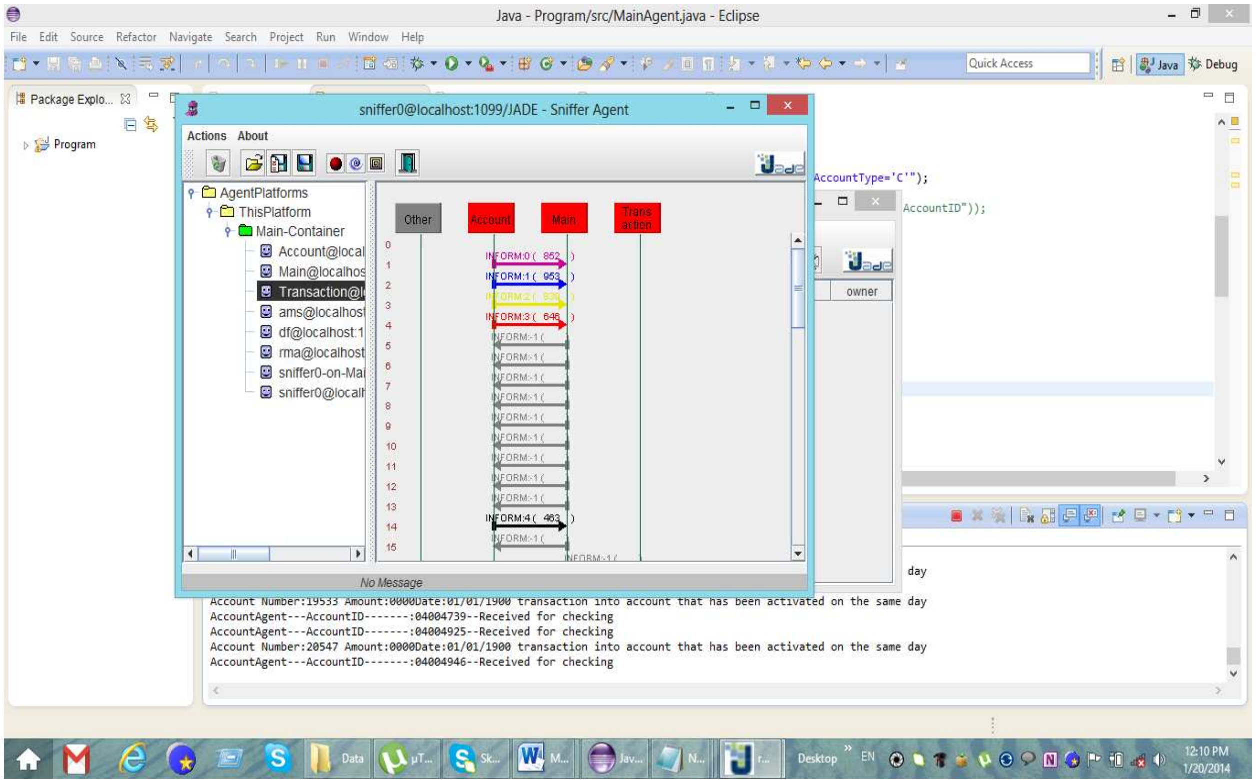
5.3 Discussion of Results

5.3.1 Challenges Facing Real time Fraud Detection

- The biggest challenge facing the system is that it needs to be prompted by the user in order to start monitoring the transactions, these being a user monitoring system a user can purposely disable the system to avoid transactions being monitored.
- The user can also manipulate the log file if they have access to the file.
- User is also in a position to disable the LAN or WAN to disconnect the link between the system and the core bank system database
- Core banking system vendors may refuse to give access to their database or refuse to give the table format thus system unable to know where to check for necessary information

5.4 System Results

The Agent Based system for Real time fraud detection was able to monitor and perform real time alert and notification to accounts that had suspicious entries based on the rules set to monitor what would be considered as suspicious as the transactions happen. It also was able to log the information into a log file. The information on the log file is then made accessible to the user via an interface that the user can use to further analyze the data. The data can be selected by date, grouped by available administrative agent reports .The diagrams and table below show the number of entries that the agents reported as suspicious and how the agents captured the account numbers and check against the set rules and reported back those that had broken any of the rules and updated a log file



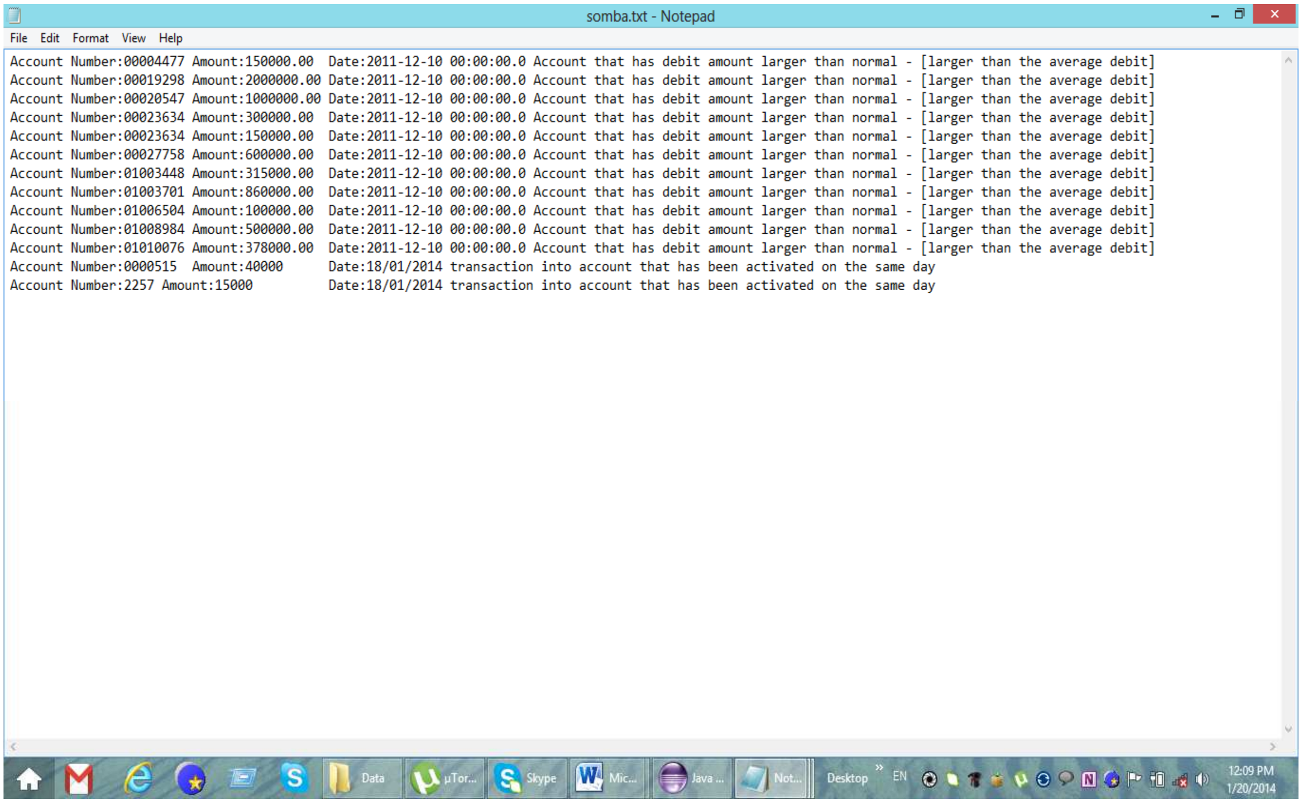
```
Account Number:19533 Amount:0000Date:01/01/1900 transaction into account that has been activated on the same day
AccountAgent---AccountID-------:04004739--Received for checking
AccountAgent---AccountID-------:04004925--Received for checking
Account Number:20547 Amount:0000Date:01/01/1900 transaction into account that has been activated on the same day
AccountAgent---AccountID-------:04004946--Received for checking
```

Figure 6: Agent communication [Transaction Agent] send message to Account and transaction agents [Administrative Agents] and they send back message to Main Agent [Transaction Agent]

Table 2: Agents Results

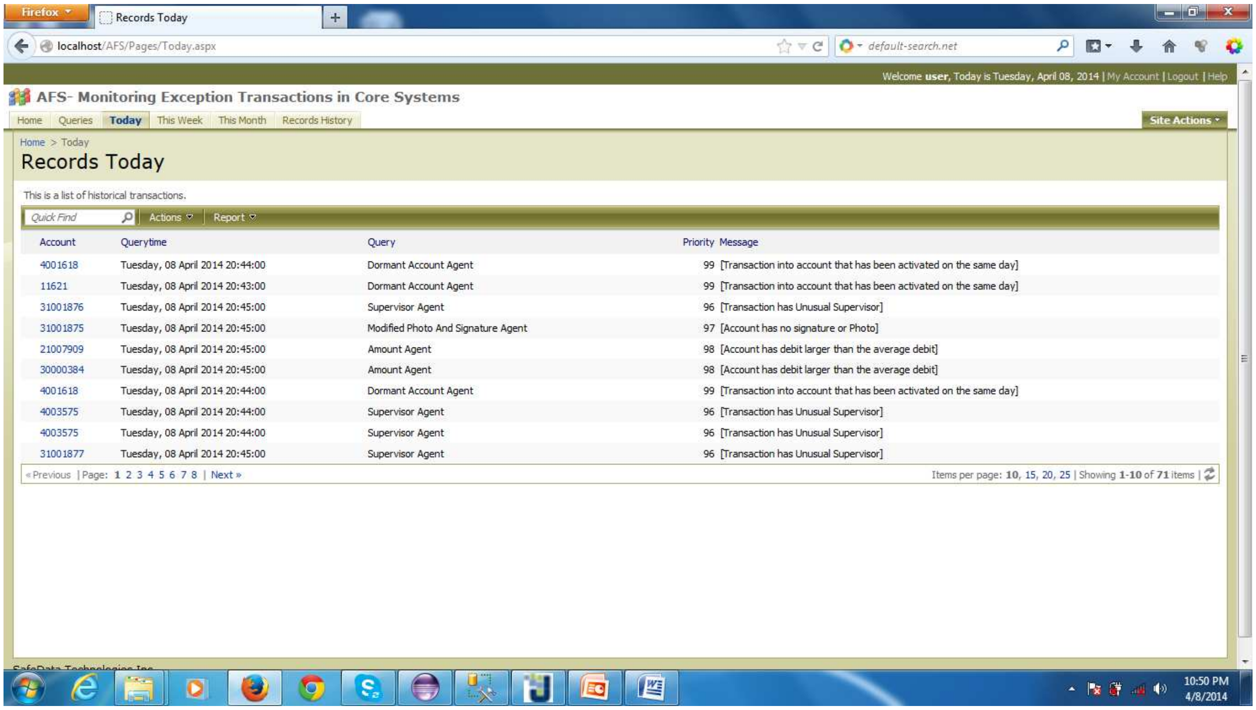
Agent	**Agent Description**	**Total Entries**	**Number Of Alerts**	**Percentage of Alert**
Dormant account agent	checks for transaction into account that has been activated on the same day	248	5	2.4%
Photo and signature Agent	checks for transaction into account that has either no signature or photo or either of the two is modified	248	14	1.4%
Average debit Transaction Agent	checks for Transaction amount that is larger than the average debit	100	15	4%

Transaction Frequency Agent	checks for transaction frequency of the account above average for the day	248	2	0.97%
Freeze Amount Agent	checks for transaction into account that amount was frozen then released and debited on the same day	248	0	0%
Transaction Supervisor Agent	checks for transaction that the supervision is done by an unusual supervisor	248	39	0.48%
Login Agent	checks for transaction by operator with several failed login attempts	248	0	0%



```
Account Number:00004477 Amount:150000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:00019298 Amount:2000000.00 Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:00020547 Amount:1000000.00 Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:00023634 Amount:300000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:00023634 Amount:150000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:00027758 Amount:600000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:01003448 Amount:315000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:01003701 Amount:860000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:01006504 Amount:100000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:01008984 Amount:500000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:01010076 Amount:378000.00  Date:2011-12-10 00:00:00.0 Account that has debit amount larger than normal - [larger than the average debit]
Account Number:0000515  Amount:40000      Date:18/01/2014 transaction into account that has been activated on the same day
Account Number:2257 Amount:15000          Date:18/01/2014 transaction into account that has been activated on the same day
```

Figure 7: System Log file [Transaction Agent] writes alert from Administrative agents to Text file


Firefox
Records Today
localhost/AFS/Pages/Today.aspx
default-search.net
Welcome user, Today is Tuesday, April 08, 2014 | My Account | Logout | Help
AFS- Monitoring Exception Transactions in Core Systems
Home
Queries
Today
This Week
This Month
Records History
Site Actions
Home > Today

Records Today

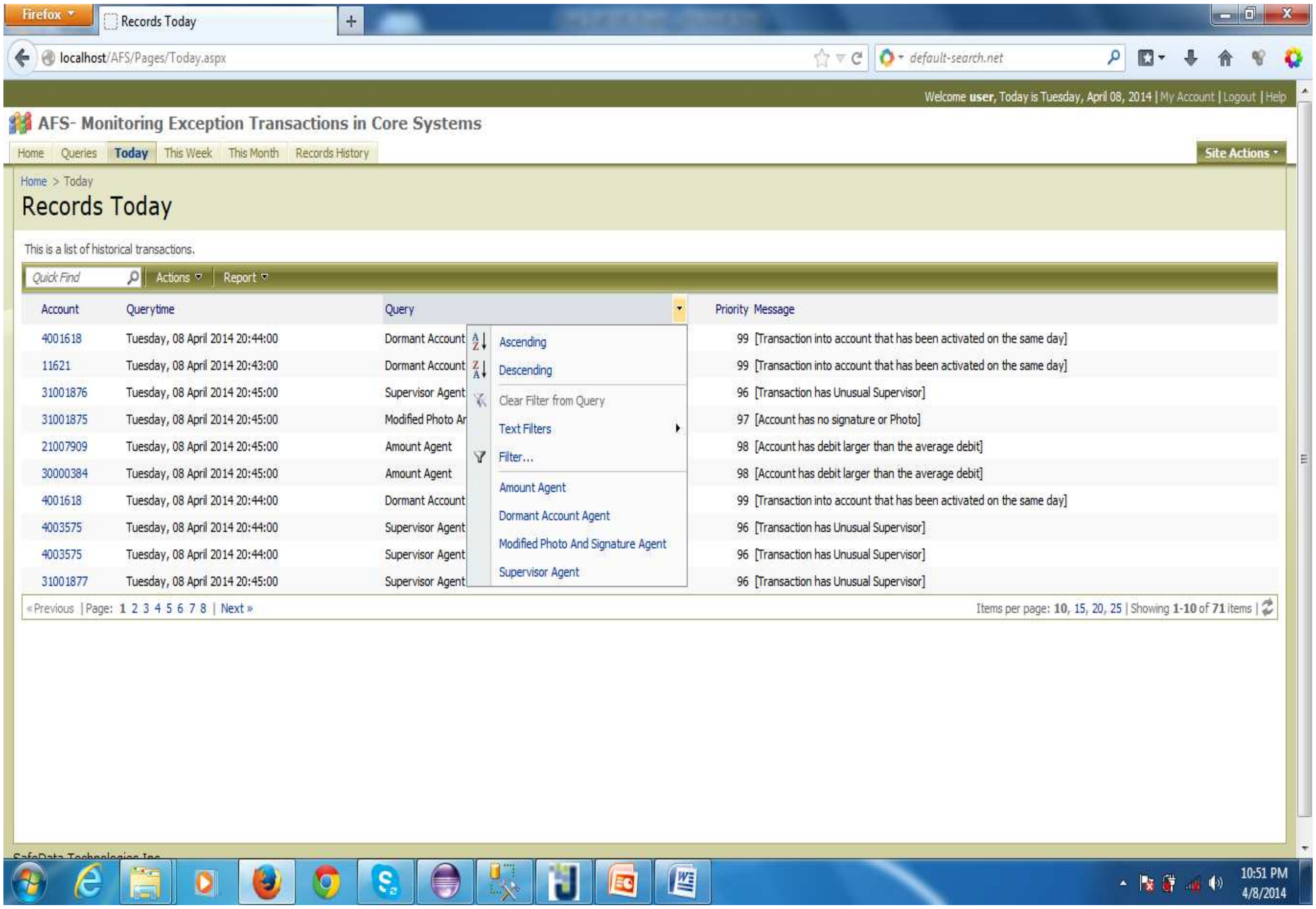
This is a list of historical transactions.

Quick Find
Actions
Report

Account	Querytime	Query	Priority	Message
4001618	Tuesday, 08 April 2014 20:44:00	Dormant Account Agent	99	[Transaction into account that has been activated on the same day]
11621	Tuesday, 08 April 2014 20:43:00	Dormant Account Agent	99	[Transaction into account that has been activated on the same day]
31001876	Tuesday, 08 April 2014 20:45:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]
31001875	Tuesday, 08 April 2014 20:45:00	Modified Photo And Signature Agent	97	[Account has no signature or Photo]
21007909	Tuesday, 08 April 2014 20:45:00	Amount Agent	98	[Account has debit larger than the average debit]
30000384	Tuesday, 08 April 2014 20:45:00	Amount Agent	98	[Account has debit larger than the average debit]
4001618	Tuesday, 08 April 2014 20:44:00	Dormant Account Agent	99	[Transaction into account that has been activated on the same day]
4003575	Tuesday, 08 April 2014 20:44:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]
4003575	Tuesday, 08 April 2014 20:44:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]
31001877	Tuesday, 08 April 2014 20:45:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]

« Previous | Page: 1 2 3 4 5 6 7 8 | Next » Items per page: 10, 15, 20, 25 | Showing 1-10 of 71 items |

Figure 8: System user interface to display logged entries from the Transaction Agent



AFS- Monitoring Exception Transactions in Core Systems

Records Today

This is a list of historical transactions.

Account	Querytime	Query	Priority	Message
4001618	Tuesday, 08 April 2014 20:44:00	Dormant Account	99	[Transaction into account that has been activated on the same day]
11621	Tuesday, 08 April 2014 20:43:00	Dormant Account	99	[Transaction into account that has been activated on the same day]
31001876	Tuesday, 08 April 2014 20:45:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]
31001875	Tuesday, 08 April 2014 20:45:00	Modified Photo Ar	97	[Account has no signature or Photo]
21007909	Tuesday, 08 April 2014 20:45:00	Amount Agent	98	[Account has debit larger than the average debit]
30000384	Tuesday, 08 April 2014 20:45:00	Amount Agent	98	[Account has debit larger than the average debit]
4001618	Tuesday, 08 April 2014 20:44:00	Dormant Account	99	[Transaction into account that has been activated on the same day]
4003575	Tuesday, 08 April 2014 20:44:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]
4003575	Tuesday, 08 April 2014 20:44:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]
31001877	Tuesday, 08 April 2014 20:45:00	Supervisor Agent	96	[Transaction has Unusual Supervisor]

« Previous | Page: 1 2 3 4 5 6 7 8 | Next »

Items per page: 10, 15, 20, 25 | Showing 1-10 of 71 items |

Figure 9: user interface used for data analysis on the information in the log file

CHAPTER SIX

CONCLUSION AND FURURE WORK

6.1 Achievements

The objective of the research which was to develop a proactive multi agent based real time fraud detection system is considered to have been technically achieved .Before addressing the design of the system, several experts in charge of diverse risk management activities inside several commercial and community banks were consulted, the research has been inspired by the information collected during these meetings. The research project demonstrated that through treasuring the knowledge and the opinions collected during the meetings, a multi-agent system prototype of pure reactive agents intended to detect fraud using rules set to determine suspicious activities in the core banking system is achievable. The prototype is intended for the management of fraud detection situations where system users and auditors are collaborating according to a three-phase detection process. In the first phase users and auditors, respectively, an effective environment to explicit their knowledge, select the most likely fraud attack components. In the second phase, the components are structured into rules that act as thresholds used for monitoring each account that is transacted into in the core system. In the third phase an alarm is raised for each threshold reached

The successful solution of this kind of problems depends to a large extent on a proper definition of rules that determine a suspicious event.

6.2: Research Contributions

The research has focused on a proactive fraud detection mechanism; this research project demonstrates the agent based technology development approach in detection of suspicious events in a bank .The fraud detection using multi agents technology is event driven in that as a transaction takes place on the financial system the agents performs a check against a set of set rules to determine any suspicious anomaly in the involved account to prevent data manipulation or user assumption on key aspects pertaining to the account that could lead to fraud

6.3 Recommendations/Future Work

The system can be improved and advanced based on the following:

- Addition of intelligent agents to the current reactive agents to include user profiling on the process of fraud detection
- Adaptation of the fraud detection system so as to use databases that are not of SQL query nature for example graph based database

- Figure out a way for the agents to be constantly checking the events on the core banking system other than the current user prompted way that’s activated when user logs in the fraud detection system, an example could be a web service that runs the agents when user is not logged in the system

6.4 Assumptions and Limitations

One of the major limitations of the fraud detection system is that it is user prompted and depends on a user login in for the agents to start their checks ,fraud can occur while the system is off.

Another limitation is that the system only gives alerts that a user needs to act upon; the system does not stop the transaction from taking place so if user ignores an alert the fraud can still occur despite being detected in good time.

The system is currently based on databases that support SQL query Language if the core banking system operates on a database that doesn’t support SQL query language the fraud detection system will not work

The assumption here is that the existing checks are sufficient, this checks are based on users and auditors observation and experience, so if a new fraudulent event occurs it will go unnoticed unless added to the existing events to detect such future occurrences.

REFERENCES

i) Alessandro, Buoni. (2012-11-08) Fraud detection in the banking sector: a multi-agent approach

ii) Anyanzwa ,James. (2012)“More worries as Kenyan banks lose Sh1.5 billion to fraud”.http://www.standardmedia.co.ke/?articleID=2000090071&story_title=more-worries-as-banks-lose-sh1-5-billion-to-fraud”

iii) “Eug´enio Rosas and Cesar Analide”.Telecommunications Fraud: Problem Analysis -an Agent-based KDD Perspective,pages 408-410

iv) “Fraud Solutions for Africa Banks .(2013) – A Kenyan Perspective”. The Voice Of Banking AndFinanceInAfrica.http://www.technologybanker.com/security-risk-management/fraud-solutions-for-africa-banks-a-kenyan- perspective#.Uh4Z_lK2tdg

v) Fawcett, T., & Provost, F. (1997). Adaptive Fraud Detection. Data Mining and Knowledge Discovery, 1(3):291-316.

vi) Hand, D.J. (2007) Statistical Techniques for Fraud Detection and Evaluation, http://langtech.jrc.ec.europa.eu/mmdss2007/htdocs/Presentations/Docs/MMDSS_Hand_PUBLIC.pdf

vii) Gao, S., Xu, D., Wang, H., & Wang, Y. (2006). Intelligent Anti-Money Laundering System. IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI 06), Shanghai, China, 21-23 June 2006, pp. 851-856. Los Alamitos, CA: IEEE.

viii) KamauMbote.(2013)” Kenyan banks lose $17.5m to fraud”.The East African.http://www.humanipo.com/news/5981/kenyan-banks-lose-175m-to-fraud/

ix) KPMG Forensic fraud barometer (2009) http://www.yhff.co.uk/Fraud%20Barometer%20-%20Feb%202009%20_2_.pdfKPMG Fraud survey (2009) http://www.kpmginstitutes.com/aci/insights/2009/pdf/kpmg-fraudsurvey-2009.pdf

x) Mugwe,David.(2013)”Business_daily”.http://www.businessdailyafrica.com/Kenyan+banks+biggest+victims+of+Sh4bn+fraud+/-/539552/1467902/-/2aehgrz/-/index.html

xi) WambuiNdonga.(2013)”4 bank workers freed on Sh10m cash bail each”. Capital News.http://www.capitalfm.co.ke/news/2013/08/4-bank-workers-freed-on-sh10m-cash-bail-each

xii) Lin ,P & Michael,W (2004) The Prometheus Methodology pages 4

Appendix 1

```
package program;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import jade.core.behaviours.*;
import jade.lang.acl.*;
import jade.core.AID;
import jade.core.Agent;
import java.io.BufferedWriter;
import java.io.FileWriter;


public class MainAgent extends Agent

//Main agent collects Account Number and distributes to other agents
 {


    public static final int QUIT = 0;

            protected void setup()
    {

        addBehaviour(new SimpleBehaviour( this )
            {
                int n=0;
                int i=0;


    public void action()
      {
        try
      {
        //Connect to the database
         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
  Connection connection = DriverManager
.getConnection("jdbc:sqlserver://localhost:1433;databaseName=mwanga;selectMethod=cursor","sa", "friend");

 // select Account that has been transacted into
Statement statement = connection.createStatement();
ResultSet resultSet = statement
.executeQuery("select distinct AccountID,ColumnID from t_Transaction where AccountType='C' and ColumnID
not in (select distinct(ColumnID) from t_ColumnID)");
while (resultSet.next()) {
System.out.println("Account--Number:" + resultSet.getString("AccountID"));
//Distribute Account Number to available Agents
ACLMessage  Trackmsg = new ACLMessage(ACLMessage.INFORM);
Trackmsg.setContent( resultSet.getString("AccountID"));
Trackmsg.addReceiver( new AID( "TrackAgent", AID.ISLOCALNAME));
send(Trackmsg);
//checks for transaction into account that has been activated and transacted into on the same day

ACLMessage  DormantAccountmsg = new ACLMessage(ACLMessage.INFORM);
DormantAccountmsg.setContent( resultSet.getString("ColumnID"));
DormantAccountmsg.addReceiver( new AID( "DormantAccountAgent", AID.ISLOCALNAME));
send(DormantAccountmsg);

// checks for transaction amount that is above average for the account
ACLMessage  TransactionAmountmsg = new ACLMessage(ACLMessage.INFORM);
TransactionAmountmsg.setContent( resultSet.getString("AccountID"));
TransactionAmountmsg.addReceiver( new AID( "TransactionAmountAgent", AID.ISLOCALNAME));
send(TransactionAmountmsg);

checks for transaction frequency of the account above average for the day
ACLMessage  TransactionFrequencymsg = new ACLMessage(ACLMessage.INFORM);
TransactionFrequencymsg.setContent( resultSet.getString("AccountID"));
TransactionFrequencymsg.addReceiver( new AID( "TransactionFrequencyAgent", AID.ISLOCALNAME));
send(TransactionFrequencymsg);

//checks for transaction into account that has either no signature or photo or either of the two is modified
ACLMessage PhotoAndSignaturemsg = new ACLMessage(ACLMessage.INFORM);
PhotoAndSignaturemsg.setContent( resultSet.getString("AccountID") );
PhotoAndSignaturemsg.addReceiver( new AID( "PhotoAndSignatureAgent", AID.ISLOCALNAME));
send(PhotoAndSignaturemsg);

//checks for transaction into account that has freeze funds released and transacted into on the same day
ACLMessage FreezeAmountAgent = new ACLMessage(ACLMessage.INFORM);
FreezeAmountAgent.setContent( resultSet.getString("AccountID") );
FreezeAmountAgent.addReceiver( new AID( "FreezeAmountAgent", AID.ISLOCALNAME));
send(FreezeAmountAgent);

//checks for transaction that the supervision is done by an unusual supervisor
ACLMessage supervisorIDmsg = new ACLMessage(ACLMessage.INFORM);
supervisorIDmsg.setContent( resultSet.getString("AccountID") );
supervisorIDmsg.addReceiver( new AID( "supervisorIDAgent", AID.ISLOCALNAME));
send(supervisorIDmsg);

//checks for any changes done directly into the system key tables
ACLMessage Checksummsg = new ACLMessage(ACLMessage.INFORM);
Checksummsg.setContent( resultSet.getString("ColumnID") );
Checksummsg.addReceiver( new AID( "ChecksumAgent", AID.ISLOCALNAME));
send(Checksummsg);

// Store record for AccountID already checked by agents
Statement statement1 = connection.createStatement();
ResultSet resultSet1 = statement1
.executeQuery("insert into t_ColumnID (columnID) select distinct(columnID) from t_Transaction where AccountType='C' and columnID not in (select columnid from t_ColumnID )");

}
  } catch (Exception f) {f.printStackTrace();}
      try
         {
           ACLMessage Message = receive();
```

```
// Writes results to output file for user interaction with the system
  System.out.println(Message.getContent());
  FileWriter fileWritter = new FileWriter("D:/Data/RAFS.txt",true);
  BufferedWriter bufferWritter = new BufferedWriter(fileWritter);
  bufferWritter.write(Message.getContent());
  bufferWritter.newLine();
  bufferWritter.flush();
  bufferWritter.close();

          }
                       catch (Exception c) {c.printStackTrace();}


          }


         public boolean done() {  return n>=10;  }


         });

         }

    }
```

Appendix 2

```
package program;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import jade.core.Agent;
import jade.core.behaviours.*;
import jade.lang.acl.ACLMessage;

public class PhotoAndSignature  extends Agent
{
protected void setup()
{
          addBehaviour(new CyclicBehaviour(this)
          {
                    int n=0;
                    public void action()
                    {
                              ACLMessage PhotoAndSignaturemsg = receive();
                              ACLMessage PhotoAndSignatureSentmsg = receive();

try {

// connect to database
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
Connection connection = DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databaseName=mwanga;
selectMethod=cursor", "sa", "friend");
System.out.println("PhotoAndSignature Agent AccountID:" + PhotoAndSignaturemsg.getContent() +"--"+
"Received for checking");

// Perform task
Statement statement = connection.createStatement();
ResultSet resultSet = statement.executeQuery( "SELECT distinct
t_Transaction.AccountID,t_Transaction.Amount,t_Transaction.date from t_Transaction inner join
t_AccountOperatedBy "
+ "on t_AccountOperatedBy.AccountID=t_Transaction.AccountID where t_Transaction.AccountID=" +
PhotoAndSignaturemsg.getContent() +""
+ "and signID is null ");

while (resultSet.next()) {
ACLMessage replyA = PhotoAndSignatureSentmsg.createReply();
replyA.setPerformative( ACLMessage.INFORM );
replyA.setContent(("Account Number:" + resultSet.getString("AccountID") + " "
+ "Amount:" + resultSet.getString("Amount")
+ "Date:" + resultSet.getString("Date")
+ " "+"Account has no signature or Photo "));


try{
      if (replyA!=null)
       send(replyA);


}
catch (Exception e) {e.printStackTrace();}

try {
// update track for Account
Statement statement1 = connection.createStatement();
ResultSet resultSet1 = statement1
.executeQuery("insert into t_ColumnID (columnID) select " + PhotoAndSignatureSentmsg.getContent() +"");

}
      catch (Exception p) {p.printStackTrace();}
}

}
catch (Exception e) {e.printStackTrace();} }

});}       }
```

Appendix 3

```
package program;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import jade.core.Agent;
import jade.core.behaviours.*;
import jade.lang.acl.ACLMessage;

public class supervisorIDAgent  extends Agent
{
protected void setup()
{
        addBehaviour(new CyclicBehaviour(this)
        {
                int n=0;
                public void action()
                {
                        ACLMessage supervisorIDmsg = receive();
                        ACLMessage supervisorIDSentmsg = receive();


try {
 Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
Connection connection =
DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databaseName=Mwanga;selectMethod=cursor",
"sa", "friend");

System.out.println("Supervisor Agent AccountID:" + supervisorIDmsg.getContent() +"--"+ "Received for
checking");

Statement statement = connection.createStatement(); ResultSet resultSet = statement.executeQuery
("select distinct t_transaction.AccountID,t_transaction.Amount,t_transaction.Date from t_transaction inner join " +
"t_AccountTransactions on t_AccountTransactions.AccountID=t_Transaction.AccountID " + "and
t_AccountTransactions.SuperVisorID=t_Transaction.SuperVisorID where t_Transaction.AccountID=" +
supervisorIDmsg.getContent() +"");


while (resultSet.next()) {
ACLMessage replyA = supervisorIDSentmsg.createReply();
replyA.setPerformative( ACLMessage.INFORM );
replyA.setContent(("Account Number:" + resultSet.getString("AccountID") + " "
+ "Amount:" + resultSet.getString("Amount")+ "Date:" + resultSet.getString("Date")
+ " "+"Transaction has Unusual Supervisor "));

  try{
     if (replyA!=null)
     send(replyA);
 }
     catch (Exception e) {e.printStackTrace();}


  try {
     Statement statement1 = connection.createStatement();
     ResultSet resultSet1 = statement1
        .executeQuery("insert into t_ColumnID (columnID) select " + supervisorIDmsg.getContent() +"");

        }
         catch (Exception p) {p.printStackTrace();}
        }
            }
```

```
                        catch (Exception e) {e.printStackTrace();}
                    }

                              });
                 }
              }
```

Appendix 4

```
package program;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import jade.core.AID;
import jade.core.Agent;
import jade.core.behaviours.*;
import jade.lang.acl.ACLMessage;

public class AmountTransactionAgent extends Agent
{
protected void setup()
{
        addBehaviour(new CyclicBehaviour(this)
        {
                int n=0;
                public void action()
                {
                        ACLMessage TransactionAmountmsg = receive();
                        ACLMessage TransactionAmountSentmsg = receive();


        try {
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
Connection connection =
DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databaseName=mwanga;selectMethod=cursor",
"sa", "friend");

System.out.println("Transaction Agent AccountID:" + TransactionAmountmsg.getContent() +"--"+ "Received for
checking");
Statement statement = connection.createStatement();
ResultSet resultSet = statement.executeQuery
( "SELECT  AccountID,Amount,date from t_Transaction where trxtype in ('D','Z','I') and AccountID=" +
TransactionAmountmsg.getContent()+"" + " and amount >(select sum(Amount)/COUNT(*) from
t_Accounttransactions where trxtype in ('D','Z','I')and OperatorID <>'realm' "+ "and AccountID=" +
TransactionAmountmsg.getContent()+")");

while (resultSet.next()) {
```

```
replyA = TransactionAmountSentmsg.createReply();
replyA.setPerformative( ACLMessage.INFORM );
replyA.setContent(("Account Number:" + resultSet.getString("AccountID") + " "
+ "Amount:" + resultSet.getString("Amount")+ "   Date:" + resultSet.getString("Date")
+ " "+"Account that has debit amount larger than normal - [larger than the average debit] "));

           try{
           if (replyA!=null)
           send(replyA);

                        }
    catch (Exception e) {e.printStackTrace();}
try {
Statement statement1 = connection.createStatement();
ResultSet resultSet1 = statement1
.executeQuery("insert into t_ColumnID (columnID) select " + TransactionAmountmsg.getContent() +"");

  }
           catch (Exception p) {p.printStackTrace();}


}

}

catch (Exception e) {e.printStackTrace();}
}

});
  }
}
```

Appendix 5

```
package program;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import jade.core.Agent;
import jade.core.behaviours.*;
import jade.lang.acl.ACLMessage;

public class checksumAgent extends Agent
 {
    protected void setup()
    {
       addBehaviour(  // -------- Anonymous SimpleBehaviour

          new SimpleBehaviour( this )
          {
             int n=0;
```

```
                public void action()
                {
                            ACLMessage Checksummsg = receive();

                            try {
                                  Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
Connection connection = DriverManager.getConnection(
"jdbc:sqlserver://localhost:1433;databaseName=project;selectMethod=cursor",
"sa", "friend");
Statement statement = connection.createStatement();
ResultSet resultSet = statement.executeQuery("SELECT top 1 (BINARY_CHECKSUM (*)) as Total from
t_Account");
while (resultSet.next()) {System.out.println("CheckSumAgent:" + resultSet.getString("Total"));  n++;
                                  }
                              } catch (Exception e) {
                                  e.printStackTrace();
                              }
                }

 public boolean done() {  return n>=1;  }
            }
        );
    }
 }
```

Apendix 6

```
package program;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import jade.core.AID;
import jade.core.Agent;
import jade.core.behaviours.*;
import jade.lang.acl.ACLMessage;

public class AmountTransactionAgent extends Agent
{
protected void setup()
{
       addBehaviour(new CyclicBehaviour(this)
       {
              int n=0;
              public void action()
              {
                   ACLMessage TransactionFrequencymsg = receive();
                   ACLMessage TransactionFrequencySentmsg = receive();
try {
              Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
             Connection                            connection                            =
DriverManager.getConnection("jdbc:sqlserver://localhost:1433;databaseName=mwanga;sele
ctMethod=cursor", "sa", "friend");

System.out.println("Transaction            Agent          AccountID:"            +
TransactionFrequencymsg.getContent() +"--"+ "Received for checking");

Statement statement = connection.createStatement();
ResultSet resultSet = statement.executeQuery
(  "SELECT    AccountID,Amount,date  from  t_Transaction  where   AccountID="  +
TransactionFrequencymsg.getContent()+"" + " and (select COUNT(*) from  t_transaction
where  trxtype  in  ('D','Z')  and  OperatorID  <>'realm'  "  +  "and  AccountID="  +
TransactionFrequencymsg.getContent()+" > 4)");

while (resultSet.next()) {
ACLMessage replyA = TransactionFrequencySentmsg.createReply();
replyA.setPerformative( ACLMessage.INFORM );
replyA.setContent(("Account Number:" + resultSet.getString("AccountID") + " "
+ "Amount:" + resultSet.getString("Amount")+ "   Date:" + resultSet.getString("Date")
+ " "+"Account has High number of withDrawals - [larger than the average withdrawal
frequency] "));

try{
   if (replyA!=null)
     send(replyA);

}
      catch (Exception e) {e.printStackTrace();}
      try {

Statement statement1 = connection.createStatement();
ResultSet  resultSet1  =  statement1.executeQuery("insert  into  t_ColumnID  (columnID)
select " + TransactionFrequencymsg.getContent() +"");

}
   catch (Exception p) {p.printStackTrace();}


                       }


                        }


                              catch (Exception e) {e.printStackTrace();}
                         }

                           });
                     }
                }
```