UNIVERSITY OF NAIROBI

MASTER OF SCIENCE IN COMPUTER SCIENCE

PROJECT

MULTI AGENT SYSTEM FOR INFORMATION SERVER INTEGRITY MAINTENANCE

BY: JOHN M MACHARIA

REG NO: P58/61537/2010

SUPERVISOR: Mr. ERIC AYIENGA

DECLARATION

This project as presented in this report is my original work and has not been presented for any other institutional award.

Sign: ______________________ Date: ____________________

John Muritu Macharia

P58/61537/2010

This project has been submitted in partial fulfillment of the requirements for the Masters of Science in Computer Science of the University of Nairobi with my approval as the university supervisor.

Sign: ______________________ Date: ____________________

Mr. Eric Ayienga

School of Computing and Informatics

University of Nairobi

ABSTRACT

Decision making about the health workforce in Kenya is very crucial as far as efficient delivery of health services is concerned. Today, more than 60% of the health workforce is located in Nairobi, leaving the rest of the country with insufficient health personnel thus lowering the quality of health services provided in those areas especially in the North Eastern part of Kenya.

Currently, this very crucial information is being held by the various regulatory bodies which manage the specific cadres; i.e. the Medical Practitioners and Dentists Board, the Nursing Council of Kenya, The Clinical Officers Council among others. Current practices are that whenever the health managers need information to assist them in decision making, they have to send someone to the respective regulatory boards to get the information using a storage device. This has proven to be inefficient since moving from the Ministries of Health to the regulatory bodies is prone to many issues.

Due to the inefficiencies involved with the current practices of getting the data, this research has identified that there is need to make this data more readily available to the Health managers to support them in their decision making using better practices.

In this research, we set out to ensure that data is always available to health managers to support them in decision making using multi-agent systems. We used the GAIA methodology for multi-agent systems for analysis and design of the system. We also used an open source Java toolkit, JADE framework for building multiple agents. The multiple agents are responsible for ensuring that the information available to the health managers is always up to date by transporting the most current data from the regulatory bodies to the ministries of health.

Use of the multi agent based system proved to be a better way of transporting the data compared to current practices in terms of speed, security and confidentiality.

ACKNOWLEDGEMENTS

First, my appreciations go to the Lord Almighty, for enabling me to get this far with the quest for knowledge.

Secondly, my appreciations go to my parents who have sacrificed a lot and worked tirelessly to get me where I am today, and to them I dedicate this work.

Thirdly, I am very grateful to my supervisor, Mr. Eric Ayienga for his insights and guidance.

And finally, to other supervisors at the school of computing, thank you for your advice and encouragement.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

MAS – Multi Agent Systems	
DBMS – Database Management System	
DDB – Distributed Database	
JDBC - Java Database Connectivity	
MASIF - Mobile Agent System Interoperability Facility	
FIPA – Foundation for Intelligent Physical Agents	
JADE - Java Agent Development Environment	
JADE-LEAP - Java Agent Development Environment-Lightweight Extensible Agent Platform	
SOA - Service Oriented Architecture	
ETL - Extract, Transform and Load	
MPDB – Medical Practitioners and Dentists Board	
NCK – Nursing Council of Kenya	
COC – Clinical Officers Council	
KMLTTB – Kenya Medical Laboratory Technicians and Technologists Board	
WHO – World Health Organization	
MOH – ministries of Health	
Legacy System – The system currently in use which involves sending someone to go get the data using a flash disk.	

1. INTRODUCTION

1.1 BACKGROUND

The Health industry is one of the key industries that determines the development of a country and therefore requires careful and efficient monitoring. Until recently, the health workforce in Kenya was all in paper form which made it really difficult to know the number of health professionals we had in Kenya and if they were:

- Practicing or not.
- Practicing in Kenya or abroad.
- Where they were practicing (distribution of health professionals in Kenya)
- Etc

It used to take the Ministries of health a few months to get reports on health professionals in Kenya which was really inconveniencing the health managers. Thanks to the Kenya Health Workforce Informatics System, we are now able to accurately know the numbers of health professionals we have in Kenya and it only takes a few minutes now. Being a registered health professional is not enough; the professionals need to comply with Continuous Professional Development which is mandatory before one can be retained in the annual register. This helps the professionals sharpen their skills and be in touch with current health practices. With the implementation of the Kenya Health Workforce Informatics System, this can now be efficiently tracked and monitored by the regulatory bodies. This information is not readily available at the Ministries of Health.

According to the 2006 World Health Organization report, Kenya should have 250 health professionals per 100,000 of the population so as to provide quality health care. Due to the accurate information now available at the health professionals' regulatory bodies, it is now easy to calculate and compare our health standards to the international health standards set by international organizations like the WHO. This and many other health standards and requirements are meant to be met by the Health workforce and the Health Managers need an easy and efficient way to track and get access to this information for management and planning purposes.

1.1.1 RELATIONSHIP AND ROLES BETWEEN REGULATORY BODIES AND MINISTRIES OF HEALTH

The regulatory bodies are responsible for ensuring that health professionals have the right qualifications before they can be employed by the ministries of health. The regulatory bodies are responsible for ensuring;

- that the health professionals are registered,
- that the health professionals are complying with continuous professional development,
- that health professionals do not have pending disciplinary cases,
- that health professionals are licensed to practice among other responsibilities.

The ministries of health employ and deploy the health professionals in the different health facilities and departments. Before the ministries of health employ the health professionals, they need to verify that the health professionals are fit to be employed and that's when they communicate with the regulatory bodies to ensure that the health professionals are fit to practice. Even before they deploy the health professionals, the ministries of health communicate with the regulatory bodies to verify their qualifications so that they can be deployed in the relevant departments.

1.2 PROBLEM STATEMENT

The distribution of health professionals in the country is biased to certain areas due to inadequate information; the number of health professionals in Kenya per 100,000 of the population is way below the International standards set by the WHO in 2006. The Kenya Health Workforce Informatics project which was implemented recently has automated the regulatory bodies and is in the process of automating the ministries of health, but forgot to automate how the health managers at the ministries of health would retrieve the data at the regulatory bodies to help them verify the qualifications (and other details) of the health professionals they want to employ. The health managers can now be able to better manage and plan issues regarding health professionals thanks to the implementation of the new system. The only problem is the fact that this information is not readily available to the decision makers at the Ministries of Health.

At the moment, this important data is being held by the various regulatory bodies who manage the different kinds of cadres (doctors, nurses, clinical officers, laboratory technicians and technologists). To make it easy for the Health managers at the ministries of health to make decisions regarding the health workforce, this distributed data should be brought together to a common place where it can be accessed and queried easily and at the convenience of the health managers.

Currently, the regulatory bodies (Medical Practitioners and dentists Board, Nursing council of Kenya, Clinical Officers council and the Medical Laboratory Technicians and Technologists Board) each have their own database servers which they use independently. When the Health managers at the ministries of health (MMS & MOPHS) require this information, the information has to be retrieved from the servers manually and taken to them (via storage devices e.g. flash disks). This process is costly in terms of time consumed to retrieve the information and also the transportation costs.

To solve this problem, we propose the use of agents to retrieve this information from the servers at the regulatory bodies and export that information to an information server at the Ministries of Health automatically.

1.3 OBJECTIVES

- Research on how multi agent systems have been used in health information systems.
- To design and develop a multi agent system model that will efficiently transport data to a remote server; keeping the data current.

- Develop the multi agent system.
- Analysis and Evaluation of the Agent based method in comparison to the current practices.

1.4 PROJECT JUSTIFICATION

With the implementation of this system, we get to enhance quick and accurate decision making by the health managers at the ministries of health regarding the health workforce, we also get to eliminate the time and money wasted by sending someone to the regulatory bodies anytime a decision needs to be made regarding the health workforce in Kenya.

1.5 LIMITATIONS AND ASSUMPTIONS

Some of the assumptions and limitations in this study will include:

- We will not be in a position to implement the system in the real life scenario therefore we will have to implement it on a simplified environment which will reflect the real life scenario.
- We will not be in a position to use real data; therefore we will use dummy data.
- Internet is available and works perfectly.
- Local area connection is a good representation of the internet connection.

1.6 DOCUMENT ORGANIZATION

This document is organized as follows:

- Chapter 2 is a literature review of technologies used and related work.
- Chapter 3 discusses the methodology used and the reasons for using that methodology.
- Chapter 4 discusses the system analysis and design of the MAS.

- Chapter 5 discusses the implementation of the system.
- Chapter 6 discusses the results collected, analysis of the results and a comparison between the existing and proposed systems.
- Chapter 7 concludes what we set out to do.
- Chapter 8 lists all the references we have used.

2. LITERATURE REVIEW

2.1 MULTI AGENT SYSTEMS

Although there is no universally accepted definition of agent, in this work such an entity is to be understood as a computing artefact, being it in hardware or software, that exhibit the following properties:

- autonomy; i.e., whereby such entities have the ability to act without the direct intervention of their peers, namely humans;
- reactivity; i.e., whereby such entities are situated in an environment that can perceive through sensors and act in reaction to stimuli (e.g., revising their beliefs according to or in reaction to new inputs);
- pro-activity; i.e., whereby such entities exhibit intelligent problem solving capabilities (e.g., planning their activities in order to achieve short or long term goals); and
- social behaviour; i.e., whereby such entities are aware of one another, can interact with one another and may modify their behaviour in response to others.

Agents can communicate via a set of low or high level constructs and protocols as well as means of addressing and direct communication; can cooperate in order to achieve joint as well as individual goals, what means that must have the ability to negotiate with other agents either to accomplish their own goals or to joint plans to achieve common goals; to perform belief revision in the context of additional sources of information provided by their peers.

A Multi-agent system (MAS) is a computerized system composed of multiple interacting intelligent agents within an environment. MAS set a new paradigm in problem-solving via theorem proving; i.e., agent-based computing has been hailed as a significant breakthrough in problem solving and/or a new revolution in software development and analysis. Indeed, agents are the focus of intense interest on many sub-fields of computer science, being used in a wide variety of applications, ranging from small systems to large, open, complex and critical ones; i.e., agents are not only a very promising technology, but are emerging as a new way of thinking, a conceptual paradigm for analyzing problems and for designing systems, for dealing with complexity, distribution and interactivity, may be a new form of computing and intelligence.

MAS will require a software platform on which to execute and a communication link between the software platforms in the case of mobile agents. In the system described in [10], Java agents move from one database to another interacting with databases via the Java Database Connectivity (JDBC) interface. In [11], a framework for accessing databases using mobile agents and JDBC has been developed called the DBMS-Aglet Framework. A DBMS mobile agent (called a DBMS-aglet) carries an itinerary of database servers to visit, security certificates and queries. The DBMS-aglet is sent from a Java applet [12] to a place residing on the database server's host. On arrival, the DBMS-aglet initiates loading of appropriate JDBC drivers for communicating queries to the database server, and is then parked at the host. Additional queries can be sent from the applet to the parked DBMS-aglet via messenger aglets.

2.1.1 AGENT MOBILITY

Mobile agents are a paradigm that derives from two different disciplines. The first is artificial intelligence, which created the concept of an agent, and the second is distributed systems, which defines the concept of code mobility. [1]

According to standard definitions, mobile agents are everything that a non-mobile agent is (i.e. autonomous, reactive, proactive and social), but in addition they are also moveable; they can migrate between platforms in order to accomplish assigned tasks.

From the distributed systems point of view, a mobile agent is a program with a unique identity that can move its code, data and state between networked machines. To achieve this, mobile agents are able to suspend their execution at any time and to continue once resident in another location.

We can position mobile agents in relation to other classical paradigms as follows:

- Client-server: The most widely used paradigm where services are offered by a server and consumed by one or more, usually remote, clients.

- Remote execution: One component sends code to another component for remote execution, either resulting from its own decision, a request from the remote component, or perhaps even as a part of a pre-existing contract. Once executed the executing component will typically return any result to the originating component.
- Mobile agents: One component sends itself (or another, if allowed) to a remote host for execution. The component transitions with its code, data and perhaps state intact. Motivations may be similar to the previous case, but most typically result from the component (i.e. mobile agent) deciding for itself that it wishes to move to an alternate location.

2.1.1.1 ADVANTAGES OF MOBILE AGENTS

There have been many debates over the various advantages and disadvantages of mobile agents, usually in comparison with their non-mobile cousins. Some of the typical advantages are:

- Asynchronous and independent processing: Once they have migrated to a new platform, agents do not have to contact their owner in order to perform their task. They may only need to send back the results. This is especially useful when considering mobile devices with limited resources; an agent can be migrated to another machine to perform complex tasks and periodically return results.
- Fault tolerance: They can address and aid with fault conditions by moving to an alternative platform when problems are detected. Equally if a migration destination is down, an intermediary may be selected as a temporary host. This makes them quite suitable for hostile and disruptive environments.
- Sea of data applications: Mobile agents are well suited to applications that need to process large amounts of remote data. Mobile agents can move to the data, rather than vice versa which in many cases is a much more efficient option.

2.1.1.2 DISADVANTAGES OF MOBILE AGENTS

Mobile agents also have some disadvantages. The most relevant of them are:

- Scalability and performance: Even though mobile agents reduce network load, they also tend to increase processing load. This is because they are usually programmed with interpreted languages and also often need to observe rigorous interoperability standards that can incur data processing overheads.
- Portability and standardization: Agents cannot interoperate if they do not follow common communication standards. Adoption of these standards, such as FIPA is typically necessary, especially for inter-platform mobility.
- Security: The use of mobile agents can bring about security problems. Any mobile code offers a potential threat and should be carefully authenticated before invocation.

2.1.2 STRONG AND WEAK MIGRATION

In mobile agents systems, two primary types of migration can be distinguished: strong migration and weak migration.

Strong migration is more complex and is the case where an agent's execution is frozen, migration takes place, and then execution is restarted from the very next instruction. This technique requires the storage and protection of the agent state during the migration process. Implementation of this technique can be complex as it requires access to internal parameters of agent execution, generally only available to the operating system, and that can typically be very architecture dependent.

Weak migration, on the other hand, does not send the agent state and is therefore much simpler; agent execution always restarts from the beginning of the code. This kind of migration requires that the agent be implemented as a finite state machine if state is to be preserved.

2.1.3 STATELESS AND STATEFUL INFORMATION GATHERING

Mobile agents are sent to gather information and perform tasks on behalf of a user, migrating to the source of information and reducing network bandwidth of messages required to process data. An agent is seen as a migrating program that has both a static part (code) and a dynamic state (data). Each host execution can be said to add new information progressively to the data state of an agent. Initial work in mobile agents such as [9] identified two modes of information gathering: stateless and stateful. In a stateless approach, agents can intermittently or at every hop send information acquired back home to the originator. In a stateful mode, the agent embodies in its data state the results of each host execution and carries with it a growing collection of information to each subsequent host in its itinerary.

2.2 INFORMATION SERVERS

An information server is an integrated software platform consisting of a set of core functional modules that enables organizations to integrate data from disparate sources, clean it up, provide centralized querying and deliver trusted and complete information, at the time it is required and in the format it is needed. [4] An information server delivers consistent information to consuming applications.

The information server can be queried without going back to the original databases. An issue is keeping the Information Server up to date whenever the databases change. The agents extract data from different sources and transfer them to the information server.

Maamar proposes the use of mobile agents to automate this process. [3]

2.2.1 ADVANTAGES OF AN INFORMATION SERVER

Trust

An information server can be deployed to continuously validate the consistency, accuracy and quality of information as it flows from data sources and across applications and business processes. Data quality is ensured throughout the information lifecycle.

Productivity

A unified platform with roll-based user interfaces reduces training costs and learning curves previously required to learn and manage different data integration tools from multiple vendors. Technical teams get up to speed quickly and deliver projects faster.

Collaboration

A unified metadata management layer facilitates the alignment of business users and technical teams through a shared understanding of information's meaning, context, and lineage. Leveraging the metadata shortens the time between specification and build in projects by understanding impact analysis and lineage of data.

Scalability

Parallel processing technology ensures that enormous volumes of information can be processed very quickly. It further ensures that processing capacity is never an inhibitor to achieving project results, allowing solutions to easily expand to new hardware, and to fully leverage the processing power of all available hardware.

Reuse

By enabling integration logic to be packaged and deployed as a service, technical teams using a service oriented architecture (SOA) can leverage work done on previous projects to more efficiently build their solutions while ensuring that consistent rules are applied to information integration, improving data governance.

2.2.2 BUSINESS VALUE OF AN INFORMATION SERVER

Most key business initiatives cannot succeed without effective integration of information. In fact, the IBM Global CEO survey found that organizations that were highly effective at integrating information were five times more likely to generate value than those who were poor at it. Critical business initiatives such as single view of a customer, business intelligence, supply chain management require consistent, complete, and trustworthy information. An information server helps companies to integrate information in order to deliver business results within these initiatives faster, with higher quality results.

- For business intelligence, it helps organizations develop a unified view of their business for better decisions by enabling them to understand existing data sources to cleanse, correct, and standardize information, and to load analytical views.

- For master data management, it helps organizations develop authoritative master data by enabling them to understand where and how information is stored across systems, and to consolidate disparate data into a single, reliable record.
- For infrastructure rationalization, it helps organizations reduce operating costs by allowing them to understand relationships between systems, and to define migration rules to consolidate instances or to move data from obsolete systems to new applications and databases.
- For business transformation initiatives, it helps organizations speed development and increase business agility by providing reusable information services that can be seamlessly plugged into applications, business processes, and portals. These standards-based services are maintained centrally by information specialists, with a single point of maintenance, but are widely accessible throughout the enterprise.
- For risk and compliance projects, it helps organizations improve visibility and data governance by allowing organizations to define and maintain complete, authoritative views of information with proof of lineage and quality. These views can be made widely available and reusable as shared services.

2.3 DISTRIBUTED DATABASES

A distributed database (DDB) is a collection of multiple, logically interrelated databases distributed over a computer network. It consists of loosely coupled sites that share no physical component. Database systems that run on each site are independent of each other and transactions may access data at one or more sites. [6]

A distributed database management system (distributed DBMS) is the software system that permits the management of the distributed database and makes the distribution transparent to the users [5].

2.3.1 HOMOGENEOUS DISTRIBUTED DATABASES

In a homogeneous distributed database all sites have identical software, are aware of each other and agree to cooperate in processing user requests, each site surrenders part of its autonomy in terms of right to change schemas or software, appears to user as a single system. [6]

2.3.2 HETEROGENEOUS DISTRIBUTED DATABASES

In a heterogeneous distributed database, different sites may use different schemas and software:

- Difference in schema is a major problem for query processing
- Difference in software is a major problem for transaction processing

Sites may not be aware of each other and may provide only limited facilities for cooperation in transaction processing.

2.4 DATA INTEGRITY FOR DATABASES AND INFORMATION SERVERS

Data integrity refers to the accuracy, consistency, and reliability of data that is stored in the database. [7]

2.4.1 TYPES OF DATA INTEGRITY

There are four types of data integrity:

Row integrity

Column integrity

Referential integrity

User-defined integrity

2.4.1.1 ROW INTEGRITY

Row integrity refers to the requirement that all rows in a table must have a unique identifier that can be used to tell apart each record. This unique identifier is normally known as Primary Key of the table. A Primary Key can be formed by a single column or a combination of multiple columns.

2.4.1.2 COLUMN INTEGRITY

Column integrity refers to the requirement that data stored in a column must adhere to the same format and definition. This includes data type, data length, default value of data, range of possible values, whether duplicate values are allowed, or whether null values are allowed.

2.4.1.3 REFERENTIAL INTEGRITY

Referential integrity is a database concept that ensures that relationships between tables remain consistent. When one table has a foreign key to another table, the concept of referential integrity states that you may not add a record to the table that contains the foreign key unless there is a corresponding record in the linked table. [8]
For referential integrity to hold in a relational database, any field in a table that is declared a foreign key can contain only values from a parent table's primary key or a candidate key. For instance, deleting a record that contains a value referred to by a foreign key in another table would break referential integrity.

2.4.1.4 USER-DEFINED INTEGRITY

Some applications have complex business logic that can't be enforced by defining criteria in the three data integrity types we have discussed so far (row integrity, column integrity, and referential integrity). In this circumstance, we need to implement our own code logic to make sure data is saved accurately and consistently across all business domains. The code logic can be implemented by using database triggers, stored procedures or functions, or by using tools external to the database engine such as embedding non SQL languages (like VBScript or C# in SQL Server) in the database, or by using scripting or programming languages in the middle-tier or front-tier of the application.

2.4.2 DATA INTEGRITY IS ENFORCED BY DATABASE CONSTRAINTS

Database Constraints are declarative integrity rules of defining table structures. They include the following 7 constraint types: [7]

Data type constraint:

This defines the type of data, data length, and a few other attributes which are specifically associated with the type of data in a column.

Default constraint:

This defines what value the column should use when no value has been supplied explicitly when inserting a record in the table.

Nullability constraint:

This defines that if a column is NOT NULL or allow NULL values to be stored in it.

Primary key constraint:

This is the unique identifier of the table. Each row must have a distinct value. The primary key can be either a sequentially incremented integer number or a natural selection of data that represents what is happening in the real world (e.g. Social Security Number). NULL values are not allowed in primary key values.

Unique constraint:

This defines that the values in a column must be unique and no duplicates should be stored.

Foreign key constraint:

This defines how referential integrity is enforced between two tables.

Check constraint:

This defines a validation rule for the data values in a column so it is a user-defined data integrity constraint. This rule is defined by the user when designing the column in a table. Not every database engine supports check constraints. As of version 5.0, MySQL does not support check constraint. But you can use enum data type or set data type to achieve some of its functionalities that are available in other Relational Database Management Systems (Oracle, SQL Server, etc).

2.5 MULTI AGENT SYSTEMS FOR DATA INTEGRITY MAINTENANCE

The autonomy and interactive characteristics allows agents with different capabilities to secure the data, which involve interacting with the user/sender, organizing the information obtained from the user, listening to any connection from other users (recipients) and applying cryptography protocols. Agents have the ability to interact, coordinate and cooperate with each other in order to achieve the overall goal of the system (which is to send secure message to the recipients). The extensible property of the agent allows it to be added or instantiated when a new communication is needed and deleted when the communication has ended. Thus, the agents can handle multiple communications at once. The mobility characteristic of the agent can be used to carry the message across the network, which allows the agent to carry out tasks on behalf of the user/sender.

Autonomous: The autonomy of an agent is a characteristic that allows agents to do the assigned tasks independently. Each agent has its own behaviour(s). The agent's actions or tasks assigned to the agent are performed from within the behaviours. Communications and coordination between agents, which are performed in order to complete a task, are done and handled through behaviours. When an agent is executed, it automatically performs its behavior without being invoked by any external entity. Agents are considered active. They have control on their actions as they have goals and rules. They know when to act, or update their states. Autonomous is performed by not providing the agent with call-backs function to its own object reference to other agents. Consequently, this will lessen any chance of other entities taking control of its services. Thus, control complexity is reduced and divided within the agents themselves. [1][11]
Mobility: Mobility is the ability of the agent to migrate or move from home platform to another platform, carrying its code and data. Mobile agents can be characterized as follows:

- Mobile agents are used in the wide-area and heterogeneous networks where there is no reliability on the connection or the security of the network.
- The migration of the agents is initiated by the agent (or programmer)
- The agents migrate to access resources only available at the remote hosts

Extendible: Extendible focuses on adding or removing capabilities or skills of an agent to an existing system. Author in [27] defines extensibility as "the abilities to easily add new functionality to a system, or upgrading any existing functionality". To be extensible, Multi Agent Systems should be capable of performing new functionality that it currently is not able to perform. A new agent representing a new system's functionality can be added to the system, without reconfiguring the whole system.

Interactive: The key element for multi-agent interaction and social organization is communication. Agents are able to cooperate and coordinate their actions to execute tasks through communications. Agent Communication Language (ACL) enables an agent to exchange data and information with other agents. FIPA-ACL and KQML are the most commonly used agent language. [26]

2.6 APPLICATION OF MULTI AGENT SYSTEMS IN THE HEALTH INDUSTRY

Multi Agent Systems have shown their potential in meeting critical needs in high-speed, mission-critical, content-rich, distributed information systems where mutual interdependencies, dynamic environments, uncertainty, and sophisticated control play a notable role [16]. Healthcare applications can take outstanding advantage of the intrinsic characteristics of multi-agent systems because of notable features that most healthcare applications share:

(i) they are composed of loosely coupled (complex) systems;
(ii) they are realized in terms of heterogeneous components and legacy systems;
(iii) they dynamically manage distributed data and resources; and
(iv) they are often accessed by remote users in (synchronous) collaboration [15][17].

In the last ten years, MAS have already been used for realizing e-health applications and, in particular, for realizing assistive living, diagnostic, physiological telemonitoring , smart hospital and smart emergency applications.
Some of the multi agent systems in the health industry include:
CASIS is an event-driven service-oriented and multi-agent system framework whose goal is to provide context-aware healthcare services to the elderly resident in the intelligent space. CASIS framework allows remote caretakers, such as concerned family members and healthcare providers, to closely monitor and attend to the elder's physical and mental well-beings anytime, anywhere. The smart environment interacts with the elder through a wide variety of appliances for data gathering and information presentation. The environment tracts the location and specific activities of the elder through sensors, such as pressure-sensitive floors, cameras, bio-sensors, and smart furniture. Meanwhile, the elder receives multimedia messages or content through speakers, TV, as well as personal mobile devices. The caregivers may access the elder's health and dietary information through any Web enable device like a PC or PDA. Context-aware computing enables the environment to respond at the right time and the right place, to the elder's needs based on the sensor data collected. The environment is further equipped with integrated control for convenience, comfort and safety. In particular, CASIS is able to infer the status of the elder and performs appropriate actions. For example, upon sensing that the elder has fallen asleep, it turns off the TV, and switches the telephone into voice mail mode. It informs and plays back any incoming messages when the elder wakes up [18].

IHKA is a healthcare knowledge procurement system based on the use of multi-agent technologies. This system is based on six different agent types. These types are the user interface agent, an agent to convert the search result into a viable format for
passing to the UI agent, a query optimizing agent which optimizes the query, the knowledge retrieval agent that performs the query, the knowledge adaptation agent to adapt the knowledge to the current circumstances and the knowledge procurement agent which if all else fails searches the web for the knowledge. In particular, IHKA features autonomous knowledge gathering, filtering, adaptation and acquisition from some healthcare

enterprise/organizational memories with the goal of providing assistance to non-expert healthcare practitioners [19].

HealthAgents is a research project with the goal of improving the classification of brain tumours through multi-agent decision support over a secure and distributed network of local databases. HealthAgents does not only develop new pattern recognition methods for distributed classification and analysis of in vivo MRS and ex vivo/in vitro HRMAS and DNA data, but it also defines a method to assess the quality and usability of a new candidate local database containing a set of new cases, based on a compatibility score. Using its multi-agent architecture, HealthAgents applies cutting-edge agent technology to the biomedical field and it provides an infrastructure for the so called HealthAgents network, a globally distributed information and knowledge repository for brain tumour diagnosis and prognosis [19][18].

2.7 APPLICATION OF MAS FOR INFORMATION SERVER INTEGRITY MAINTENANCE

Mobile agents have several strengths.
First, by migrating to the location of a needed resource, an agent can interact with the resource without transmitting intermediate data across the network, conserving bandwidth and reducing latencies. Similarly, by migrating to the location of a user, an agent can respond to user actions rapidly. In either case, the agent can continue its interaction with the resource or user even if network connections go down temporarily. These features make mobile agents particularly attractive in mobile-computing applications, which often must deal with low-bandwidth, high-latency, and unreliable network links.
Second, mobile agents allow clients and servers to offload work to each other, and to change who offloads to whom according to the capabilities and current loads of the client, server and network.
Although each of these strengths is a reasonable argument for mobile agents, it is important to realize that none of these strengths are unique to mobile agents [15]. Any specific application can be implemented just as efficiently with other techniques. These

other techniques include message passing, remote procedure calls (RPC), remote object-method invocation, stored procedures (where SQL procedures can be uploaded into a relational database for later invocation), Java applets and servlets etc.
The true strength of mobile agents is that a wide range of distributed applications can be implemented efficiently, easily and robustly within the same, general framework, and these applications can exhibit extremely flexible behavior in the face of changing network conditions.

2.7.1 PROPOSED ARCHITECTURE

For this study, we propose an architecture in which we will have 2 agents at the regulatory bodies, the *extraction agent* and the *mobile agent*, and a single agent at the ministries of health, the *information server agent*.

The extraction agent will extract the data from the regulatory body database and handover the data to the mobile agent. The mobile agent will transport the data to the ministries of health via the internet and handover the data to the information server. Finally, the information server agent will update the information server thus keeping it current.

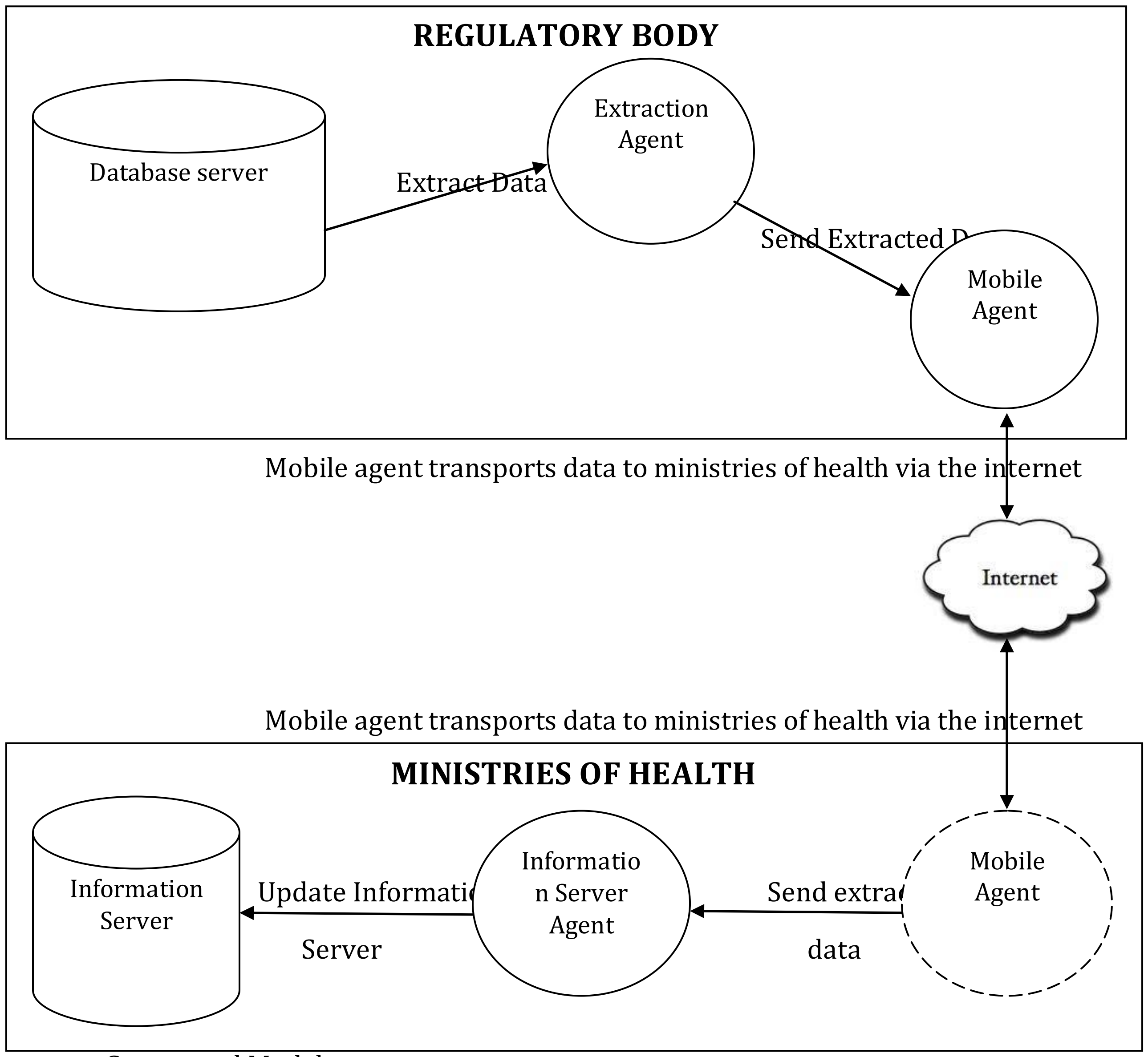


Figure 2-1: Conceptual Model

Refer to Figure 4-5 for the actual model which shows all entities and how they communicate.

2.8 SUMMARY

In this chapter, we have looked at the literature on the available technologies, their advantages and disadvantages and a review of related studies done in a similar domain. It is key to note that, other techniques that can do the same thing the mobile agents can do exist, like remote procedure calls, remote object-method invocation, servlets among others, but mobile agents is the preferred technology due to its efficiency and it can be implemented easily and robustly within the same general framework.

Also, for the agents, we decided to use the stateless approach as opposed to the stateful approach due to bandwidth usage. In a stateful mode, the agent embodies in its data state the results of each host execution and carries with it a growing collection of information to each subsequent host in its itinerary, therefore utilizing more bandwidth than the stateless approach based on our scenario.

The security of agents has been questioned, especially the mobile agents, but in our implementation, we have encrypted all the data in the mobile agent using AES (Advanced Encryption standard) algorithm, which is fast and very secure and has been adopted by the United States government [21]. It is noted by researchers from Microsoft and Belgian Katholieke Universiteit Leuven that it would take billions of years of computer time to break the AES algorithm [24]. We have also not included any crucial information that may compromise the security of the servers in the mobile agent. That information is held by the extraction agent and information server agent, for example the login credentials of the databases.

We decided to use the Jade platform because of its interoperability, uniformity, portability and ease of use.

3. METHODOLOGY

3.1 INTRODUCTION

Methodology is the systematic, theoretical analysis of the methods applied to a field of study, or the theoretical analysis of the body of methods and principles associated with a branch of knowledge. We selected the GAIA methodology because its characteristics best fit our scenario, as will be further explained in this chapter.

3.2 GAIA METHODOLOGY

Gaia is a Methodology which has been specifically tailored to the analysis and design of agent-based systems. Gaia is appropriate for the development of systems with the following main characteristics:

- Agents are coarse-grained computational systems, each making use of significant computational resources (think of each agent as having the resources of a Unix process).
- It is assumed that the goal is to obtain a system that maximizes some global quality measure, but which may be sub-optimal from the point of view of the system components. Gaia is not intended for systems that admit the possibility of true conflict.
- Agents are heterogeneous, in that different agents may be implemented using different programming languages, architectures, and techniques. We make no assumptions about the delivery platform;
- The organization structure of the system is static, in that inter-agent relationships do not change at run-time.
- The abilities of agents and the services they provide are static, in that they do not change at run-time.
- The overall system contains a comparatively small number of different agent types (less than 100).

3.2.1 GAIA'S MODEL

Gaia is intended to allow an analyst to go systematically from a statement of requirements to a design that is sufficiently detailed that it can be implemented directly. Analysis and design can be thought of as a process of developing increasingly detailed models of the system to be constructed [13].

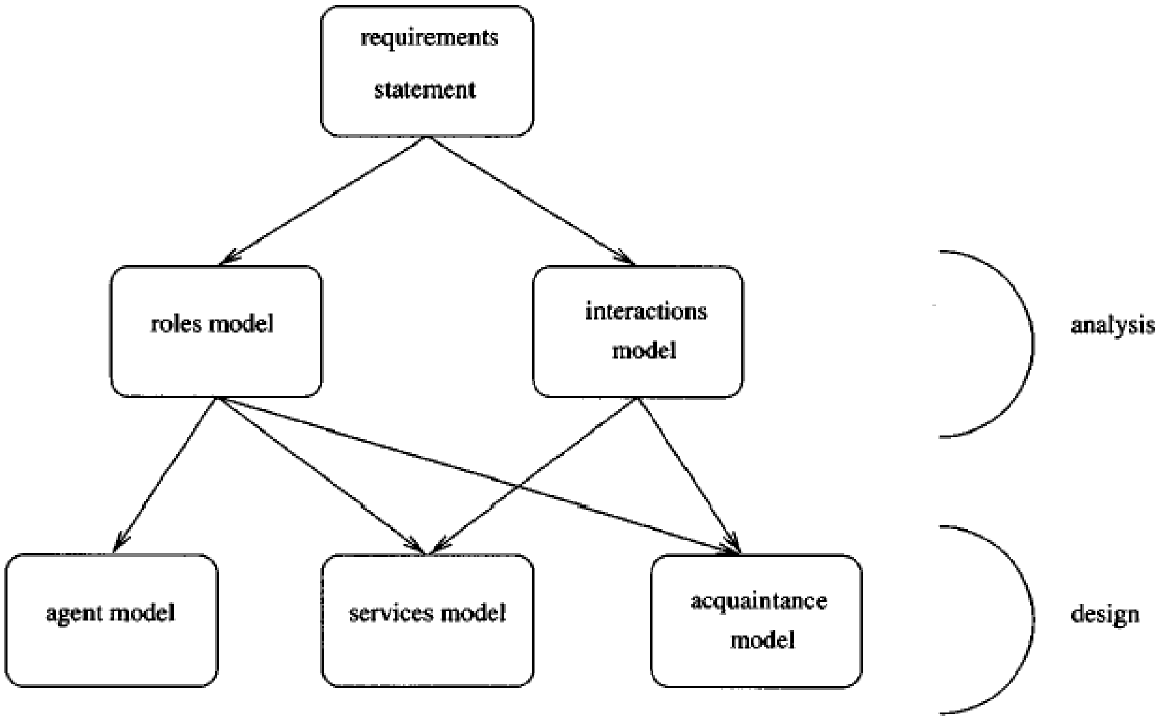


Figure 3-1: GAIAs Model

3.2.1.1 JUSTIFICATION OF USING GAIA METHODOLOGY

The GAIA methodology is used because of the following reasons which are all characteristics appropriate with the GAIA methodology:

- The organizational structure of this system is static, in that inter-agent relationships do not change at run-time.
- The abilities of agents and the services they provide are static, in that they do not change at run-time.

- In this system, the agents are heterogeneous, in that different agents can be implemented using different programming languages and techniques.

- The overall system contains a comparatively small number of different agent types (less than 100).
- The goal is to obtain a system that maximizes some global quality measure, but which may be sub-optimal from the point of view of the system components.

3.2.1.2 LIMITATIONS OF GAIA METHODOLOGY

- Gaia cannot explicitly model and represent important social aspects of a multi-agent system. Among them, Gaia cannot explicitly model the organizational structure of the agents in the system, or alternatively, the architecture of the system with merely non-recursive roles. It also lacks the ability to explicitly model the social goals, social tasks or organizational rules within an organization of agents. These factors, together with the fact that Gaia implicitly assumes all agents to be cooperative, make it clearly unsuitable for open agents systems.

- Gaia employs Gaia-specific notations for representing roles and protocols. Also, these notations although simple and easy to catch may be somewhat poor for expressing complex problems such as, e.g., complex multi-phase interaction protocols.

- Does not directly deal with particular modeling techniques. It proposes but does not commit to, specific techniques for modeling (e.g., roles, environment and interactions). In the future: "… AUML is a useful companion to GAIA."

- Does not directly deal with implementation issues. The outcome is a detailed but technology-neutral specification. Should be easy to implement with, for example, a FIPA-compliant agent system.

- Does not explicitly deal with activities of requirements capturing and modeling. In the future: "… integrate methods and techniques from goal-oriented analysis."

3.3 METHODS FOR DATA COLLECTION AND ANALYSIS

In the study, we made use of both primary and secondary sources of data. Primary sources of data included questionnaires and observation, while secondary sources were input of data into the prototype to demonstrate the working of the system.

Questionnaires

Questionnaires will be issued out to staff who use the current system so as to determine some key aspects which will be compared against the prototype output to determine the systems efficiency.

Observation

Observation will be one of the data collection methods we will use in this study. After the predefined time period elapses, observing the data at the information server and comparing it to the data at the regulatory bodies' databases will play a very big role in determining the success of the system.

Data Analysis

For analyzing the data in both databases, we will use the use of SQL queries to ensure that the data in both databases is the same. Since the main goal of the system is ensuring that the information server is updated with the most recent data from the databases at the regulatory bodies, analyzing the data from the information server and the databases will be critical to ensure that the data is the same.
A comparison will also be made between data collected from questionnaires and data retrieved from prototype so as to compare the security, efficiency and speed of current system and proposed system.

3.4 PROTOTYPE IMPLEMENTATION AND TESTING

The implementation of the prototype system will be done on two computers (one representing the server at the regulatory bodies and another representing the information server at the ministries of health). The two computers will be connected via a network cable.

To enable testing of the system, java runtime environment and the jade platform has to be installed on both computers. The jade user interface is started on both computers, on the regulatory side; the regulatory and mobile agents are started. On the information server side; the information server agent is started. Data is then entered, edited or deleted on the regulatory side. After set time elapses (30 seconds in our case), changes made are reflected on the information server.

3.5 SUMMARY

In this chapter we have looked at the GAIA methodology which is a methodology which has been specifically tailored to the analysis and design of agent based systems. We settled on GAIA methodology because it's appropriate for development of systems where:

- The organization structure of the system is static, which is the case in our scenario.
- The abilities of the agents and the services they provide are static, which is the case in our scenario.
- The overall system contains a comparatively small number of different agent types; we only have 3 different agents in our scenario.
- It is assumed that the goal is to obtain a system that maximizes some global quality measure, but which may be sub-optimal from the point of view of the system components, which is the case in our scenario.

Based on these characteristics, we felt that the GAIA methodology best suits our scenario.

4. SYSTEM ANALYSIS AND DESIGN

4.1 OVERVIEW

The analysis and design of the system was guided by the GAIA methodology which has been discussed in chapter 3. In this chapter we will look at how the methodology was used in analyzing and designing the system. As outlined by the GAIA Methodology, we will look at three major steps which are requirements statements, analysis and design of the system, as shown in figure 3-1.

4.2 SYSTEM REQUIREMENTS STATEMENTS

- Officers at the regulatory bodies will insert new records, update records and delete records on the database server in their respective regulatory bodies.
- Regulatory side Agents will be activated after the elapsing of a predefined period of time.
- Regulatory side agents have rights and permissions to extract data from their respective databases.
- Regulatory side agents will reside only in their respective regulatory bodies' servers (immobile).
- Regulatory side agents will hand-over the extracted data to the mobile agents which will transport the data to the information server at the Ministries of Health.
- On reaching the information server, the mobile agents will hand-over the data to the extraction agent at the information server which will update the information server.
- Agents at the Information server reside at the information server (immobile).
- A user can be able to manually activate the agents.

4.3 DESIGN DIAGRAMS

4.3.1 USE CASE DIAGRAM AND SCENARIO

The data entry clerks at the regulatory bodies are responsible for entering, updating and deleting data from the regulatory bodies' databases. When that is done, the regulatory body agent is responsible for extracting the changes made by the data clerks from the regulatory bodies' databases and sending those changes to the mobile agent. The mobile agent transports those changes to the information server agent at the ministries of health which in turn updates the information server at the ministries of health with the changes, thus keeping the information server up to date. Once the information server is updated, the health managers can be able to make accurate decisions.

This interaction is illustrated in figures 4-1, 4-2, 4-3 and 4-4 below.

Figure 4-1 illustrated how officers in the regulatory bodies will be entering new data, updating existing data and deleting data from the database server in their respective regulatory bodies. Other officers will just be retrieving the data.

Figure 4-2 illustrates how the extraction agent extracts data and hands it over to the mobile agent. The mobile agent then transports the data to the MOH.

Figure 4-3 illustrates how the health managers retrieve data from the information server at the MOH.

Figure 4-4 illustrated how the mobile agent hands over the data to the information server agent. The information server agent updates the information server, then hands the mobile agent an identifier to keep track of the update progress.

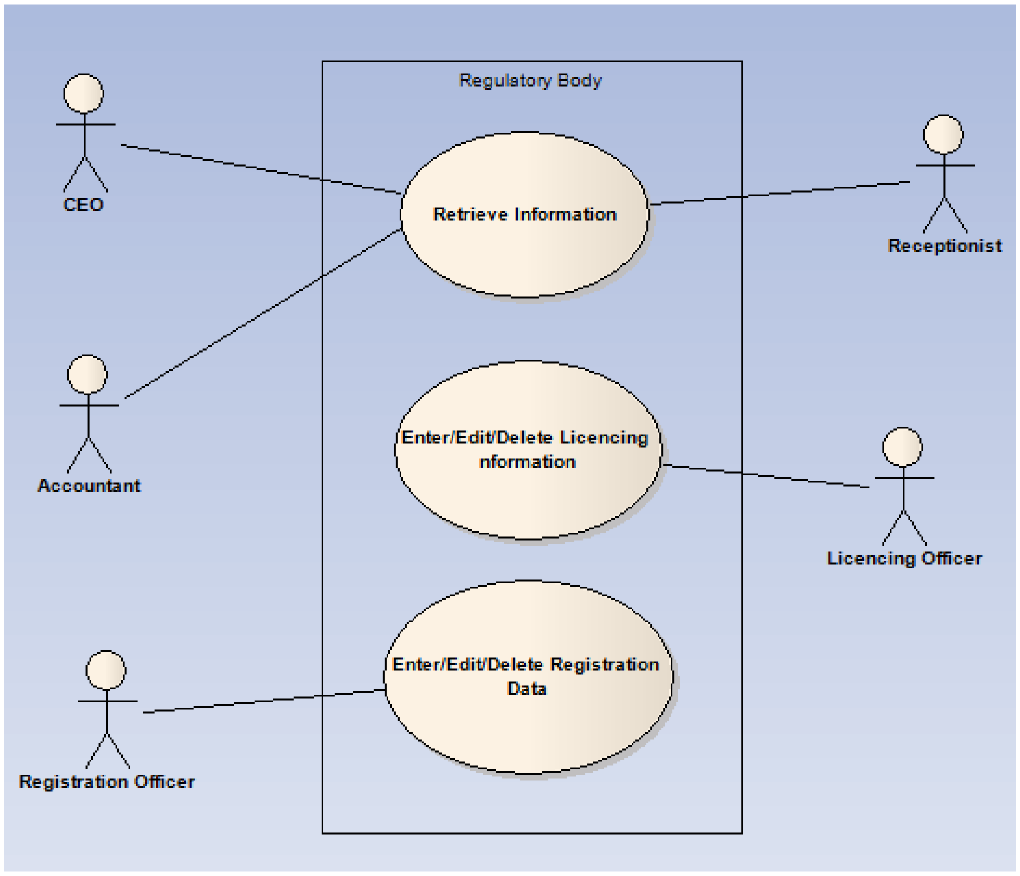


Figure 4-1: Use case Diagram for Regulatory body – staff interaction

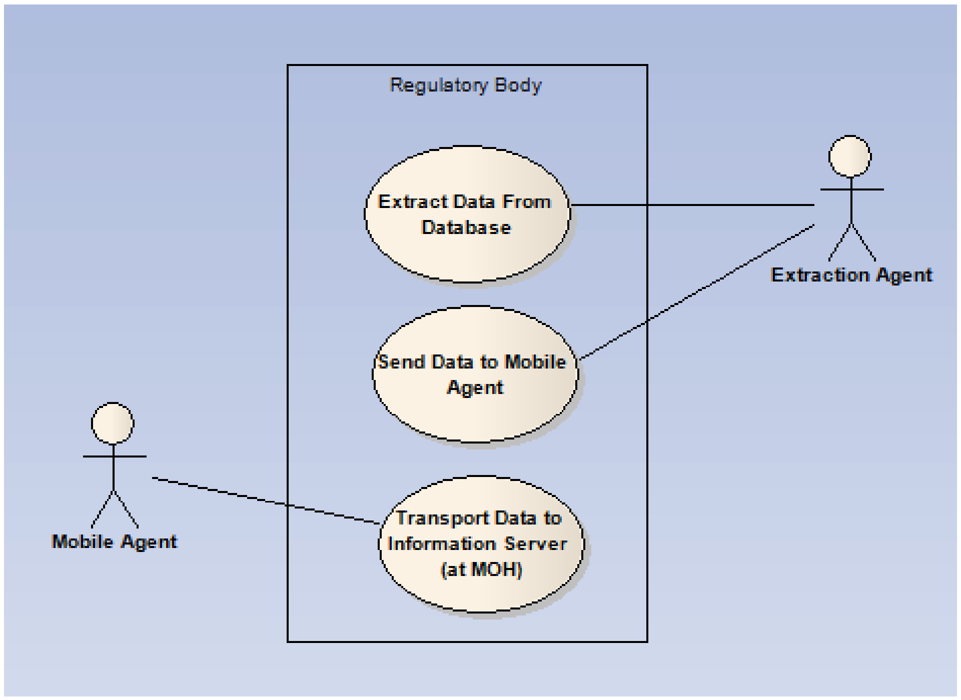

Regulatory Body
Extract Data From Database
Extraction Agent
Send Data to Mobile Agent
Mobile Agent
Transport Data to Information Server (at MOH)

Figure 4-2: Use case Diagram for Regulatory Body – Agent Interaction

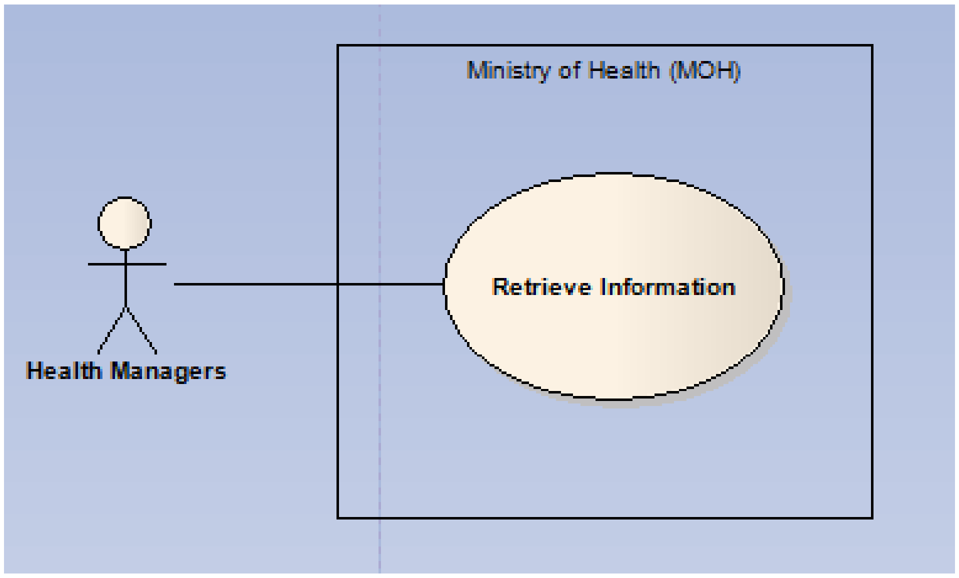


Figure 4-3: Use case Diagram for MOH – Health managers' interaction

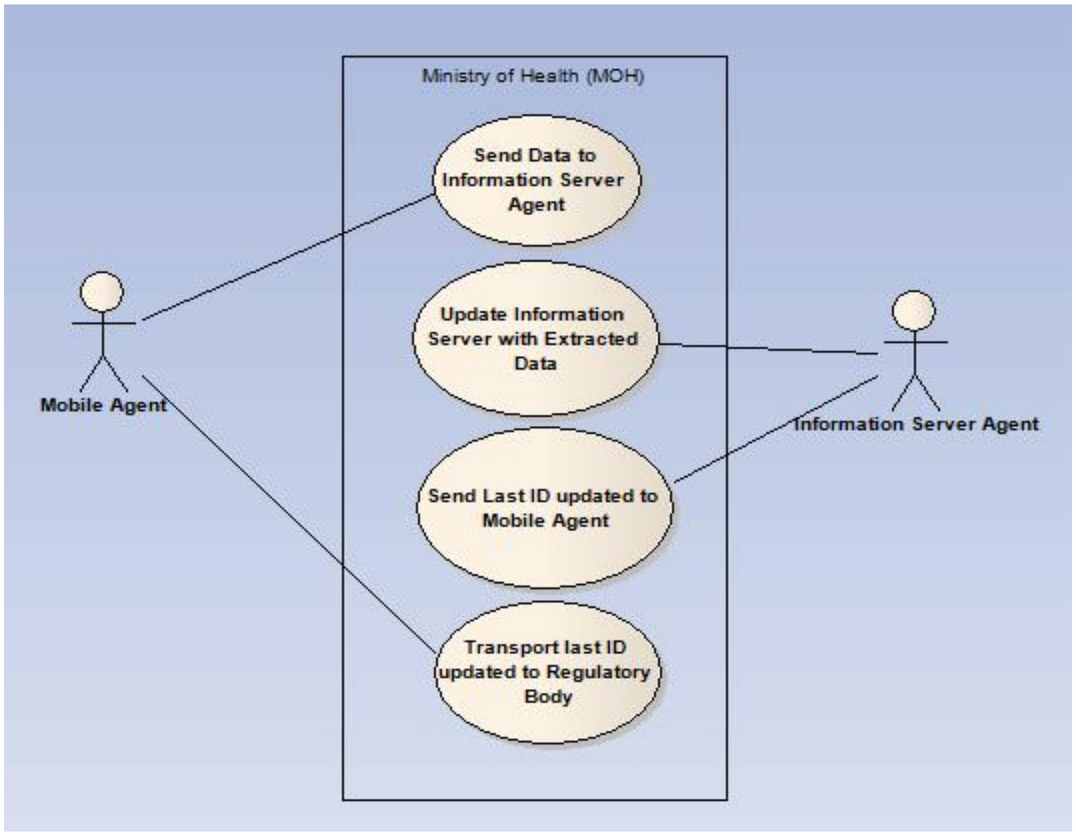

Ministry of Health (MOH)
Send Data to Information Server Agent
Update Information Server with Extracted Data
Send Last ID updated to Mobile Agent
Transport last ID updated to Regulatory Body
Mobile Agent
Information Server Agent

Figure 4-4: Use case Diagram for MOH – agent interaction

4.3.2 ARCHITECTURAL DESIGN OF THE SYSTEM

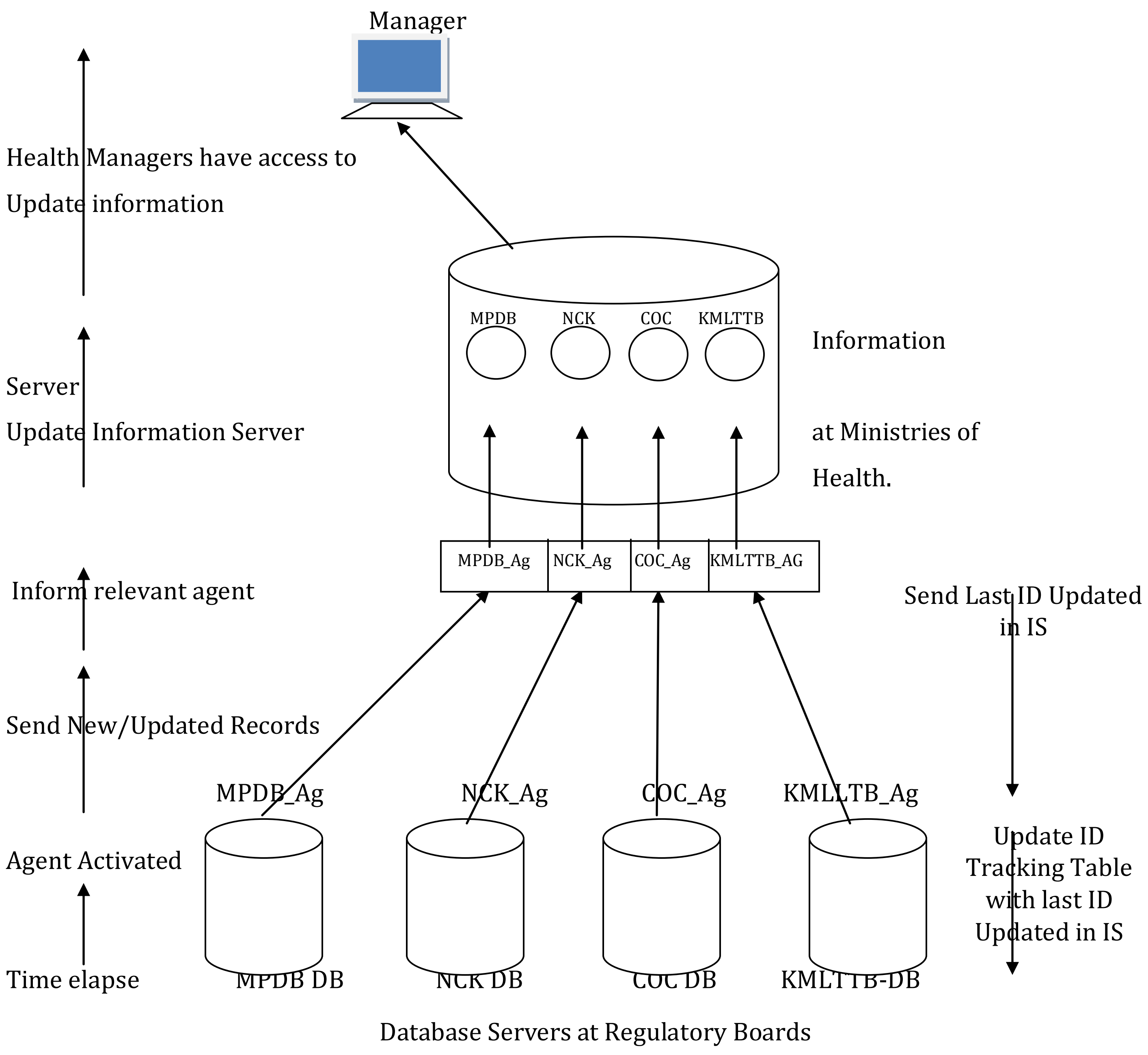


Figure 4-5: Architectural Design of System

Figure 4-5 illustrates how all entities will communicate with each other to ensure smooth running of the MAS. Each regulatory body will have its own extraction agent and mobile agent, while at the information server there will be an information server agent for each regulatory body. This will enhance the performance of individual information server agents while maintaining separation of data of different regulatory bodies.

4.3.3 PROCESS FLOW DIAGRAM

The process flow diagram shown below illustrates the decision process of the agents in the system.

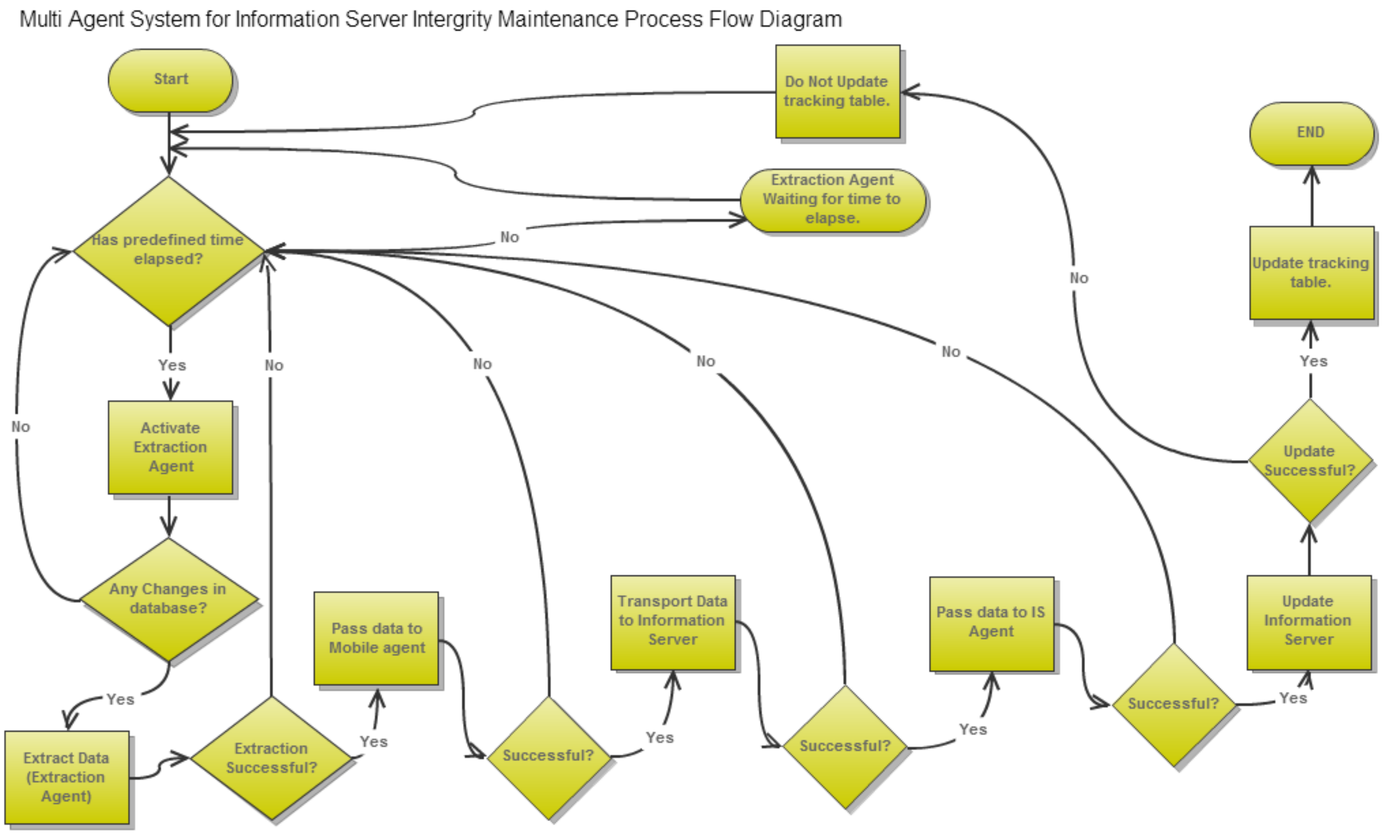

Multi Agent System for Information Server Intergrity Maintenance Process Flow Diagram
Start
Do Not Update tracking table.
Extraction Agent Waiting for time to elapse.
END
Has predefined time elapsed?
No
Yes
Activate Extraction Agent
Any Changes in database?
No
Yes
Extract Data (Extraction Agent)
Extraction Successful?
No
Yes
Pass data to Mobile agent
Successful?
No
Yes
Transport Data to Information Server
Successful?
No
Yes
Pass data to IS Agent
Successful?
No
Yes
Update Information Server
Update Successful?
No
Yes
Update tracking table.

Figure 4-6 : Process Flow Diagram

Figure 4-6 illustrates the process flow of the entire process from extraction of data by the extraction agent at the regulatory bodies, handing over of extracted data to the mobile agent, transporting of data from the regulatory body to the information server by the mobile agent, handing over of extracted data to the information server agent at the information server, updating of the information server, mobile agent moving back to the regulatory body with a tracking identifier to keep track of the update process.

4.4 THE ANALYSIS PHASE

The objective of the analysis stage is to develop an understanding of the system and its structure (without reference to any implementation detail). This understanding is captured in the system's organization. An organization can be seen as a collection of roles that stand in certain relationships to one another, and that take part in systematic, institutionalized patterns of interactions with other roles

4.4.1 ROLES MODEL

The roles model identifies the key roles in the system. Here a role can be viewed as an abstract description of an entity's expected function. Such roles are characterized by two types of attribute:

- The permissions/rights associated with the role.
- The responsibilities of the role.

In our model, the analysis phase has led to the identification of three roles:

- ExtractRole which extracts the changes that have taken place in the databases at the regulatory bodies.
- TransportRole which deals with transporting data from regulatory bodies to information server.
- UpdateRole which updates the data to the information server.

Role: ExtractRole

Description: It waits for a predetermined time period to elapse so that it can query the database server and extract records which are either new or have been edited since the last time it did the same. It then hands-over this data to the mobile agent.

Protocols and Activities: Query Database Server, Transfer data to the mobile agents.

Permissions: read database server at regulatory bodies.

Responsibilities: Liveness: ExtractRole = (extractData. InformMobileAgent. AwaitTimeExpire)$^{\omega}$

Safety: A successful connection with the database server is established.

Successful communication with the mobile agents.
Role: TransportRole **Description:** It receives the data extracted in the ExtractRole from the regulatory body agents and transports it to the Information Server. It must therefore have the capability to communicate with the control agent at the information server and the agents at the regulatory bodies. **Protocols and Activities:** Receive data from agents at the regulatory bodies, transport data to the information server, hand-over data to the control agent at the information server. **Permissions:** communicate with agents at the regulatory bodies and the control agent at the information server. **Responsibilities: Liveness:** TransportRole = getDataEA. transportDataIS **Safety:** A successful communication with agents at the regulatory bodies. Successful communication with control agent at IS Successful transportation of data to the IS
Role: UpdateRole **Description:** It receives the data extracted in the ExtractRole by the regulatory body agents and transported to the Information Server by the mobile agents. It must therefore have the capability to communicate with the mobile agent. It updates the data it receives to the information server to keep it up to date. **Protocols and Activities:** Receive data from the mobile agent, updates the information server with received data. **Permissions:** communicate with the mobile agent, read and write to the information server. **Responsibilities: Liveness:** UpdateRole = getDataMA. updateIS **Safety:** Successful communication with the mobile agent Successfully updating the information server with the received data.

Table 4-1: Roles Model defining all the roles that will be required in the multi-agent system and which agent will be responsible for each role.

4.4.2 THE INTERACTIONS MODEL

In Gaia, links between roles are represented in the interaction model. This model consists of a set of protocol definitions, one for each type of inter-role interaction. Here a protocol can be viewed as an institutionalized pattern of interaction. That is, a pattern of interaction that has been formally defined and abstracted away from any particular sequence of execution steps. Viewing interactions in this way means that attention is focused on the essential nature and purpose of the interaction, rather than on the precise ordering of particular message exchanges

	Extract Data	**Transport Data**	**Update Data**	**TrackID**
Initiator	Elapse of predetermined time	Successful completion of ExtractRole	Successful arrival of mobileAgent at Information Server.	Successful completion of Update Role.
Receiver	extractAgent	mobileAgent	Information Server	mobileAgent
Responding Action	ExtractRole	TransportRole	UpdateRole	IDTrackingRole
Purpose/ parameters	Activates the extractAgent.	Activates the mobileAgent.	Activates the updateAgent	Activates mobileAgent

Table 4-2: Interactions Model shows the links between different roles, who/what initiates the role, receiver and responding actions of the role and the purpose of the role.

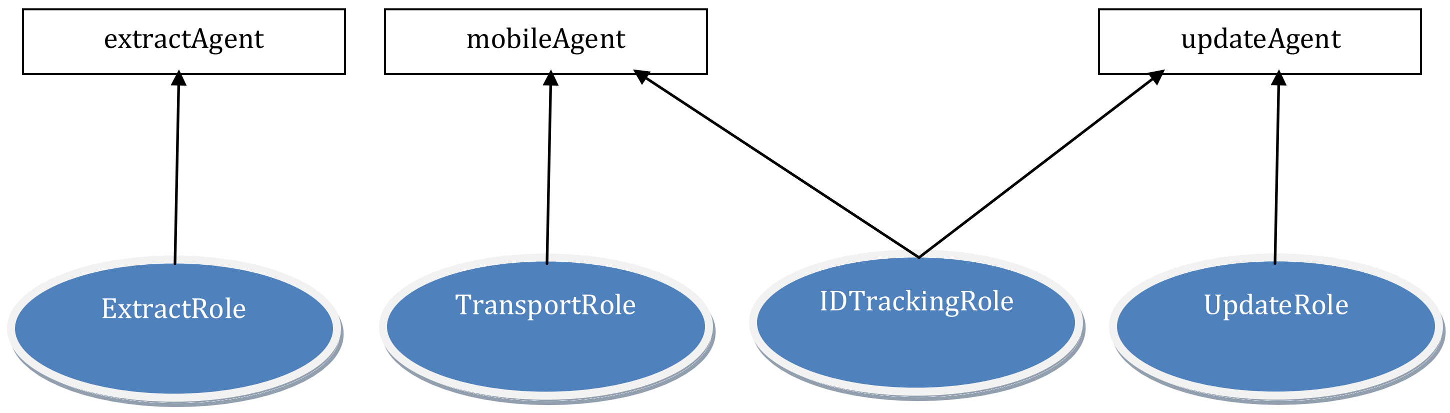
4.5 THE DESIGN PHASE

The aim in Gaia is to transform the analysis models into a sufficiently low level of abstraction that traditional design techniques (including object-oriented techniques) may be applied in order to implement agents. To put it another way, Gaia is concerned with how a society of agents cooperate to realize the system-level goals, and what is required of each individual agent in order to do this. The Gaia design process involves generating three models:

1. The agent model identifies the agent types that will make up the system, and the agent instances that will be instantiated from these types.
2. The services model identifies the main services that are required to realize the agent's role.
3. The acquaintance model documents the lines of communication between the different agents.

4.5.1 THE AGENT MODEL

The purpose of the Gaia agent model is to document the various agent types that will be used in the system under development, and the agent instances that will realize these agent types at run-time. An agent type is best thought of as a set of agent roles.



LEGEND

: Role

: Agent Type

Figure 4-7: Agent Model – documents various agent types and the agent instances that will realize these agent types at run-time

4.5.2 THE SERVICES MODEL

The aim of the Gaia services model is to identify the services associated with each agent role, and to specify the main properties of these services. By a service, GAIA means a function of the agent. A service is a coherent block of activity in which an agent will engage. It should be clear that every activity identified at the analysis stage will correspond to a service, though not every service will correspond to an activity. For each service that may be performed by an agent, it is necessary to document its properties. Specifically, we must identify the inputs, outputs, pre-conditions, and post-conditions of each service.

Service	**Extract data**	**Transport data**	**Update data**	**TrackID**
Inputs	-	Extracted data by extractAgent.	Extracted data by extractAgent.	Last ID updated by updateAgent.
Outputs	New or updated data since the last extraction.	Arrival of extracted data at the information server.	Data updated to the Information Server.	Last id updated to the tracking id table at the regulatory body.
Pre-condition	Elapse of time.	Data handed over to mobileAgent by extractAgent	Data handed over to updateAgent by controlAgent.	Successful execution of the update role.
Post-condition	Handover data to mobileAgent.	Data handed over to controlAgent	Data updated to the Information Server	Last id updated to the tracking id table at the regulatory body.

Table 4-3: Services Model - identify the services associated with each agent role, and to specify the main properties of these services.

4.5.3 THE ACQUAINTANCE MODEL

The final Gaia design model is the acquaintance model. Acquaintance models simply define the communication links that exist between agent types. They do not define what messages are sent or when messages are sent, they simply indicate that communication pathways exist. In particular, the purpose of an acquaintance model is to identify any potential communication bottlenecks, which may cause problems at run-time.
For this model we have slightly modified the original definition presented in Gaia.

	extractAgent	**mobileAgent**	**updateAgent**
extractAgent		**I**	
mobileAgent	**I**		**I**
updateAgent		**I**	
Legend: I: Interacts (e.g. the extractAgent interacts with the mobileAgent).			

Table 4-4: Acquaintances Model – communication links that exist between agent types.

5. SYSTEM IMPLEMENTATION

5.1 OVERVIEW

The prototype will be implemented using JAVA platform and JADE. Java Platform will be the main environment where the agents will operate. JADE will be used for developing the agents. The implementation of the system will involve development of three agents:

- An Extraction Agent which will be based at the Regulatory bodies.
- A mobile agent which will transport data from regulatory body to information server at ministries of health.
- An Information server agent based at the information server at the ministries of health which will be responsible for updating the information server with the most current data from the mobile agent.

Storage of data will be done on MySQL Databases.

5.2 RESOURCES NEEDED

5.2.1 SOFTWARE

- JAVA Runtime Environment
- JADE (Java Agent Development Framework)
- Relational Database Management Systems (MySQL Database)

5.2.2 HARDWARE

- Network cables to interconnect the computers.
- 2 Computers (1 computer to represent the regulatory bodies, 1 computer to represent the Information Server).

5.2.3 JADE OVERVIEW

JADE is the middleware developed by TILAB for the development of distributed multi-agent applications based on the peer-to-peer communication architecture [23]. The intelligence, the initiative, the information, the resources and the control can be fully distributed on mobile terminals as well as on computers in the fixed network. The environment can evolve dynamically with peers (that in JADE are called agents) that appear and disappear in the system according to the needs and the requirements of the application environment. Communication between the peers, regardless of whether they are running in the wireless or wire line network, is completely symmetric with each peer being able to play both the initiator and the responder role.

JADE is fully developed in Java and is based on the following driving principles:

- Interoperability – JADE is compliant with the FIPA specifications. As a consequence, JADE agents can interoperate with other agents, provided that they comply with the same standard.
- Uniformity and portability – JADE provides a homogeneous set of APIs that are independent from the underlying network and Java version. More in details, the JADE run-time provides the same APIs both for the J2EE, J2SE and J2ME environment. In theory, application developers could decide the Java run-time environment at deploy-time.
- Easy to use – The complexity of the middleware is hidden behind a simple and intuitive set of APIs.
- Pay-as-you-go philosophy – Programmers do not need to use all the features provided by the middleware.

Features that are not used do not require programmers to know anything about them, neither add any computational overhead.

5.3 IMPLEMENTATION

In Figure 4-5, Architectural design of System, we have described how all the agents will communicate with each other. At the regulatory bodies, we will have the *extraction agent* and the *mobile agent* residing there. Therefore, we will have JADE environment set up at the regulatory bodies where the agents will operate from. We will also have the database server installed at the regulatory bodies which will store the data.

At the MOH, we will have the *information server* agent residing there; therefore we will have the JADE environment set up at the MOH. We will also have the information server installed at the MOH, where the data from the regulatory bodies will be updated.

The MOH and the regulatory bodies will be connected via the Internet, which the *mobile agent* will use as the transport media from the regulatory bodies to the MOH and back.

5.3.1 DEVELOPING THE EXTRACTION AGENT

After the development environment is set up, as shown in Appendix III, we can now start developing the agents. The first agent is the extraction agent. The work of the extraction agent is to extract data from the database servers at the regulatory bodies and handover the data to the mobile agent.

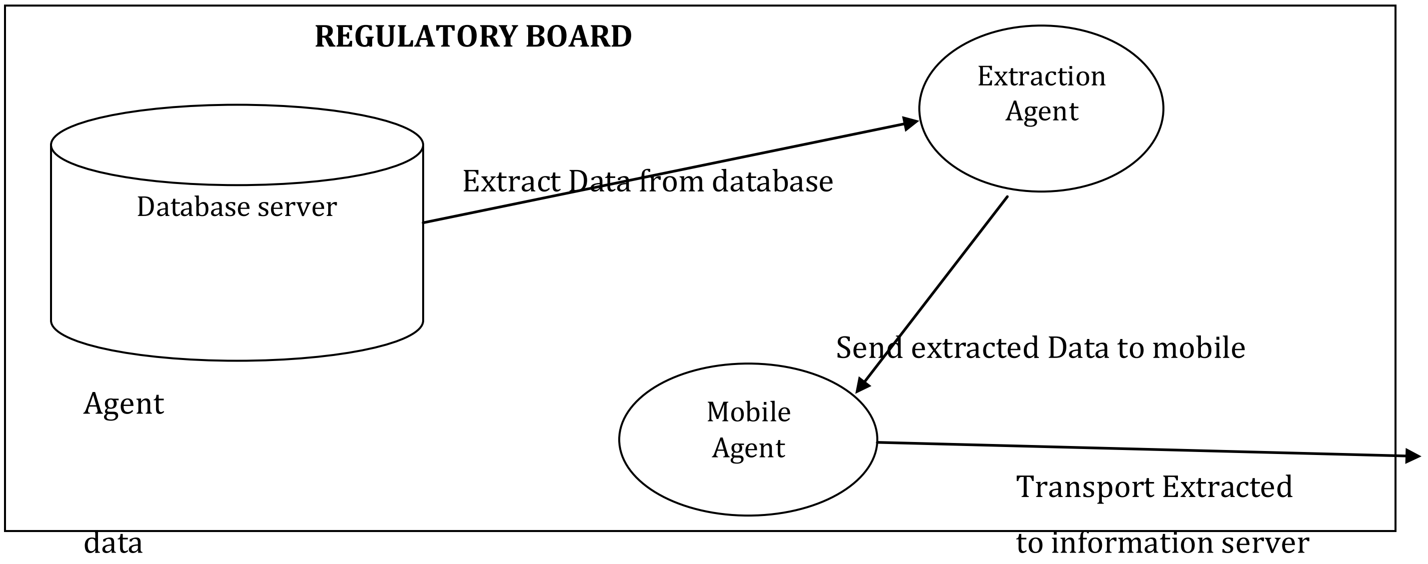


Figure 5-1: High level design of extraction agent functions at the regulatory bodies

The extraction agent was developed in such a way that it is activated after a predefined amount of time depending on how often the data changes and how often new data is entered into the database so as to maximize on bandwidth usage. In our research we set the predefined time to 30 seconds, to mean that the extraction agent checks the database server for changes made after every 30 seconds as shown in the code snippet below.

```
addBehaviour(new TickerBehaviour(this, 30000) {
```

When the extraction agent executes and finds changes to the database;

- It checks to find out if the change is an update, insert or deletion.
- It scripts an SQL query depending on its findings.
- It encrypts the scripts using AES (Advanced Encryption Standard) and stores all the scripts in an array.
- It finally sends the encrypted scripts to the mobile agent.

Below is a code snippet of the encryption algorithm and key used.

```
private static final String ALGO = "AES";

private static final byte[] keyValue = new byte[] { 'J', 'O', 'H', 'N', 'm', 'u', 'r', 'i', 't', 'u',
'M','A', 'C', 'H', 'A', 'A' };

public static String encrypt(String Data) throws Exception {

    Key key = generateKey();

    Cipher c = Cipher.getInstance(ALGO);

    c.init(Cipher.ENCRYPT_MODE, key);

    byte[] encVal = c.doFinal(Data.getBytes());

    String encryptedValue = new BASE64Encoder().encode(encVal);

    return encryptedValue;
    }
```

Below is a code snippet of how the agent scripts an SQL query, encrypts it and stores it in an array for an insert.

```
if(rs.getString("activity").equals("ins")){

System.out.println("Activity =" + rs.getString("activity"));

table[i] = "INSERT INTO test_table_dup values('" + rs.getString("new_id") +"','" +
rs.getString("new_surname")  +  "','" + rs.getString("new_fname") +"','" +
rs.getString("new_mname") + "','" + rs.getString("new_gender") + "')";

encrypted[i] = ExtractionAgent.encrypt(table[i]);

System.out.println(encrypted[i]);

}
```

Below is a code snippet of the extraction agent sending the array to the mobile agent.

```
    ACLMessage msg = new ACLMessage(ACLMessage.REQUEST);

    msg.addReceiver(new AID("mobile", AID.ISLOCALNAME));

    msg.setLanguage("English");

    msg.setOntology("Data-Integrity-ontology");

    try{

        msg.setContentObject(encrypted);

        }catch(IOException e){}
```

```
send(msg);
```

5.3.2 DEVELOPING THE MOBILE AGENT

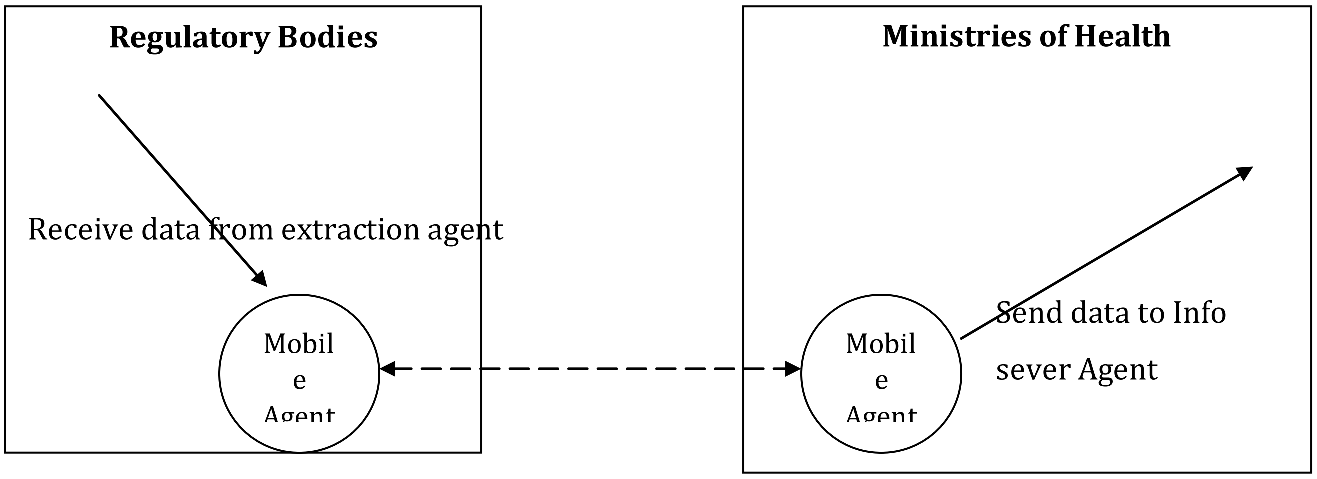

Regulatory Bodies
Ministries of Health
Receive data from extraction agent
Mobile Agent
Mobile Agent
Send data to Info sever Agent

FIGURE 5-2: MOVEMENT AND FUNCTIONS OF MOBILE AGENT IN THE REGULATORY BODIES AND MINISTRIES OF HEALTH

The mobile agent is responsible for:

- Receiving data from the extraction agent at the regulatory bodies.
- Moving to the ministries of health via network media.
- Handover the extracted data to the information server agent based at the ministries of health.
- Go back to the regulatory body via the network media.

Below is code snippet of how the mobile agent receives data from extraction agent and moves to the jade platform at the ministries of health.

```
msg = receive();

AID remoteAMS = new AID("amm@192.168.1.2:1099/JADE", AID.ISGUID);
```

```
remoteAMS.addAddresses("http://john-THINK:7778/acc");
PlatformID destination = new PlatformID(remoteAMS);
if (msg!=null){
        try{
                d = (String[])msg.getContentObject();
                }catch(UnreadableException e){}
                if(d.length != 0)
                {
                        doMove(destination);
                }
        }
```

Below is the code snippet of the mobile agent sending the extracted data to the information server agent at the ministries of health.

```
if(check.equals("infoserver")){
        ACLMessage msg1 = new ACLMessage(ACLMessage.REQUEST);
        msg1.addReceiver(new AID("afya@192.168.1.2:1099/JADE",
AID.ISGUID));
        try{
                msg1.setContentObject(d);
        }
        catch(IOException e){}
```

```
                send(msg1);
```

Below is the code snippet of the mobile agent moving back to the regulatory body after sending the extracted data to the information server agent.

```
                AID remoteAMS = new AID("amm@192.168.1.1:1099/JADE",
            AID.ISGUID);

                remoteAMS.addAddresses("http://john-PC:7778/acc");

                PlatformID destination1 = new PlatformID(remoteAMS);

                check = "regulatory";

                doMove(destination1);

            }
```

5.3.3 DEVELOPING THE INFORMATION SERVER AGENT

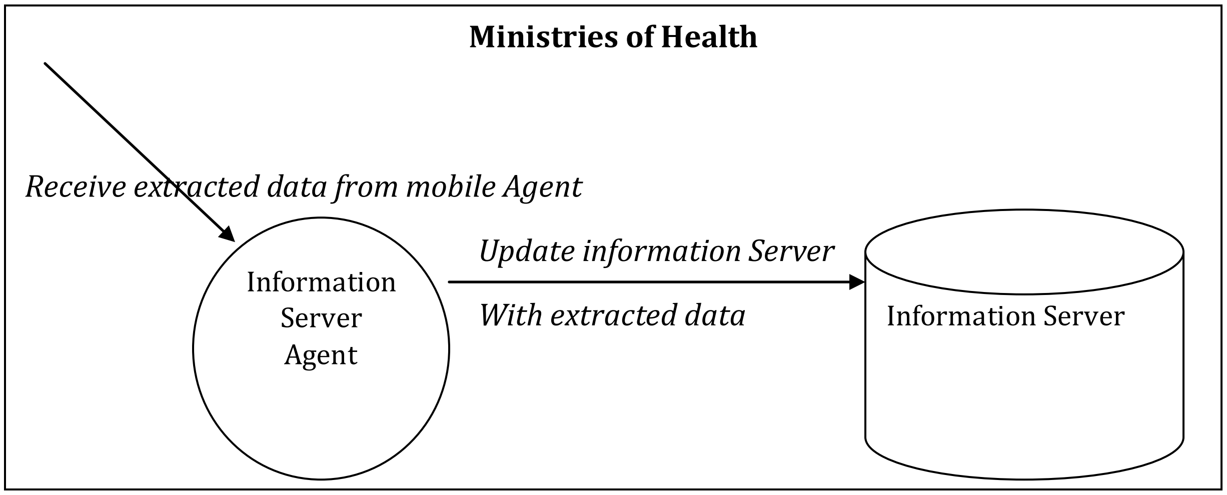


Figure 5-3: Roles and functions of the information server agent at the ministries of health

The information server agent is based at the information server in the ministries of health and is responsible for:

- Receiving extracted data from the mobile agent.

- Updating the information server with the extracted data it receives from the mobile agent.

When the information server receives data from the mobile agent;

- It decrypts the data it receives and stores it in an array.
- It updates the information server with the decrypted data.

Below is the decryption algorithm that the information server uses to decrypt the data.

```
private static final String ALGO = "AES";

private static final byte[] keyValue = new byte[] { 'J', 'O', 'H', 'N', 'm', 'u', 'r', 'i', 't',
'u', 'M','A', 'C', 'H', 'A', 'A' };

public static String decrypt(String encryptedData) throws Exception {

    Key key = generateKey();

    Cipher c = Cipher.getInstance(ALGO);

    c.init(Cipher.DECRYPT_MODE, key);

    byte[] decordedValue = new
BASE64Decoder().decodeBuffer(encryptedData);

    byte[] decValue = c.doFinal(decordedValue);

    String decryptedValue = new String(decValue);

    return decryptedValue;

  }
```

Below is the code the information server uses to decrypt the data, store the decrypted data in an array and finally update the information server.

```
Class.forName("com.mysql.jdbc.Driver");
```

```
Connection con = DriverManager.getConnection (dbUrl,"root","");
Statement stmt = con.createStatement();
for(int i = 0; i <= dLength; i++){
        decrypted[i] = InfoServerAgent.decrypt(d[i]);
        stmt.executeUpdate(decrypted[i]);
}
```

5.4 RUNNING THE SYSTEM

Before we can start running the system, we first ensure that the servers at the regulatory bodies and the information server at the ministries of health are connected via a network media for example fibre optic cable. We also need to ensure that both databases are running (turned on).

Turn on the *information server* agent at the MOH, and then turn on the *mobile agents* and *extraction agents* at the regulatory bodies. The reason we turn on the information server agent first is to reduce the risk of the *mobile agent* moving to the MOH information server and not finding the *information server* agent. If that was to happen, the information server would not be updated.

After all agents have been successfully started, the officers at the regulatory bodies can proceed with their normal duties and any changes they make to the data in their databases will automatically be updated in the information server at the MOH by the agents.

Refer to Appendix III for a more detailed explanation on how to run the system.

6 RESULTS AND ANALYSIS

6.1 OVERVIEW

In this chapter we will have a look at the results obtained from the questionnaires that were handed out. Ten questionnaires were handed out but only 9 questionnaires were filled. Then we will compare the results of the existing system to the multi-agent system we developed with a bias to *security and confidentiality* and *speed* of the systems.

6.2 RESULTS OBTAINED FROM THE QUESTIONNAIRES

To gather information from the field personnel who use the current system, we handed out ten (10) questionnaires which concentrated on;

- Security and confidentiality of the current system,
- How fast the current system is in updating the information at the ministries of health.

6.2.1 SECURITY AND CONFIDENTIALITY

Under the section of security and confidentiality, we asked a number of questions which are illustrated below in graphical form.

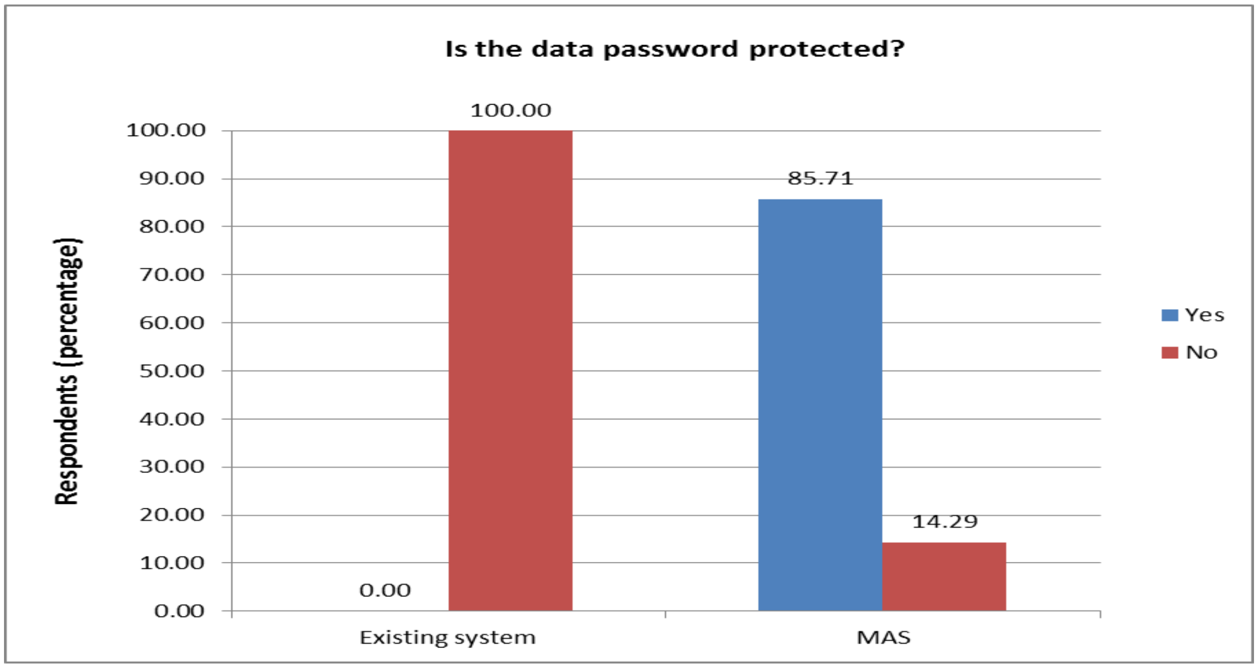


Figure 6-1: graph showing results of whether data being transported to MOH is password protected.

Figure 6-1 clearly shows that in the existing system, data which is transported to the MOH is not password protected which means that anyone can access that data, given a chance. But in the case of the Multi Agent System, 85.71% of the respondents say that the data is password protected, which means that without the password (key), one cannot access the data.

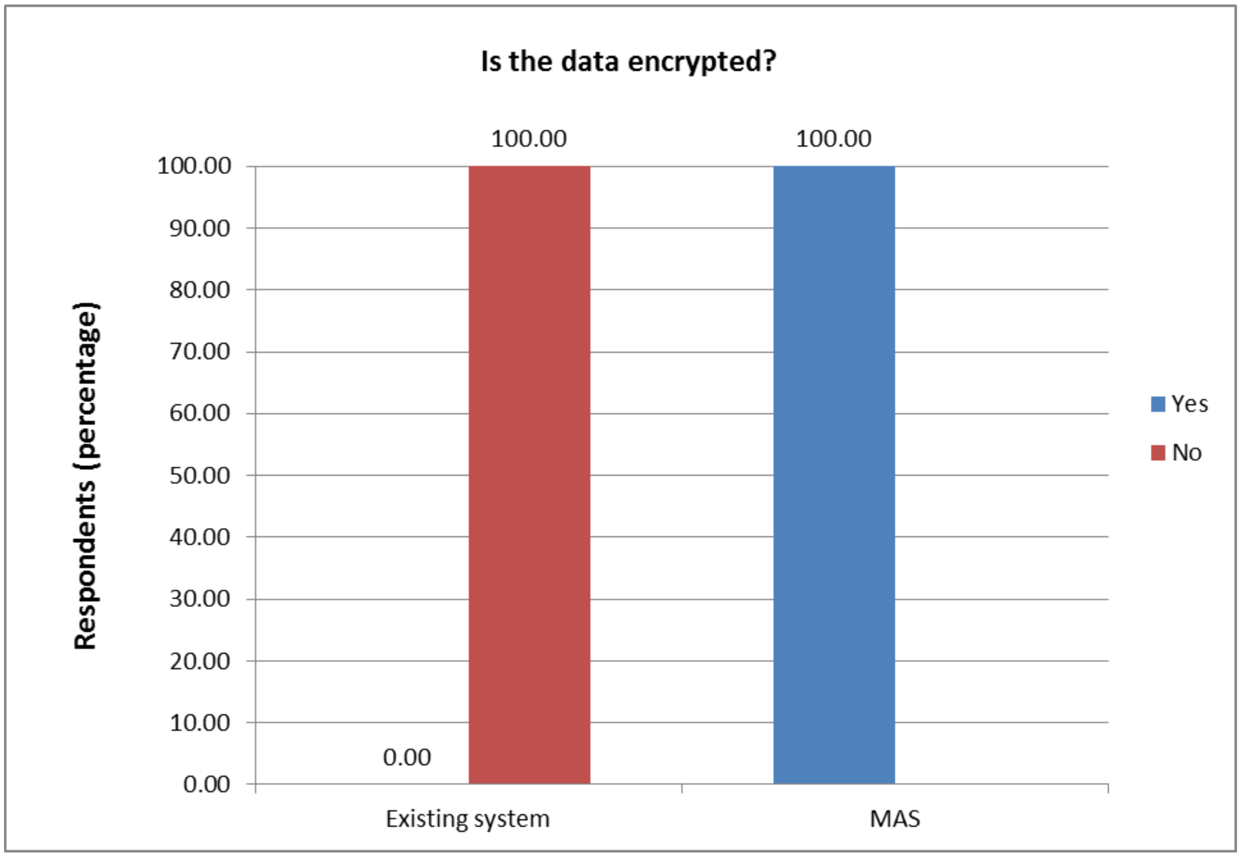


Figure 6-2: graph showing results of whether data being transported to MOH is encrypted.

Figure 6-2 clealry shows that in the existing system, data transported to the MOH is not encrypted, which means that anyone who has access to that data can easily read the data and possibly edit the data. This is a major security risk. But in the MAS, it is clear that the data is encrypted. So if the data falls in the wrong hands, it still will not be readable, unless one has the encryption key.

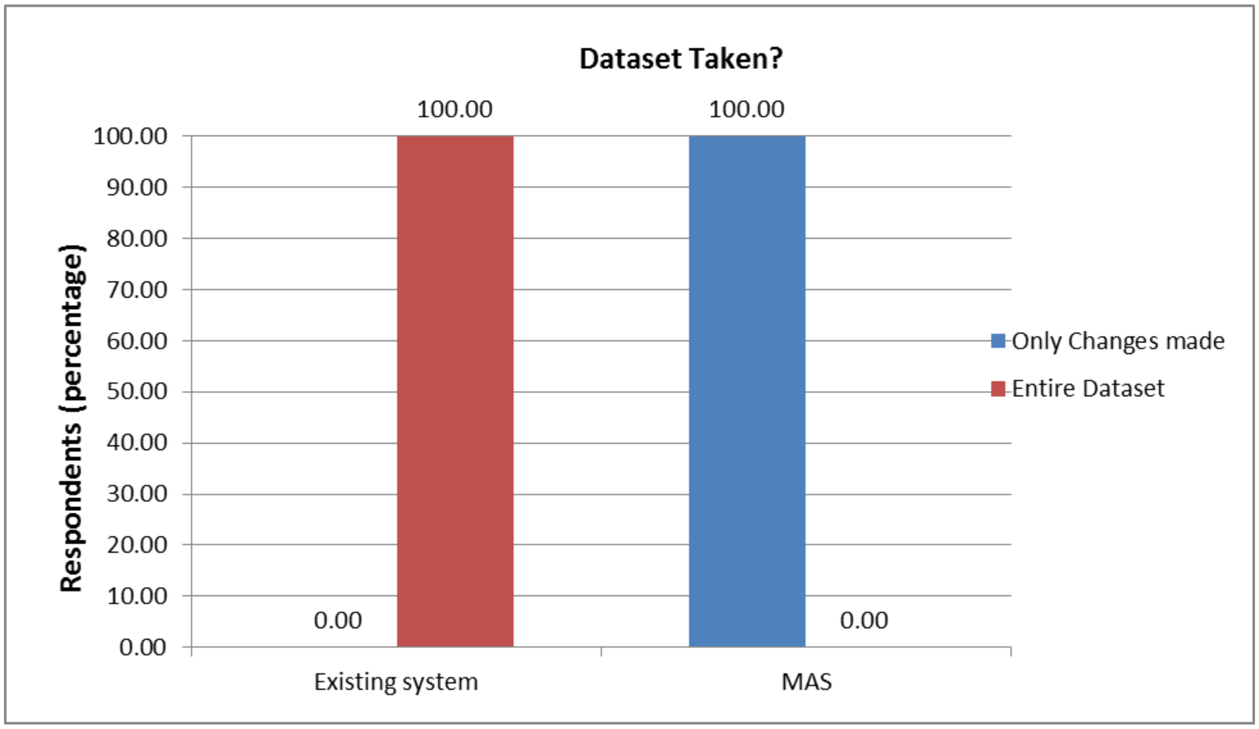


Figure 6-3 : graph showing results of whether data being transported to MOH include the entire dataset or only the changes which has taken place since the last time data was extracted.

Figure 6-3 clearly shows that in the existing system, the entire dataset is transported which means that anyone who has access to the data has access to the records of all health professionals which makes it easier to fabricate or modify the data without being noticed. This is also a major security risk. In the MAS, it is clear that only changes made to dataset are transported, making it a bit difficult to fabricate the data.

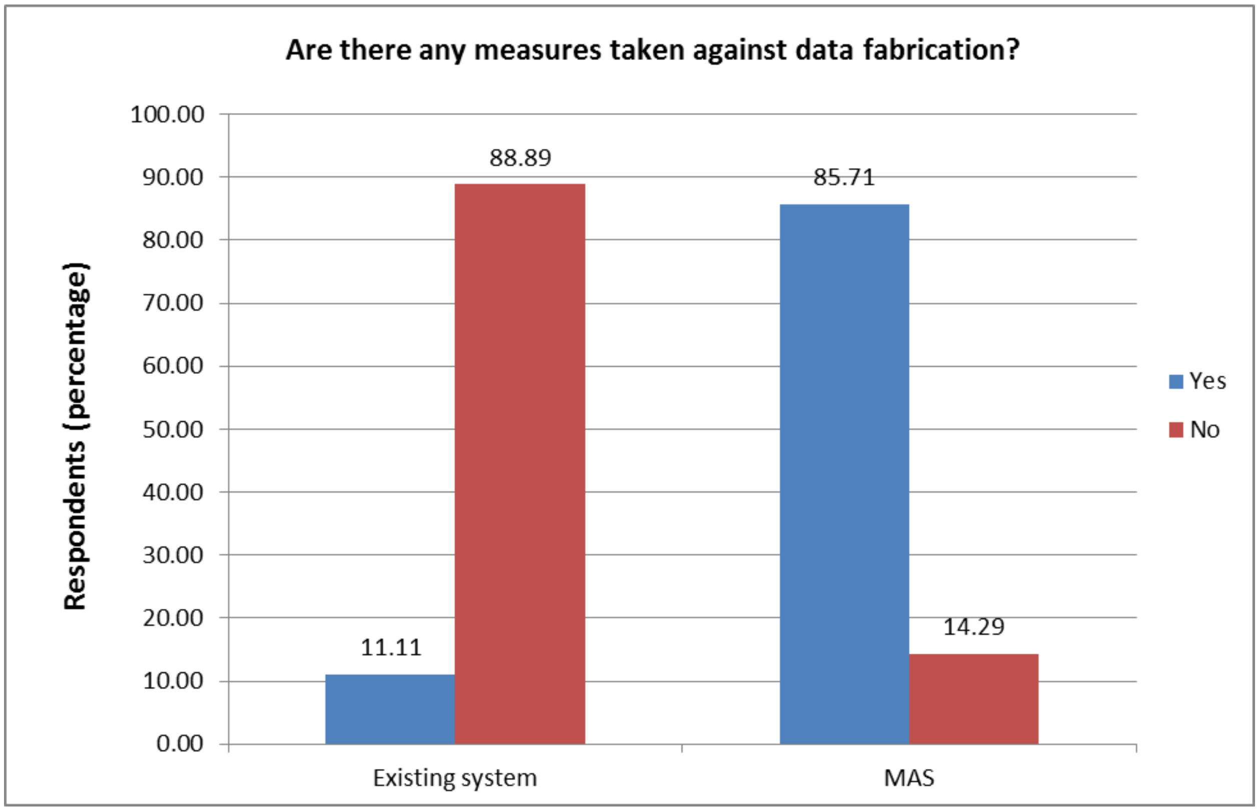


Figure 6-4: graph showing results of whether there any measures taken against data fabrication of data being transported to MOH.

From figure 6-4, we can clearly see that there aren't any measures taken to guard the data against fabrication in the existing system. This is a security risk especially because they transport the entire dataset as is shown in figure 6-3. In the MAS system, it is clear that measures have been taken to guard against data fabrication.

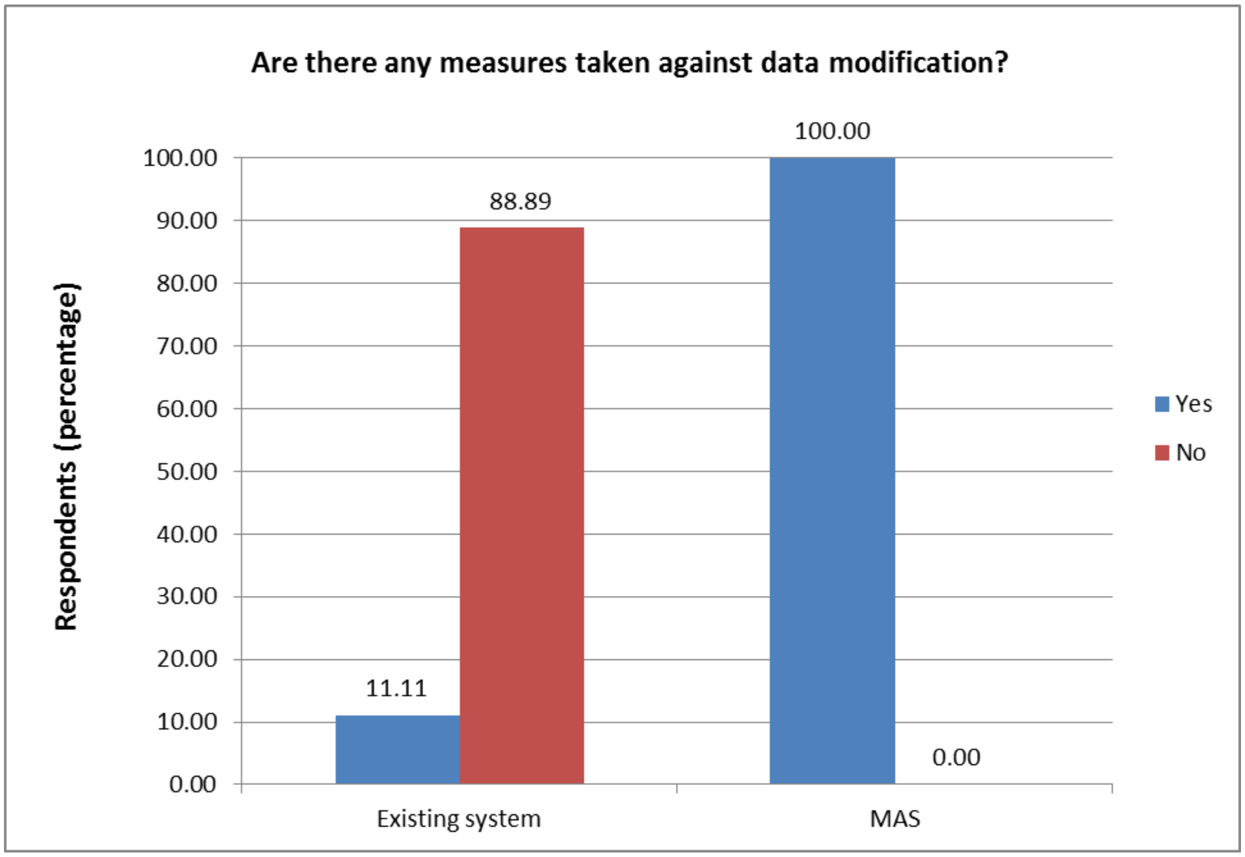


Figure 6-5: graph showing results of whether there any measures taken against data modification of data being transported to MOH.

In figure 6-5, it is clear that in the existing system, no measures have been taken against data modification, while in the MAS measures have been taken to guard the data against modification.

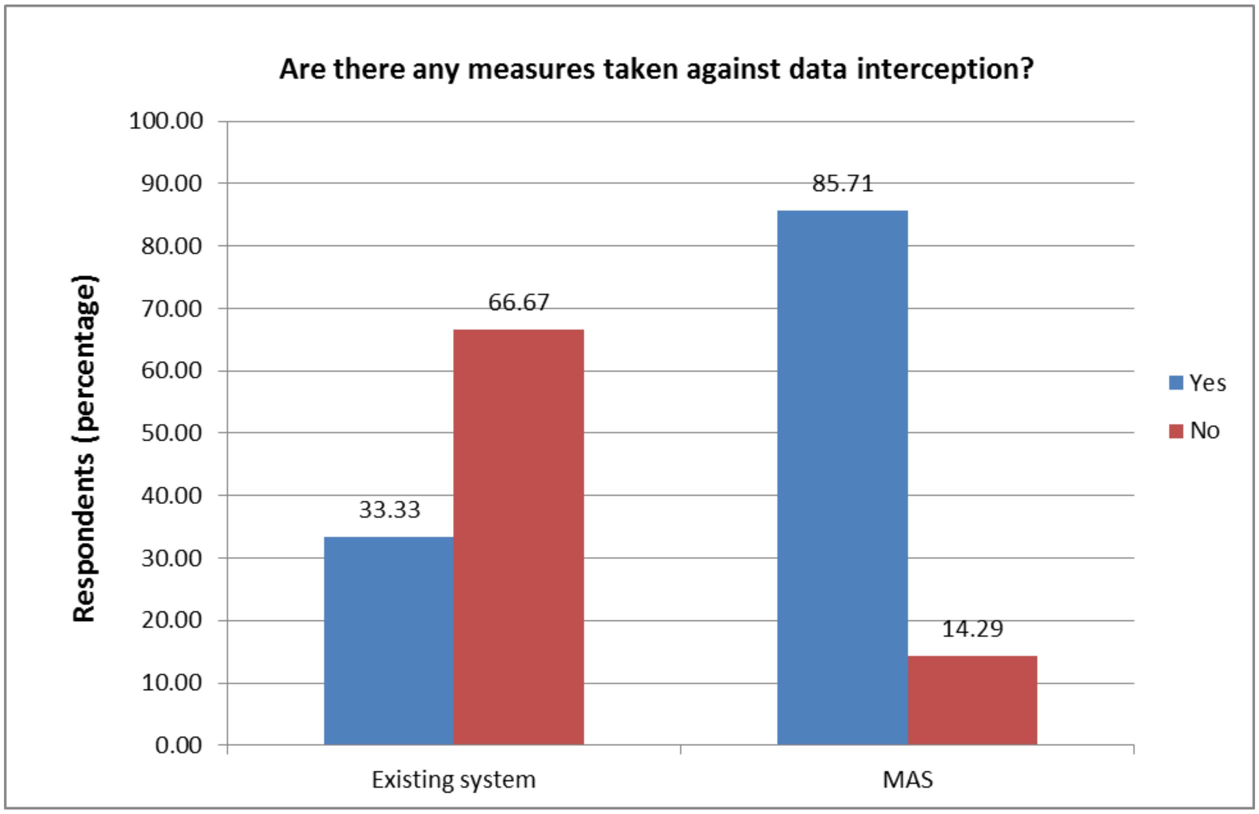


Figure 6-6: graph showing results of whether there any measures taken against data interception of data being transported to MOH.

In figure 6-6, it shows that in the existing system there are no measures taken to guard against data interception which is a security risk, especially because they carry the entire dataset. In the MAS, it is clear that measures have been taken to guard against data interception.

6.2.2 SPEED

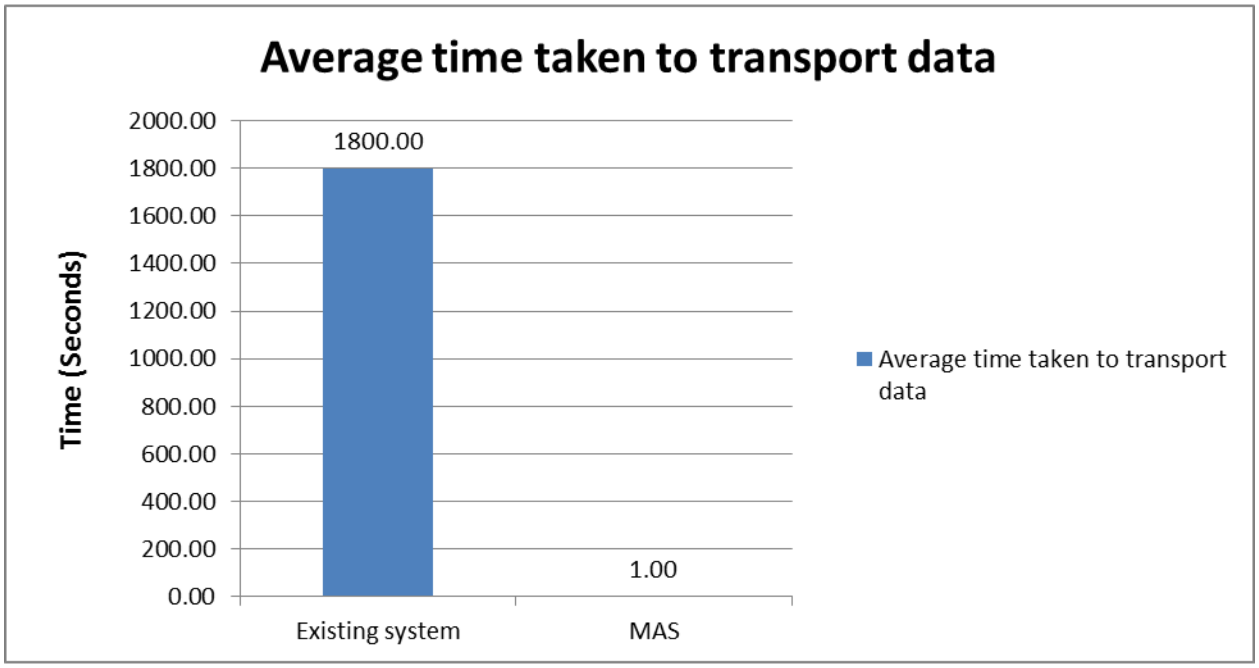


Figure 6-7: graph showing results of average time taken to move data from MOH to regulatory bodies.

In figure 6-7, it is clear that the MAS system is faster in transporting data from the regulatory bodies to the MOH compared to the existing system.

6.3 COMPARISON BETWEEN BOTH SYSTEMS

6.3.1 SECURITY AND CONFIDENTIALITY

In our implementation of the multi-agent system, we have encrypted all the data in the mobile agent using AES (Advanced Encryption standard) algorithm, which is fast and very secure and has been adopted by the United States government [21]. We have also not included any crucial information that may compromise the security of the servers in the mobile agent. This greatly enhances the security and confidentiality of the information being sent to the information server at the ministries of health. This protects the data against fabrication and modification which is not the case with the existing system.

In our implementation, the mobile agent can only hand over the data to an agent with a specific name thus guarding against data interception.

The multi-agent system only transports the changes which have occurred since the last time data was extracted from the regulatory body, thus there is no risk of the entire dataset falling into the wrong hands, as is the case with the existing system.

This clearly shows that the multi-agent system is much more secure than the existing system.

6.3.2 SPEED

Fiber-optic connectivity essentially works by sending and receiving pulses of light through a single-mode cable. The speed at which that pulse of light travels is governed by the laws of physics. Einstein determined the speed of light in a vacuum as 'c,' or approximately 186,282 miles per second. However, since the pulses of light in a trading scenario are not traveling through a vacuum but through a piece of glass, they are slowed down. The ratio between 'c' and the speed at which light travels in a material such as fiber is called the refractive index. The refractive index of light in a vacuum is one.

The refractive index for single-mode fiber can vary slightly based on a number of factors. However, a good estimate is around 1.467, meaning that light travels through fiber at 186,282/1.467 = 124,188 miles per second [20].

The distance between the Nursing council of Kenya (regulatory body) and MOH (Afya House) is 2.54762 miles [22]. It would therefore take the multi-agent system approximately 0.0000205 seconds to send the data to the MOH.

From figure 6-7, it is clearly shown that it would take the existing system at least 1,800 seconds to get the data from the regulatory body and take it to the MOH.

Based on this, it is clear that the multi-agent system would be much faster as opposed to the existing system.

6.4 CONCLUSION

In this chapter we have looked at the results we got from the questionnaires that were handed out to the staff that are currently using the existing system. The questionnaires concentrated on two main areas; security and confidentiality and speed.

From the results we got from the questionnaires, it is clear that the existing system is not secure as no measures have been taken to ensure the security of the data. In the existing system, the data is not encrypted or password protected; no measures have been taken to guard against data fabrication, modification and interception; and they always transport the entire dataset which is a security risk were it to fall in the hands of a malicious person.

In the multi-agent based system, the data is encrypted using AES encryption standard which is the encryption standard adopted by the United Stated government because it is very secure and fast, therefore guarding against data fabrication and modification. The multi-agent based system only transports the changes that have occurred in the database since the last time it extracted data thus there is no chance the entire dataset will fall in the hands of a malicious person.

Also in terms of speed, the multi-agent based system is faster than the existing system.

Based on this, it is clear that the multi-agent based system is more secure and faster than the existing system.

7 CONCLUSION

7.1 OVERVIEW

Decisions on the health workforce of Kenya are very important as far as the health of the citizens of Kenya is concerned, and thus the health managers should have ready access to this data all the time.

From this research, we clearly determined that multi agent systems have been successfully implemented and used in number of health information systems like IHKA and CASIS to mention but a few. We successfully designed and developed a multi agent system model that will keep the information server at the ministries of health current with the data at the regulatory bodies. We also successfully developed the multi agent system. After analyzing and evaluating the multi agent system and the currently existing system, we found out that;

- The multi agent system is very secure as opposed to the existing system.
- The multi agent system is way faster than the existing system.

7.2 LIMITATIONS

The main limitation of the solution is the unreliability of the Internet connection experienced at the regulatory bodies due to power failures and internet cable cuts.

7.3 FUTURE WORK

Due to time limitations, we were not able to handle Internet failures in great detail since we did not get a chance to implement the system over the internet. Therefore some of the areas that will need to be improved on in future will be on implementing the system over the internet and how to handle internet failures, that is, measures taken in case the internet connection fails between regulatory bodies and MOH.

REFERENCES

[1] Fabio Bellifemine, Giovanni Caire, Dominic Greenwood 2007, *Developing Multi-Agent Systems with JADE* pp 115 – 120, John Wiley & Sons Ltd.

[2] Z. Maamar, *A Data Warehousing Environment Based on Software and Mobile Agents*, In Proceedings of the 9th IFIP/IEEE InternationalWorkshop on Distributed Systems: Operations and Management, Newark, Delaware, U.S.A., October 1998.

[3] Stephen O'grandy October 17th 2006, *IBM Defines a New Application Category: The Information Server*, http://www.redmonk.com/sogrady/2006/10/17/ibm-defines-a-new-application-   category-the-information-server-qa/

[4] M.T. zsu and P. Valduriez 1998, *Principles of Distributed Database Systems, 2nd edition*, Prentice-Hall Inc.

[5] Korth and Sudarshan Aug 22, 2005, *Database System Concepts 5th Edition,* Silberschatz.

[6] Mike Chapple October 25th 2012, *Referential Integrity* <http://databases.about.com/b/2012/10/25/referential-integrity-3.htm>

[7] D.M. Chess, B. Grosof, C. Harrison, D. Levine, C. Parris, and G. Tsudik October 1995, *Itinerant Agents for Mobile Computing,* Journal IEEE Personal Communications, vol. 2, pp. 34-49,

[8] T. Papaioannou and J. Edwards May 1998. *Mobile Agent Technology Enabling the Virtual Enterprise: A Pattern for Database Query*. In *Proceedings of the Agent Based ManufacturingWorkshop at Autonomous Agents*.

[9] S. Papastavrou, G. Samaras, and E. Pitoura March 1999. *Mobile Agents for WWW Distributed Database Access*. In *Proceedings of the 15th International Conference on Data Engineering*, Sydney, Australia.

[10] D. Gulbransen, K. Rawlings, and J. December 1996. *Creating Web Applets with Java.* Sams Publishing.

[11] M. Wooldrige, N. R. Jennings and D. Kinny 2000, *The Gaia Methodology for Agent-Oriented Analysis and Design Autonomous Agents and Multi-Agent Systems, volume 3*, pp 285-312, Kluwer Academic Publishers.

[12] Alfredo Garro March 2012, *Modeling Notation, Source: GAIA Version 03-03-12*.

[13] David Chess, Benjamin Grosof, Colin Harrison, David Levine, Colin Parris, and Gene Tsudik October 1995*. Itinerant agents for mobile computing.* IEEE Personal Communications.

[14] Annicchiarico, R., Cortés, U., & Urdiales, C. (Eds.). 2008. *Agent Technology and e-Health. Whitestein Series in Software Agent Technologies and Autonomic Computing*, Babel, Switzerland: Birkhauser Verlag.

[15] Gasser, L. 2001. *MAS Infrastructure Definitions, Needs, and Prospects*. In Wagner, T., Rana, O. (Eds.), Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems. (pp. 1-11), Berlin Germany: Springer Verlag.

[16] Moreno, A., & Nealon, J. (Eds.). 2003. *Applications of Software Agents Technology in the Health Care Domain,* Birkhäuser*.*

[17] Jih, W., Hsu, J.Y., & Tsai 2006, *Context-aware service integration for elderly care in a smart environment.* In D.B. Leake, T.R. Roth-Berghofer, & S. Schulz, (Eds.), AAAI Workshop on Modeling and Retrieval of Context Retrieval of Context, (pp. 44-48), Menlo Park, CA: AAAI Press.

[18] Hashmi, Z. I., Sibte, S., Abidi, R., & Cheah, Y. 2002. *An Intelligent Agent-based Knowledge Broker for Enterprisewide Healthcare Knowledge Procurement*. In Procs. 15th IEEE Symposium on Computer Based Medical Systems, Maribor (Slovenia).

[19] Croitoru, M., Hu, B., Dasmahapatra, S., Lewis, P., Dupplaw, D., Gibb, A., Julia-Sape, M., Vicente, J., Saez, C., Garcia-Gomez, J.M., Roset, R., Estanyol, F., Rafael, X., & Mier, M. (2007). *Conceptual Graphs Based Information Retrieval in Health Agents*, In *CBMS'07: 20th IEEE International Symposium on Computer Based Medical Systems,* IEEE Computer Society, 618-623.

[20] Brian Quigley, April 7 2011, *Speed of Light in Fibre – The first building block of a low-latency trading infrastructure*, http://blog.advaoptical.com/speed-light-fiber-first-building-block-low-latency-trading-infrastructure/

[21] Svante Seleborg, 2007, *Advanced Encryption Standard*, Axantum Software AB.

[22] Google Maps, 2013, https://maps.google.co.ke/, Google Inc.

[23] F. Bellifemine, A. Poggi, G. Rimassa, 2001, *Developing multi agent systems with a FIPA-compliant agent framework in Software - Practice & Experience*, John Wiley & Sons, Ltd., page. 103-128

[24] Joab Jackson, August 18 2011*, AES proved vulnerable by Microsoft researchers, http://www.computerworld.com/s/article/9219297/AES_proved_vulnerable_by_Microsoft_researchers*

[25] Microsoft, 2013, Cryptographic Services, http://msdn.microsoft.com/en-us/library/92f9ye3s.aspx

[26]Bradshaw, J. M., S. Dutfield, et al. (1997), *KAoS: Toward an Industrial-Strength Open Agent Architecture. Software Agents,* AAAI/MIT Press

[27] Debenham J., 1999, *An adaptive, maintainable, extensible process agent*, Springer Berlin Heidelberg.

APPENDICES

APPENDIX A: CODE FOR THE THREE AGENTS

INFOSERVER AGENT CODE

```
import jade.core.Agent;

import jade.core.AID;

import jade.core.behaviours.TickerBehaviour;

import jade.core.behaviours.CyclicBehaviour;

import jade.lang.acl.ACLMessage;

import jade.lang.acl.UnreadableException;

import java.sql.*;

import java.util.Date;

import jade.core.ContainerID;

import jade.core.PlatformID;

public class InfoServerAgent extends Agent{

String[] d;

ACLMessage msg;

protected void setup(){

addBehaviour(new CyclicBehaviour(this)

 {

public void action()

{

msg = receive();

if (msg!=null){
```

```
String dbUrl = "jdbc:mysql://localhost:3306/medicaldb";
String dbClass = "com.mysql.jdbc.Driver";
try{
d = (String[])msg.getContentObject();
}catch(UnreadableException e){}
int dLength = d.length - 2;
for(int i = 0; i <= dLength; i++){
System.out.println( " - " + getLocalName() + " <- " + d[i]);
}
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection (dbUrl,"root","");
Statement stmt = con.createStatement();
for(int i = 0; i <= dLength; i++){
stmt.executeUpdate(d[i]);
}
}
catch(ClassNotFoundException e) {
e.printStackTrace();
}
catch(SQLException e) {
e.printStackTrace();
}
}
else
{
}
}
});
}
}
```

EXTRACTION AGENT SOURCE CODE

```
import jade.core.Agent;
import jade.core.AID;
import jade.core.behaviours.TickerBehaviour;
import jade.lang.acl.ACLMessage;
import java.sql.*;
import java.util.Date;
import java.io.IOException;

public class ExtractionAgent extends Agent{
protected void setup(){
addBehaviour(new TickerBehaviour(this, 30000) {
protected void onTick() {
String dbUrl = "jdbc:mysql://localhost:3306/medicaldb";
String dbClass = "com.mysql.jdbc.Driver";
String getID = "SELECT MAX(last_id) FROM test_id_table";
int last_id = 0;
try{
Class.forName("com.mysql.jdbc.Driver");
```

```
Connection con = DriverManager.getConnection (dbUrl,"root","");
Statement stmt2 = con.createStatement();
ResultSet gID = stmt2.executeQuery(getID);
while(gID.next()){
System.out.println("Last ID value = :" + gID.getInt("MAX(last_id)"));
last_id = gID.getInt("MAX(last_id)");
}
}
catch(ClassNotFoundException e) {
e.printStackTrace();
}
catch(SQLException e) {
e.printStackTrace();
}
String query = "SELECT * FROM test_table_audit WHERE id > " + last_id;
String rows = "SELECT count(*) FROM test_table_audit WHERE id >" + last_id;
int numberRows = 0;
String result;
String[] table;
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection (dbUrl,"root","");
Statement stmt = con.createStatement();
Statement stmt1 = con.createStatement();
ResultSet rs = stmt.executeQuery(query);
ResultSet noRows = stmt1.executeQuery(rows);
```

```
while(noRows.next()){

System.out.println("No of Rows :" + noRows.getInt(1));

numberRows = noRows.getInt(1);

}

if(numberRows != 0){

table = new String[numberRows + 1];

int i = 0;

while (rs.next()) {

if(rs.getString("activity").equals("ins")){

System.out.println("Activity =" + rs.getString("activity"));

table[i] = "INSERT INTO test_table_dup values('" + rs.getString("new_id") +"','" +
rs.getString("new_surname") + "','" + rs.getString("new_fname") +

"','" + rs.getString("new_mname") + "','" + rs.getString("new_gender") + "')";

System.out.println(table[i]);

}

else if(rs.getString("activity").equals("upd")){

System.out.println("Activity =" + rs.getString("activity"));

table[i] = "UPDATE test_table_dup set surname ='" + rs.getString("new_surname") + "',
fname ='" + rs.getString("new_fname") +

"', mname ='" + rs.getString("new_mname") + "', gender ='" + rs.getString("new_gender") +
"' WHERE id ='" + rs.getString("new_id") + "'";

System.out.println(table[i]);

}

else if(rs.getString("activity").equals("del")){

System.out.println("Activity =" + rs.getString("activity"));

table[i] = "DELETE FROM test_table_dup where id ='" + rs.getString("old_id") + "'";

System.out.println(table[i]);
}
else{
System.out.println("INVALID DATA!!!");
}
System.out.println();
table[numberRows] = "INSERT INTO test_id_table (last_id) values(" + rs.getString("id") +
")";
i++;
} //end while
con.close();
ACLMessage msg = new ACLMessage(ACLMessage.REQUEST);
msg.addReceiver(new AID("mobile", AID.ISLOCALNAME));
msg.setLanguage("English");
msg.setOntology("Data-Integrity-ontology");
try{
msg.setContentObject(table);
}catch(IOException e){}
send(msg);
}
} //end try
catch(ClassNotFoundException e) {
e.printStackTrace();
}
catch(SQLException e) {
e.printStackTrace();
}
```

```
Date date = new Date();
System.out.println(date.toString());
System.out.println();
}
} );
}
}
```

MOBILE AGENT SOURCE CODE

```
import jade.core.Agent;
import jade.core.AID;
import jade.core.behaviours.TickerBehaviour;
import jade.core.behaviours.CyclicBehaviour;
import jade.core.behaviours.OneShotBehaviour;
import jade.lang.acl.ACLMessage;
import jade.lang.acl.UnreadableException;
import java.sql.*;
import java.util.Date;
import jade.core.ContainerID;
import jade.core.PlatformID;
import java.io.IOException;
public class MobileAgentRS extends Agent{
String[] d;
ACLMessage msg;
String check = "infoserver";
```

```
protected void setup(){
addBehaviour(new CyclicBehaviour(this)
{
public void action()
{
msg = receive();
AID remoteAMS = new AID("amm@192.168.1.2:1099/JADE", AID.ISGUID);
remoteAMS.addAddresses("http://john-THINK:7778/acc");
PlatformID destination = new PlatformID(remoteAMS);
if (msg!=null){
try{
d = (String[])msg.getContentObject();
}catch(UnreadableException e){}
if(d.length != 0)
{
doMove(destination);
}
}
else
{
block();
}
}
});
}
protected void beforeMove() {
```

```
System.out.println("Moving now to new location : " +getClass().getName());
}
protected void afterMove() {
addBehaviour(new OneShotBehaviour(this)
{
public void action()
{
try
{
System.out.println("Were Here" +getName());
if(check.equals("infoserver")){
ACLMessage msg1 = new ACLMessage(ACLMessage.REQUEST);
msg1.addReceiver(new AID("afya@192.168.1.2:1099/JADE", AID.ISGUID));
try{
msg1.setContentObject(d);
}
catch(IOException e){}
send(msg1);
AID remoteAMS = new AID("amm@192.168.1.1:1099/JADE", AID.ISGUID);
remoteAMS.addAddresses("http://john-PC:7778/acc");
PlatformID destination1 = new PlatformID(remoteAMS);
check = "regulatory";
doMove(destination1);
}
//else //if(getContainerController().getContainerName().equals("MPDB"))
else if(check.equals("regulatory"))
{
String dbUrl = "jdbc:mysql://localhost:3306/medicaldb";
String dbClass = "com.mysql.jdbc.Driver";
System.out.println( " - " + getLocalName() + " <- " + d[d.length-1]);
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection (dbUrl,"root","");
Statement stmt = con.createStatement();
stmt.executeUpdate(d[d.length-1]);
}
catch(ClassNotFoundException e) {
e.printStackTrace();
}
check = "infoserver";
}
else
{
System.out.println("ELSE");
}
}
catch(Exception e){
System.out.println("Catch");
}
}
});
}}
```

APPENDIX B: GANT CHART

Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Initial Theory research and documentation	🟦	🟦	🟦	🟦	🟦	🟦													
Model framework Development					🟦	🟦	🟦	🟦											
Data record structure and format Development								🟦	🟦										
Data Creation									🟦	🟦									
System Model Prototype Development									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦		
Model testing and simulation																	🟦	🟦	
Results interpretation																		🟦	🟦

APPENDIX C: SETTING UP THE DEVELOPMENT ENVIRONMENT

Before we start on developing the agents, we first need to set up the development environment which will involve setting up jade packages, installing java and installing and setting up MySQL server.

SETTING UP JADE

i. First of all we download jade from the official jade website http://jade.tilab.com/. It is advisable to download the latest stable version of jade.

ii. Create installation directory jade. For example C:\jade.

iii. Copy there content of archive JADE-bin-x.x.zip that you downloaded from jade official website in step i. Under directory jade you should have subdirectory lib.

iv. Set CLASSPATH environment variable for c:\jade\lib\jade.jar; c:\jade\lib\http.jar; c:\jade\lib\iiop.jar; c:\jade\lib\jadeTools.jar; c:\jade\lib\commons-codec\commons-codes-x.x.jar

SETTING UP JAVA

i. Download java runtime environment and java development kit and install them in the servers.

SETTING UP MYSQL SERVER

i. Download MySQL server from MySQL official website http://dev.mysql.com/downloads/mysql/.

ii. Install it in the servers.

CONFIGURING THE MYSQL DATABASES AT THE REGULATORY BODIES

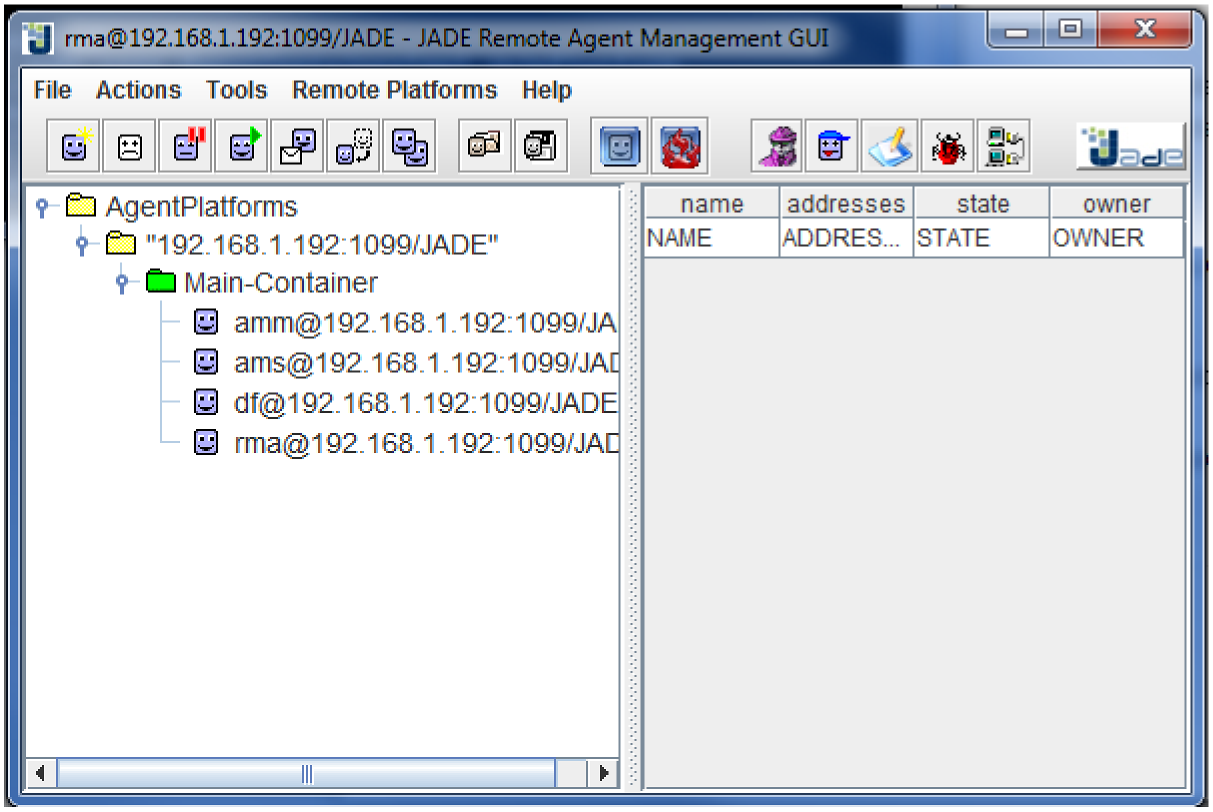
The MySQL databases at the regulatory bodies need to be configured in such a way that they will be able to record all the changes that take place to make it possible for the extraction agent to be able to extract only the changes that have taken place since the last time it extracted data.

In this research, we have implemented this with a trigger and a tracking table which stores the id of the last record to be extracted.

REGULATORY BODIES

For the extraction and mobile agents to run at the regulatory bodies, they need a jade platform on which they will reside and perform their functions. To start the jade platform;

- Start the command prompt on your server.
    - This is done by opening the start menu, click on **All Programs**, **Accessories**, and finally on **Command prompt**.
- Start the jade platform by typing the following command in command prompt - “java jade.Boot -gui -services jade.core.mobility.AgentMobilityService;jade.core.migration.InterPlatformMobilityService -accept-foreign-agents true”. Figure C-1 shows jade platform started.


rma@192.168.1.192:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help

```
AgentPlatforms
  "192.168.1.192:1099/JADE"
    Main-Container
      amm@192.168.1.192:1099/JA
      ams@192.168.1.192:1099/JAD
      df@192.168.1.192:1099/JADE
      rma@192.168.1.192:1099/JAD
```

name	addresses	state	owner
NAME	ADDRES...	STATE	OWNER

Figure C-1: Jade platform started at regulatory body

To start the extraction agent, right click on the main-container in the jade platform, and then click on start new agent as shown below.

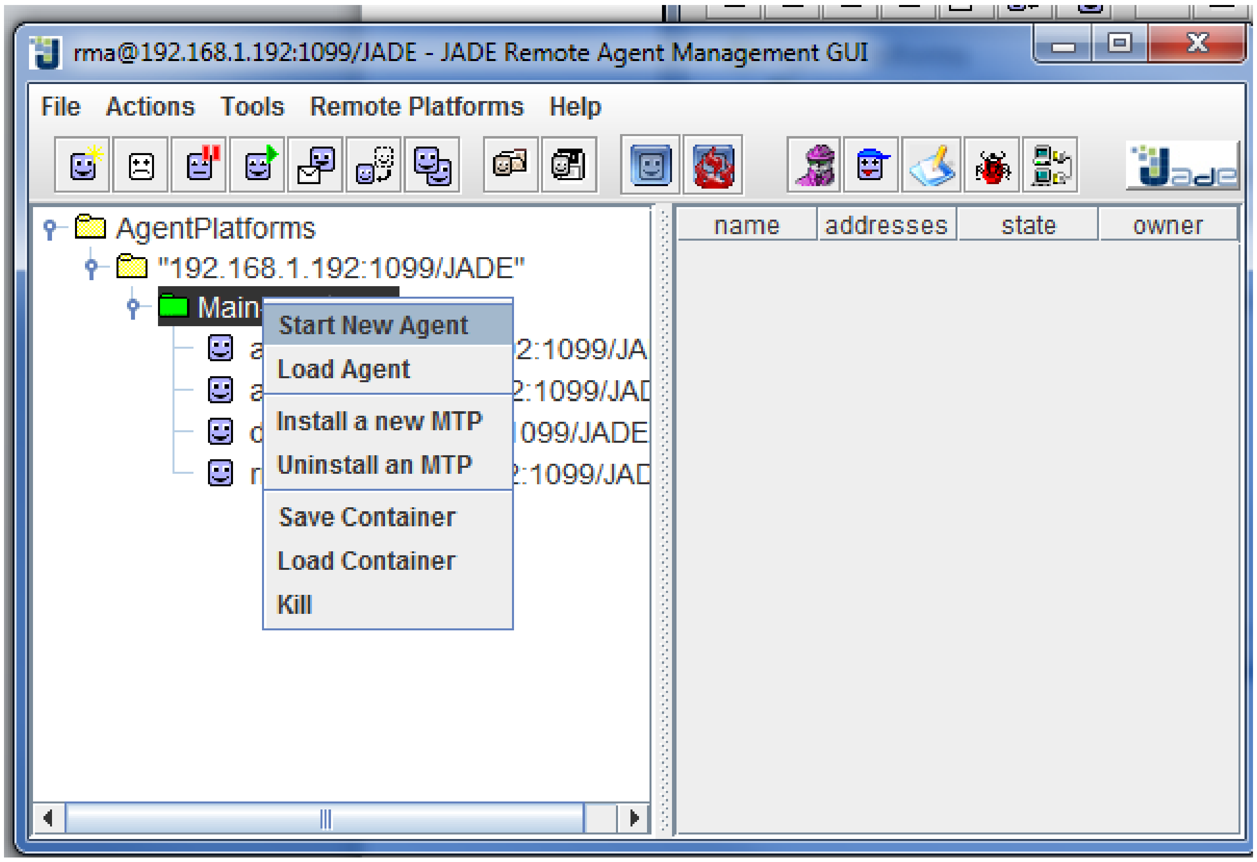
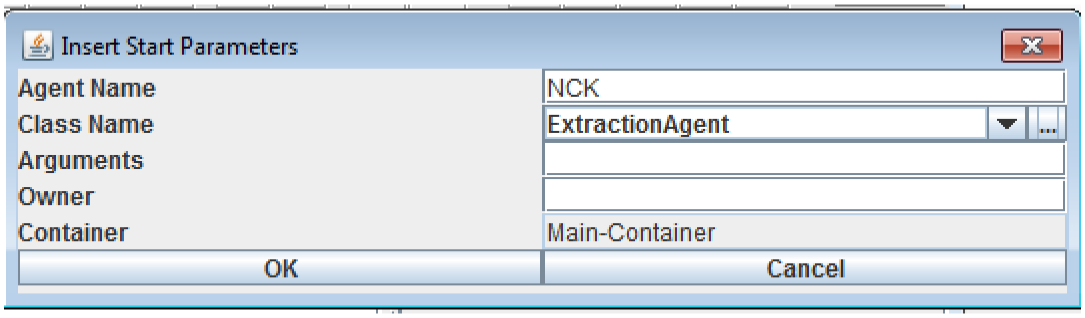


Figure C-2: How to start a new agent in the jade platform.


Insert Start Parameters

Agent Name	NCK
Class Name	ExtractionAgent
Arguments	
Owner	
Container	Main-Container

Figure C-3: Inserting parameters of the extraction agent you want to start, agent name and class

In figure C-3 above, is invoked after you click on "start new agent" in figure C-2. In figure C-3, we have set the agent name as "NCK" and the agent class as "extraction Agent" which is the class that defines the extraction agent. After you click the OK button in figure C-3, the extraction agent called "NCK" is started as shown in figure C-4 below.

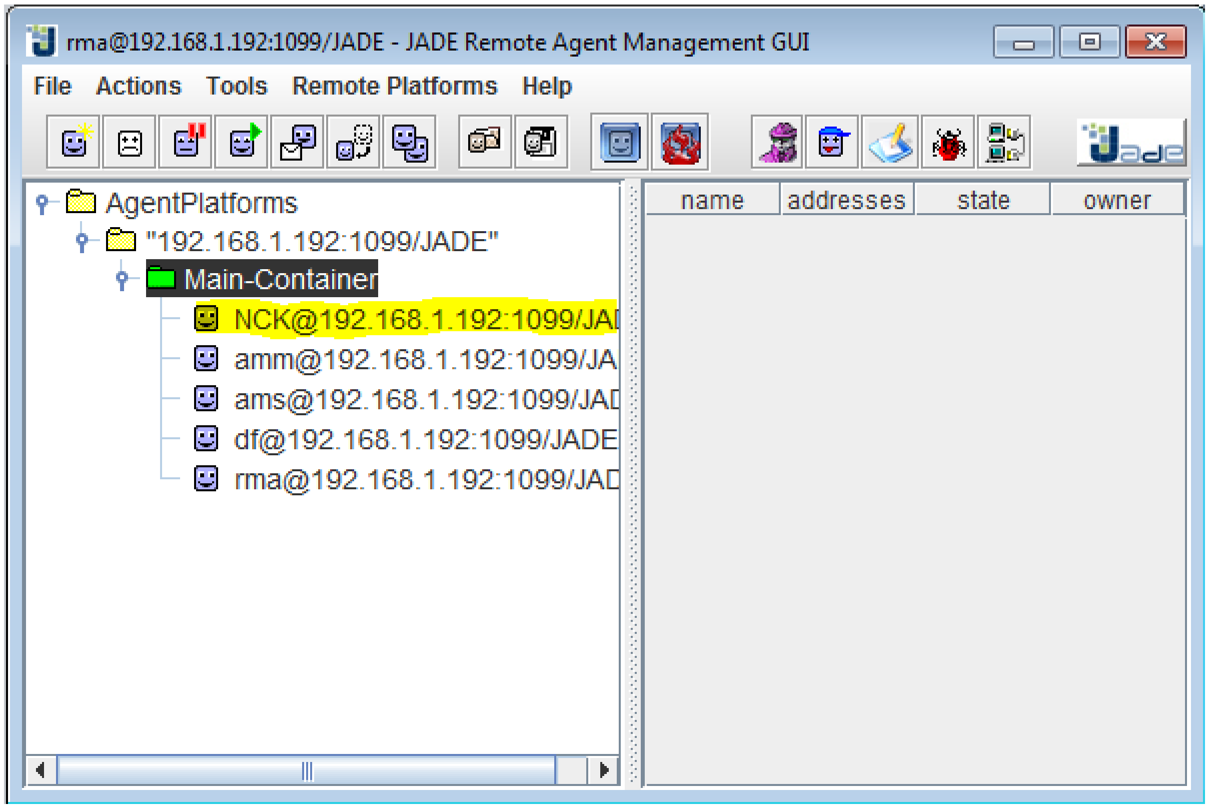
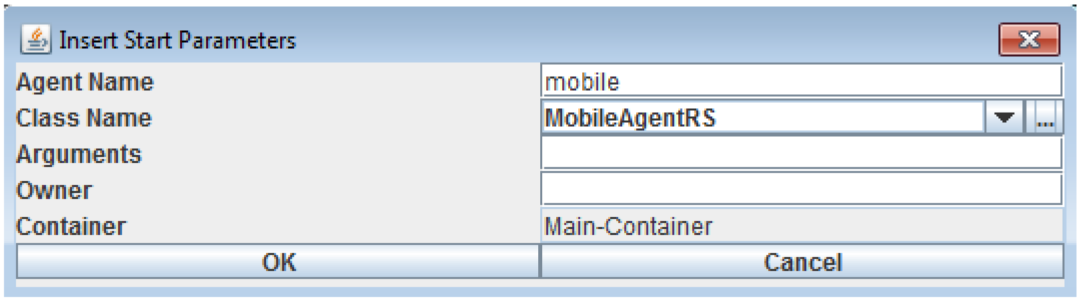

rma@192.168.1.192:1099/JADE - JADE Remote Agent Management GUI
File Actions Tools Remote Platforms Help
AgentPlatforms
"192.168.1.192:1099/JADE"
Main-Container
NCK@192.168.1.192:1099/JAI
amm@192.168.1.192:1099/JA
ams@192.168.1.192:1099/JAI
df@192.168.1.192:1099/JADE
rma@192.168.1.192:1099/JAI
name
addresses
state
owner

Figure C-4: extraction agent "NCK" started on the jade platform.

Starting the mobile agent is the same as starting the extraction agent, but change the parameters to;

- agent name = "mobile"
- agent class = "MobileAgentRS"

as shown in figure C-5 below



Agent Name	mobile
Class Name	MobileAgentRS
Arguments	
Owner	
Container	Main-Container

Figure C-5: Inserting parameters for the mobile agent

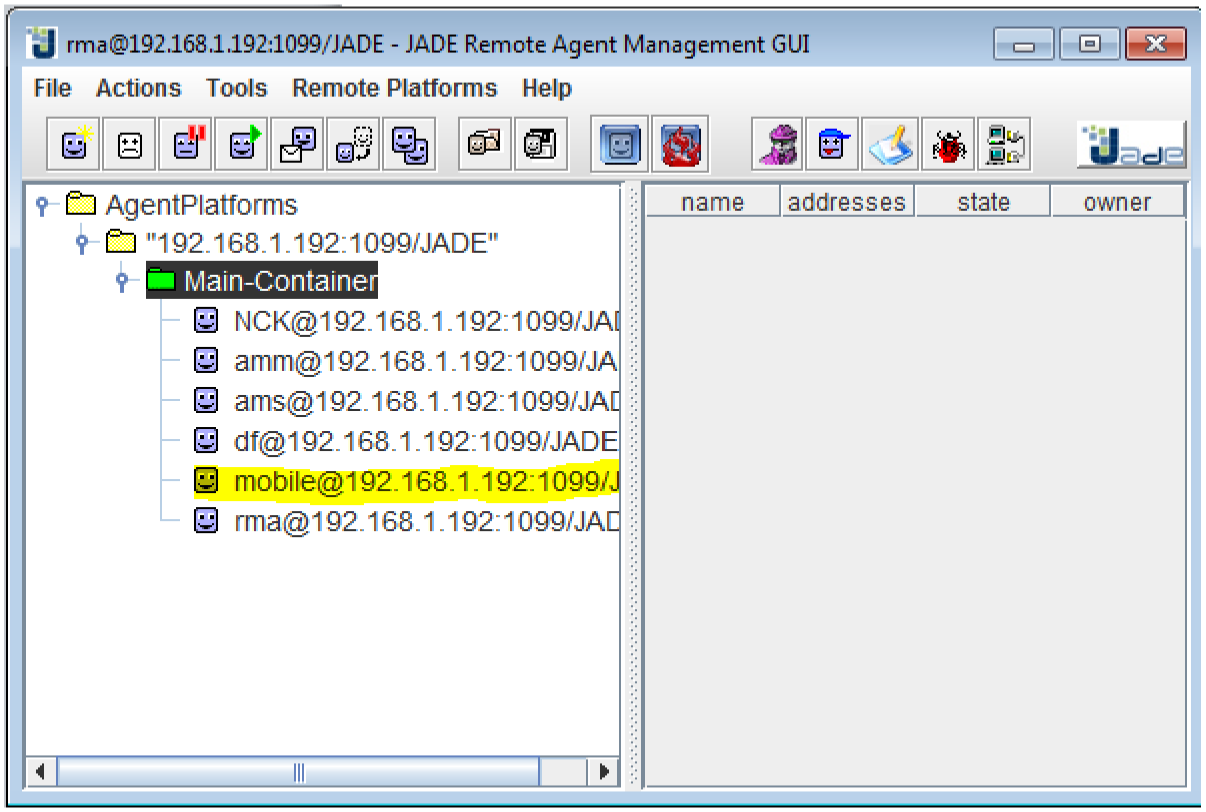

rma@192.168.1.192:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"192.168.1.192:1099/JADE"
Main-Container
NCK@192.168.1.192:1099/JAI
amm@192.168.1.192:1099/JA
ams@192.168.1.192:1099/JAI
df@192.168.1.192:1099/JADE
mobile@192.168.1.192:1099/J
rma@192.168.1.192:1099/JAI
name
addresses
state
owner

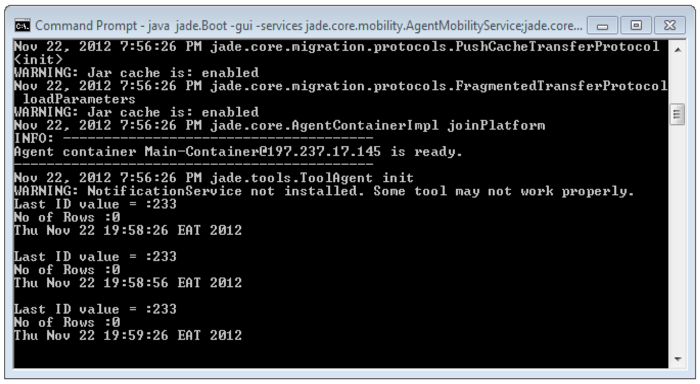
Figure C-6: Mobile Agent Started on jade platform in the regulatory body

With the extraction and mobile agents running at the regulatory bodies, all is set, as shown in Figure C-6.

Outputs of the extraction and mobile agents at the regulatory bodies

Extraction Agent

The extraction agent will be located in the server at the regulatory body and will check for any changes in the database after every 30 seconds. If the agent finds no changes, it does not contact the mobile agent. (Number of rows returned is 0).


Command Prompt - java jade.Boot -gui -services jade.core.mobility.AgentMobilityService;jade.core...

```
Nov 22, 2012 7:56:26 PM jade.core.migration.protocols.PushCacheTransferProtocol
<init>
WARNING: Jar cache is: enabled
Nov 22, 2012 7:56:26 PM jade.core.migration.protocols.FragmentedTransferProtocol
 loadParameters
WARNING: Jar cache is: enabled
Nov 22, 2012 7:56:26 PM jade.core.AgentContainerImpl joinPlatform
INFO: --------------------------------------
Agent container Main-Container@197.237.17.145 is ready.
--------------------------------------------
Nov 22, 2012 7:56:26 PM jade.tools.ToolAgent init
WARNING: NotificationService not installed. Some tool may not work properly.
Last ID value = :233
No of Rows :0
Thu Nov 22 19:58:26 EAT 2012

Last ID value = :233
No of Rows :0
Thu Nov 22 19:58:56 EAT 2012

Last ID value = :233
No of Rows :0
Thu Nov 22 19:59:26 EAT 2012
```

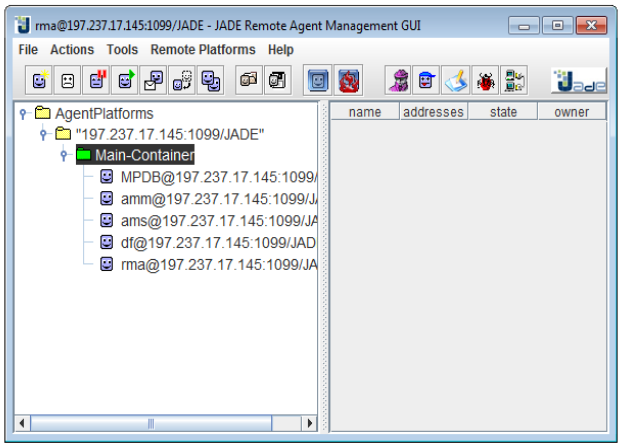
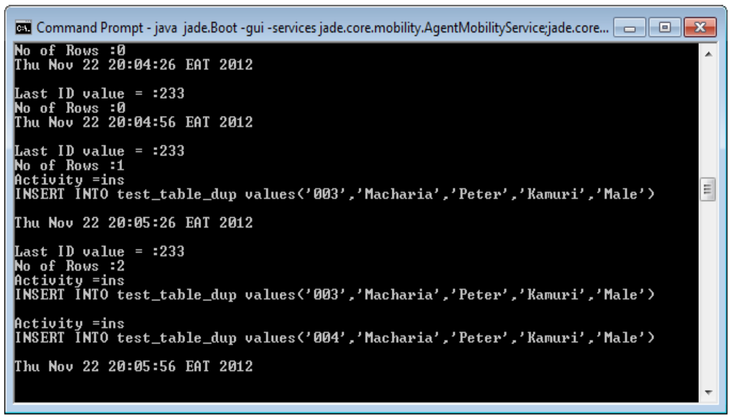

rma@197.237.17.145:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"197.237.17.145:1099/JADE"
Main-Container
MPDB@197.237.17.145:1099/
amm@197.237.17.145:1099/J
ams@197.237.17.145:1099/JA
df@197.237.17.145:1099/JAD
rma@197.237.17.145:1099/JA
name
addresses
state
owner

Figure C-7: Running Extraction Agent which finds no changes made to database – number of rows returned is 0

If the extraction agent finds changes in the database, it returns the number of rows to be affected, then contacts the mobile agent and hands over the changes to the mobile agent.


Command Prompt - java jade.Boot -gui -services jade.core.mobility.AgentMobilityService;jade.core...

```
No of Rows :0
Thu Nov 22 20:04:26 EAT 2012

Last ID value = :233
No of Rows :0
Thu Nov 22 20:04:56 EAT 2012

Last ID value = :233
No of Rows :1
Activity =ins
INSERT INTO test_table_dup values('003','Macharia','Peter','Kamuri','Male')

Thu Nov 22 20:05:26 EAT 2012

Last ID value = :233
No of Rows :2
Activity =ins
INSERT INTO test_table_dup values('003','Macharia','Peter','Kamuri','Male')

Activity =ins
INSERT INTO test_table_dup values('004','Macharia','Peter','Kamuri','Male')

Thu Nov 22 20:05:56 EAT 2012
```

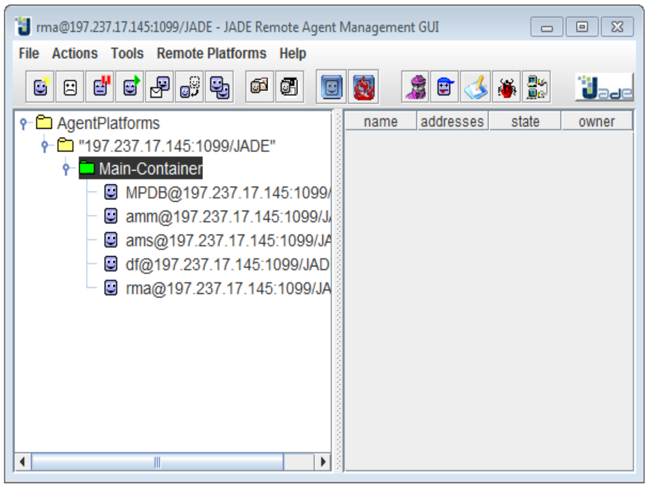

rma@197.237.17.145:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"197.237.17.145:1099/JADE"
Main-Container
MPDB@197.237.17.145:1099/
amm@197.237.17.145:1099/J
ams@197.237.17.145:1099/JA
df@197.237.17.145:1099/JAD
rma@197.237.17.145:1099/JA
name
addresses
state
owner

Figure C-8: Running Extraction Agent which finds some changes made to the database. Some new records have been inserted into the database as shown in the fugure.

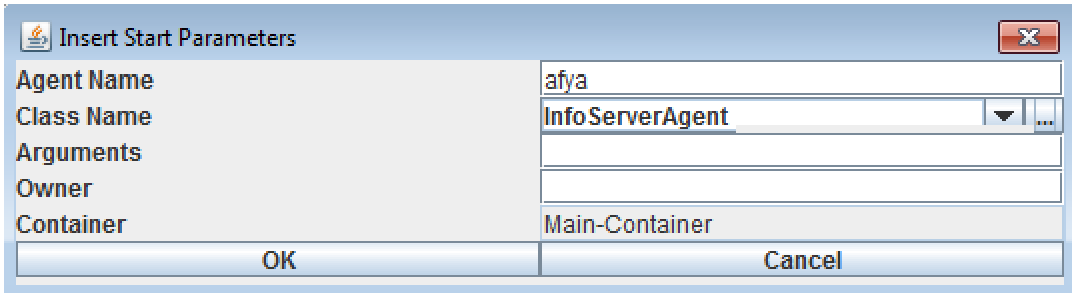
Mobile Agent

The mobile agent is also based on the regulatory body server. It stays idle until it is contacted by the extraction agent. The extraction agent hands over any changes to the database to the mobile agent which transports the data to the information server.

For the information server agent to run at the ministry of health, it needs a jade platform on which it will reside and perform its functions. To start the jade platform;

- Start the command prompt on your server.
    - This is done by opening the start menu, click on **All Programs**, **Accessories**, and finally on **Command prompt**.
- Start the jade platform by typing the following command in command prompt - "java jade.Boot -gui -services jade.core.mobility.AgentMobilityService;jade.core.migration.InterPlatformMobilityService -accept-foreign-agents true". Figure C-1 shows jade platform started.

To start the information server agent, right click on the main-container in the jade platform, and then click on start new agent as shown in figure C-2.



Agent Name	afya
Class Name	InfoServerAgent
Arguments	
Owner	
Container	Main-Container

Figure C-9: Inserting parameters of the information server agent you want to start, agent name and class

In figure C-9 above, is invoked after you click on "start new agent" in figure C-2. In figure C-9, we have set the agent name as "afya" and the agent class as "InfoServer Agent" which is the class that defines the information server agent. After you click the OK button in figure C-9, the information server agent called "afya" is started as shown in figure C-10 below.

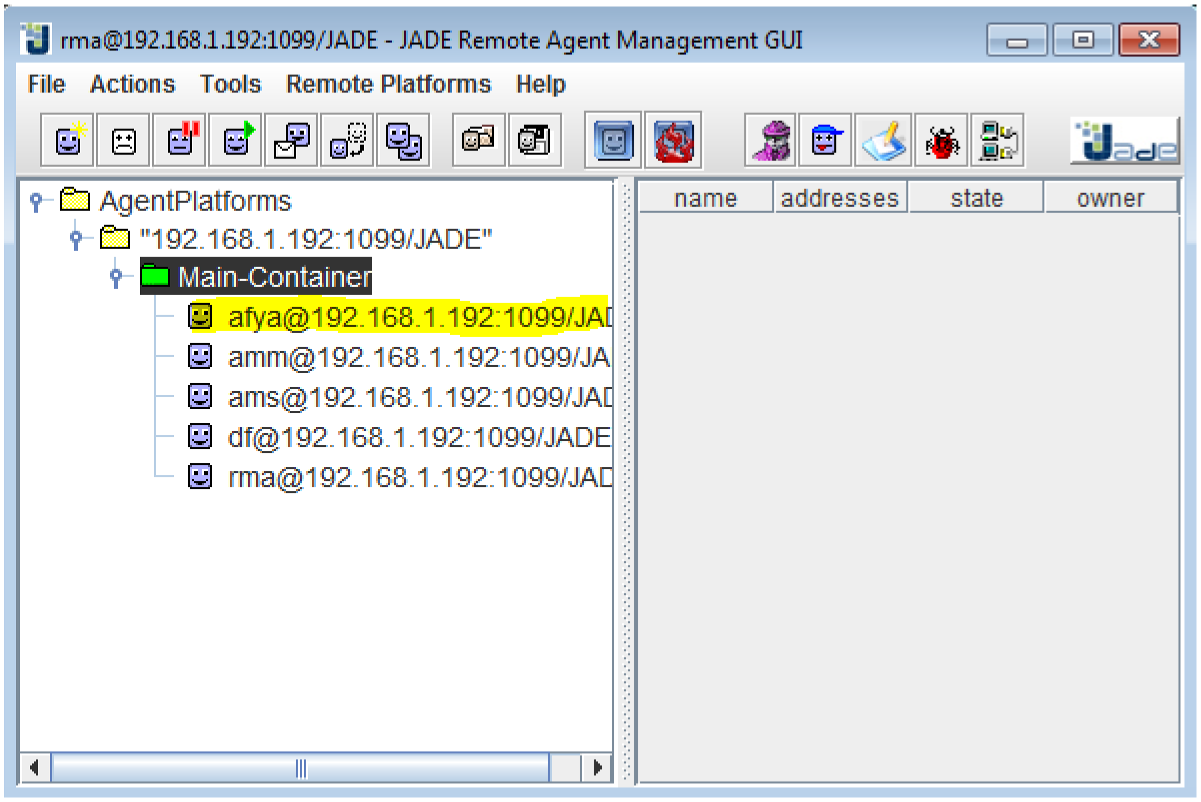
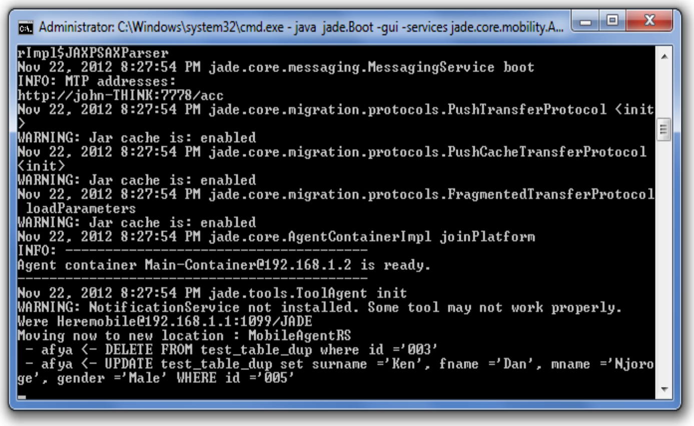

rma@192.168.1.192:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"192.168.1.192:1099/JADE"
Main-Container
afya@192.168.1.192:1099/JAI
amm@192.168.1.192:1099/JA
ams@192.168.1.192:1099/JAI
df@192.168.1.192:1099/JADE
rma@192.168.1.192:1099/JAI
name
addresses
state
owner

Figure C-10: information server agent started on jade platform in the Information Server at the ministry of health.

Output of the Information server at the ministry of health

Information Server Agent

The information server agent is located in the information server at the ministries of health. It is activated by the mobile agent when it gets to the information server to deliver the most current data from the regulatory bodies. The information server agent updates the information server with data from the mobile agent.


Administrator: C:\Windows\system32\cmd.exe - java jade.Boot -gui -services jade.core.mobility.A...

```
rImpl$JAXPSAXParser
Nov 22, 2012 8:27:54 PM jade.core.messaging.MessagingService boot
INFO: MTP addresses:
http://john-THINK:7778/acc
Nov 22, 2012 8:27:54 PM jade.core.migration.protocols.PushTransferProtocol <init
>
WARNING: Jar cache is: enabled
Nov 22, 2012 8:27:54 PM jade.core.migration.protocols.PushCacheTransferProtocol
<init>
WARNING: Jar cache is: enabled
Nov 22, 2012 8:27:54 PM jade.core.migration.protocols.FragmentedTransferProtocol
 loadParameters
WARNING: Jar cache is: enabled
Nov 22, 2012 8:27:54 PM jade.core.AgentContainerImpl joinPlatform
INFO: --------------------------------------
Agent container Main-Container@192.168.1.2 is ready.
--------------------------------------------
Nov 22, 2012 8:27:54 PM jade.tools.ToolAgent init
WARNING: NotificationService not installed. Some tool may not work properly.
Were Heremobile@192.168.1.1:1099/JADE
Moving now to new location : MobileAgentRS
 - afya <- DELETE FROM test_table_dup where id ='003'
 - afya <- UPDATE test_table_dup set surname ='Ken', fname ='Dan', mname ='Njoro
ge', gender ='Male' WHERE id ='005'
```

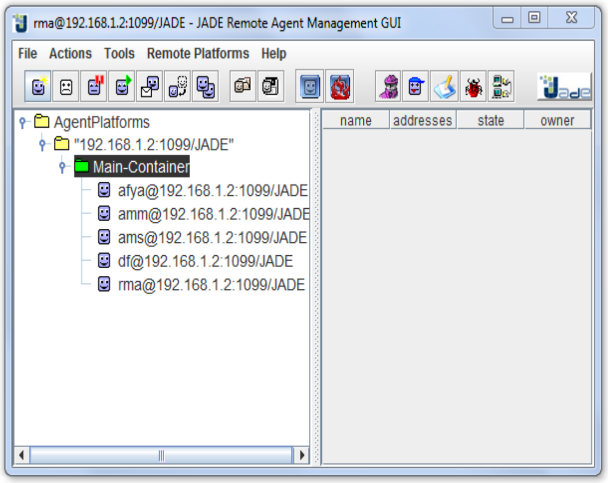
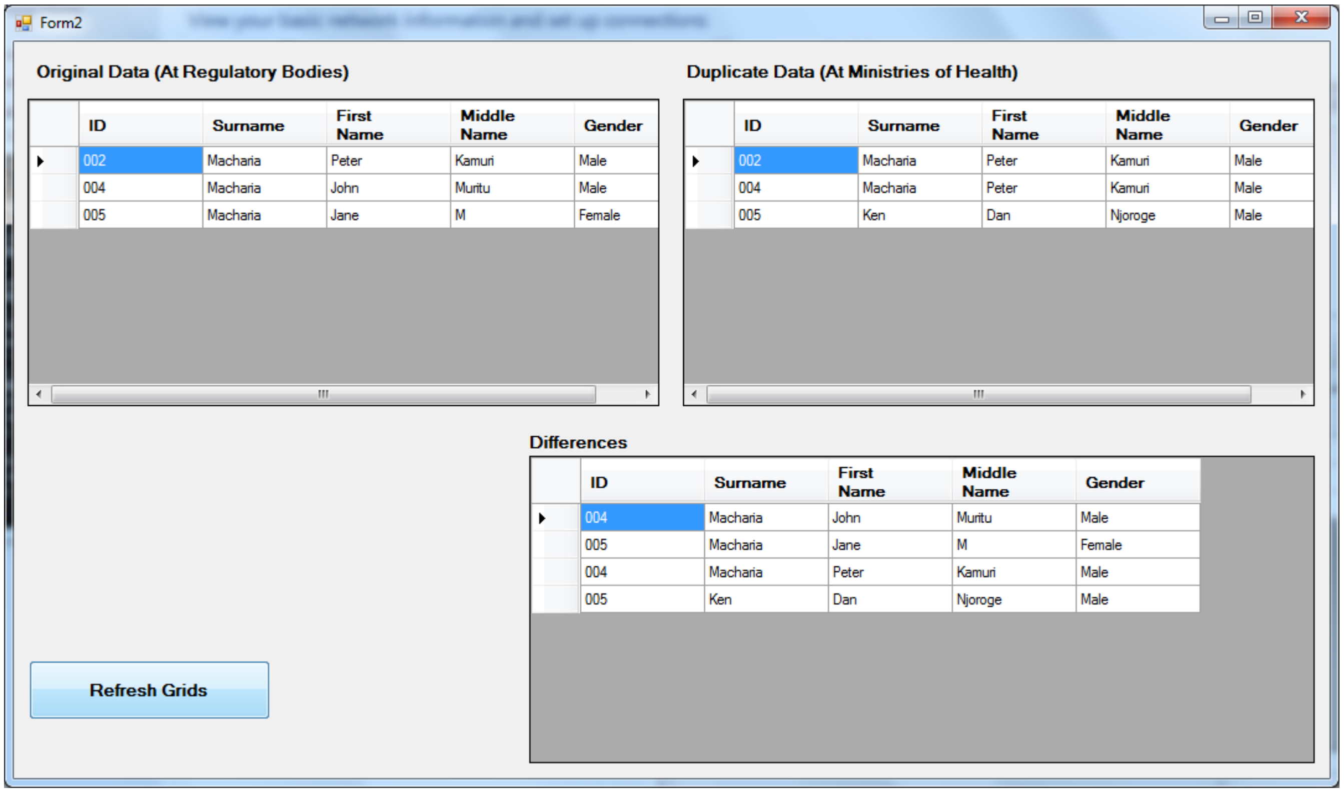

rma@192.168.1.2:1099/JADE - JADE Remote Agent Management GUI
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"192.168.1.2:1099/JADE"
Main-Container
afya@192.168.1.2:1099/JADE
amm@192.168.1.2:1099/JADE
ams@192.168.1.2:1099/JADE
df@192.168.1.2:1099/JADE
rma@192.168.1.2:1099/JADE
name
addresses
state
owner

Figure C-11: Running Information Server Agent receives and executes a delete and an update function on the information server.

Verification Tool

We developed a verification tool for comparing the two databases (regulatory server database and information server database) using C#. Below are the results.



Original Data (At Regulatory Bodies)

ID	Surname	First Name	Middle Name	Gender
002	Macharia	Peter	Kamuri	Male
004	Macharia	John	Muritu	Male
005	Macharia	Jane	M	Female

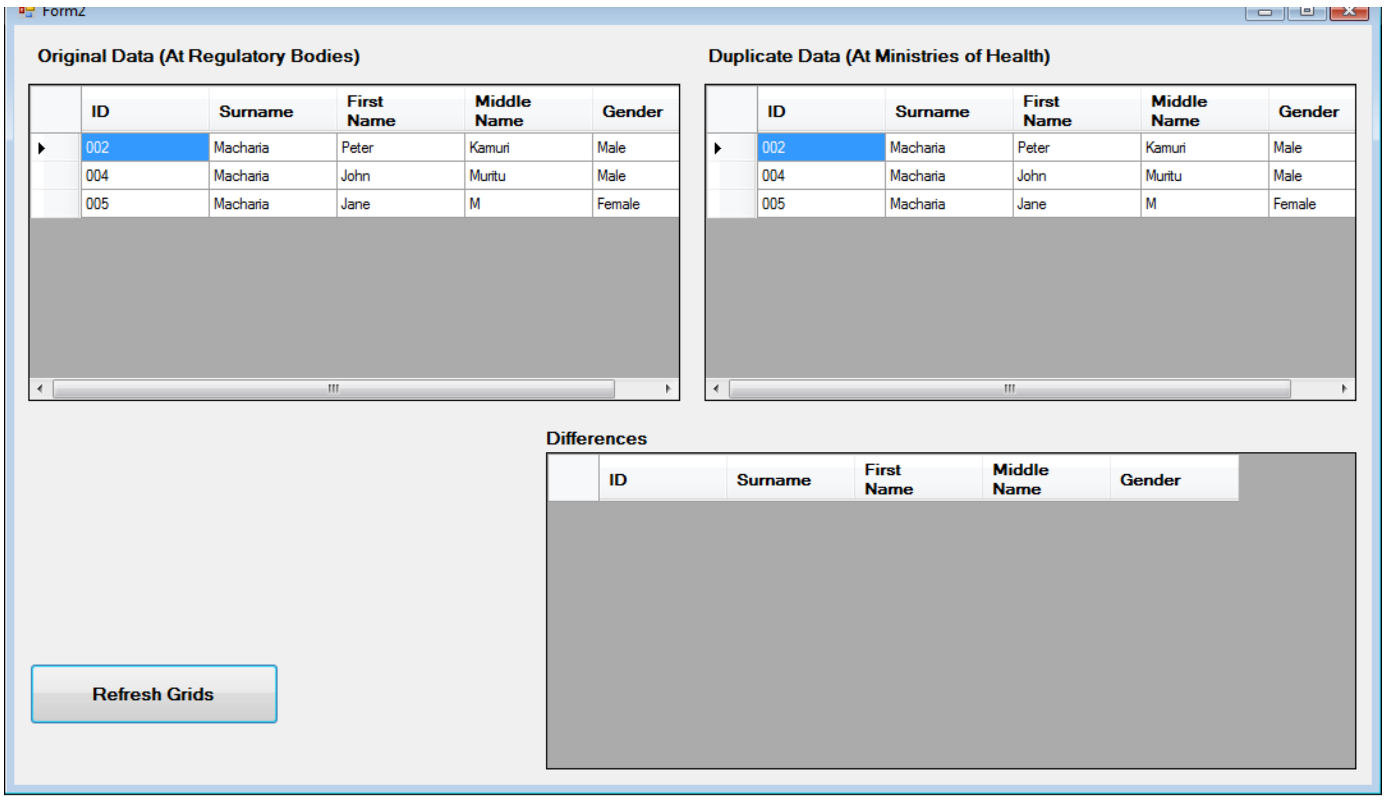
Duplicate Data (At Ministries of Health)

ID	Surname	First Name	Middle Name	Gender
002	Macharia	Peter	Kamuri	Male
004	Macharia	Peter	Kamuri	Male
005	Ken	Dan	Njoroge	Male

Differences

ID	Surname	First Name	Middle Name	Gender
004	Macharia	John	Muritu	Male
005	Macharia	Jane	M	Female
004	Macharia	Peter	Kamuri	Male
005	Ken	Dan	Njoroge	Male

Figure C-12: results right after changes are made to regulatory body database and before information server is updated. The data grid labeled "differences" shows the different records in the two databases, regulatory and ministry of health databases.



Original Data (At Regulatory Bodies)

ID	Surname	First Name	Middle Name	Gender
002	Macharia	Peter	Kamuri	Male
004	Macharia	John	Muritu	Male
005	Macharia	Jane	M	Female

Duplicate Data (At Ministries of Health)

ID	Surname	First Name	Middle Name	Gender
002	Macharia	Peter	Kamuri	Male
004	Macharia	John	Muritu	Male
005	Macharia	Jane	M	Female

Differences

ID	Surname	First Name	Middle Name	Gender

Figure C-13: results after extraction agent extracts the changes, sends them to mobile agent which transports the changes to the ministry of health and sends the changes to the information server agent which updates the information server. Data grid labeled "differences" is empty because both databases have exactly similar records.

Explanation

In figures C-12 and C-13, grid in the top left is the regulatory body data; grid on the top right is the information server data; grid at the bottom shows the differences between the two databases.

In figure C-12, it can be seen that there are differences between the two databases. This is because the mobile agent is yet to be contacted to by the extraction agent.

After 30 seconds elapse, the extraction agent contacts the mobile agent and hands over the data. Mobile agent moves via network with changes and passes on the data to information server agent which updates the information server, thus figure C-13.

It can be seen from figure C-13 that there are no differences between the two databases (the two databases are identical).

APPENDIX D: HOW TO RUN THE SIMULATION

i. Install JAVA EE SDK 6u1 JDK and JAVA JDK 1.6.10 or later versions software on the computer to run the system (if not installed)
ii. Copy extractionAgent.Class and MobileAgent.Class files on the regulatory board server, and InfoServerAgent.Class file on the Information Server.
iii. On the regulatory board Server:
iv. Open the command prompt screen by clicking on start button, then type CMD on the search program text area and click enter.
v. Then go to the folder in which you copied the extractionAgent.class and mobileAgent.class files (using the command prompt).
vi. Run the following command "java jade.Boot -gui -services jade.core.mobility.AgentMobilityService;jade.core.migration.InterPlatformMobilityService -accept-foreign-agents true". This will launch the Jade User Interface which you will use to launch the extraction agent and mobile agent.
vii. First launch the extraction agent.
viii. In the Jade User Interface, click on start new agent, type "MPDB" as name and "ExtractionAgent" as class. Then click ok.
ix. Then launch the mobile agent.
x. In the Jade User Interface, click on start new agent, type "mobile" as name and "MobileAgent" as class. Then click ok
xi. Repeat the same process on the information server, only this time only launch the InfoServerAgent.
xii. On the jade user interface, click on start new agent. Type "afya" as name and "InfoServerAgent" as class.
xiii. Ensure that the network connection between the servers running correctly.
xiv. Now any changes made on the regulatory server will be automatically updated in the information server after every 30 seconds.
xv. To stop the agents, you can click on the agent you want to stop in the jade user interface, and then click on pause agent. To destroy the agent, click on kill agent.

APPENDIX E: QUESTIONNAIRE

TRANSFER OF DATA FROM REGULATORY BODIES TO MINISTRIES OF HEALTH ASSESSMENT SURVEY

We are interested in your comments and suggestions in the current practices of transfer of health workforce data from the regulatory bodies to the ministries of health. The results of this evaluation will be used to improve the current practices.

Speed

1. How long does it take to move from the ministries of health (AFYA House) to the nursing council of Kenya (regulatory body)?

   ☐ < 5 minutes ☐ 5 to 10 minutes ☐ 10 to 20 minutes ☐ > 20 minutes

2. How long does it take to extract the data from the regulatory body?

   ☐ < 5 minutes ☐ 5 to 10 minutes ☐ 10 to 20 minutes ☐ > 20 minutes

Security and Confidentiality

3. Does the data require a password to view it?

   ☐ Yes ☐ No

4. Is the data encrypted?

   ☐ Yes ☐ No

5. Are there measures in place to ensure that the data is not interrupted during transit?

   ☐ Yes ☐ No

   If yes, what measures?

   ____
   ____
   ____

6. Are there measures in place to ensure that the data is not intercepted during transit?

   ☐ Yes ☐ No

If yes, what measures?

____
____
____

7. Are there measures in place to ensure that the data is not modified during transit?

   ☐ Yes ☐ No

   If yes, what measures?

____
____
____

8. Are there measures in place to ensure that the data is not fabricated during transit?

   ☐ Yes ☐ No

   If yes, what measures?

____
____
____

9. Do you carry the entire dataset or just the changes done since the last time?

   ☐ Entire Dataset ☐ Only changes made since last time

General

10. Given the power, how would you improve this process?

____
____
____
____