



UNIVERSITY OF NAIROBI

AGENT-BASED INTEROPERABILITY SYSTEM IN HEALTH INSURANCE

BY

LIECH JAMES GOR

This Research Project is submitted in partial fulfillment for the requirements of the Degree of Master of Science in Computer Science of University of Nairobi

August 2013

DECLARATION

I Liech James Gor do declare that this project, as presented in this report is my own original work and has not been presented anywhere for the purpose of an academic award.

Signature: _____
_____/_____/_____

Date:

Liech James Gor

P58/63063/2011

I Dr. Elisha T.O. Opiyo the University of Nairobi supervisor do confirm that this research project was presented for evaluation with my approval as the University of Nairobi supervisor.

Dr. Elisha T.O. Opiyo

Signature: _____
_____/_____/_____

Date:

ACKNOWLEDGEMENT

I would like to gratefully take this opportunity to thank my Almighty God for his care, grace and good health He offered to me throughout the Master of Science course. During this research project I was aided by many people directly and indirectly through feedback, support and motivation.

I would also like to express my appreciation to Dr. Elisha Opiyo (Supervisor), Mr. Lawrence Muchemi (Panel 5 Chairman), Madam Christine Ronge and Mr. Andrew Mwaura for their support. They have given me important directions on writing this research project. Their advices and comments were always very professional, and concise. Under their supervisory, this project writing became easier and very enjoyable.

I would also wish to convey my warm gratitude to Mr. Benard Gitangi of Insurance Regulatory Authority- Technical Department for his great support to take me through their current system. Also to my Employer NHIF who accorded me enough time to study the current system.

To UAP Insurance, Jubilee Insurance I do say a BIG THANK YOU for your overwhelming support throughout this study.

Abstract

The Information Technology industry has been very dynamic in health insurance for the last couple of decades. In this health insurance firms applications have been developed using different methodologies, and technologies. These applications cannot be integrated, which becomes an issue to health insurance interoperability. Health insurance is available to both individual and groups; there are many insurance companies in Kenya and there is no central database to extract stratified information about them. Attempts by many scholars to develop an interoperable system have not been successful largely due to lack of expertise, resources and the inherent complex nature of the health insurance system. There is also difficulty in accessing the required data by different people due to the fact that the applications of these insurance firms are independent. They are designed in any manner, and can use different technologies. This study proposed an agent-based platform which allowed users to receive and exchange data and information from distributed sources. The health insurance business was used as a case study to implement multi-agents approach in this study. In this case study, different agents were used to represent different functional areas in the developed system. The reactivity, proactive, sociability characteristics of multi-agents achieved health insurance interoperability and accomplished the health insurance business requirements. In this study we reviewed pure theory, evaluated and developed an agent-based system which allowed health insurance firms to share data and also data sharing platform that enabled collaboration among multiple health insurance institutions in Kenya and also the experimental results showed that the platform could be extended easily to support a large number of concurrent client connections.

Table of Contents

Cover Page	i
Declaration	ii
Acknowledgement	iii
Abstract	iv
Table of Contents.....	v
List of Figures.....	ix
List of Tables	x
List of Abbreviations	xi
List of Symbols.....	xiv
1.0 CHAPTER ONE: INTRODUCTION	1
1.1 Background Information	1
1.2 Problem Statement	2
1.3 Purpose of the Project	3
1.4 Objectives	3
1.5 Research Questions	3
1.6 Reasons for Using Multi-Agent System to Provide the Solution	3
1.7 Significance of Study	4
1.8 Research Outcomes.....	4
1.9 Assumptions and Limitations	4
1.10 Scope of the Study	5
1.11 Definitions of Important Terms	5
2.0 CHAPTER TWO: LITERATURE REVIEW.....	9
2.1 Health insurance in Kenya.....	11
2.2 Benefits of Health insurance.....	12
2.3 Health Insurance Institutions in Kenya	12
2.3.1 UAP Insurance	12
2.3.2 National Hospital Insurance Fund (NHIF)	13
2.3.3 Insurance Regulatory Authority.....	13
2.3.4 Functions of IRA.....	14
2.3.5 IRA Services	14
2.4 Benefits of Interoperability.....	16
2.5 Challenges of Interoperability systems	16
2.6 Review of related Work	16

2.7 What are Agents.....	18
2.8 Multi-Agent Environment	19
2.9 Agent approach for interoperability in Health Insurance	20
2.10 Multi-Agent Approach in Health Insurance Business (NHIF)	20
2.11 Multi-agent and Service Oriented Architecture of Health Insurance Business	21
2.12 Agents.....	23
2.13 Agent Architectures and Agent Modelling.....	24
2.13.1 Types of Agent Architecture.....	24
2.13.2 Symbolic agents	24
2.13.3Reactive Agents	25
2.13.4 Hybrid Agents.....	25
2.14 Multi -Agent based systems technology	26
2.14.1 Characteristics of MAS	27
2.15 Multi-Agent System (MAS) Methodologies	27
2.15.1 Features of Agent Methodology	28
2.15.2 Agent Genealogies	28
2.15.3 Agent-oriented Analysis model	29
2.16 Prometheus methodology	29
2.16.1 Stages of Prometheus Methodology.....	30
2.17 Conceptual framework	30
2.17.1 WSIG Architecture.....	32
2.17.2 WSIG Agent	32
2.17.3 Platform Monitor Agent	33
2.17.4 Registration Agent	33
2.17.5 Data Access Agent	33
2.17.6 External System Agents	33
2.17.7 RMA, DF and AMS agents	33
2.17.8 Persistence Agent	34
2.18 Exposing Agent Services as Web Services in this Platform	35
2.19 Agent-based Framework for Integration in Multi-agent Systems	35
2.19.1 JADE	36
2.19.2 The FIPA Platform.....	37

3.0 CHAPTER THREE: METHODOLOGY	38
3.1 Requirement gathering	38
3.2 Data collection process.....	38
3.3 Data collection Tools	38
3.4 Analysis and design of the data sharing platform.....	38
3.5 Coding and debugging	39
3.6 Testing	39
3.7 Evaluation of the multi-agent based system.....	39
3.8 Documentation (Report Writing).....	39
3.9 The Prometheus methodology	40
3.9.1 System specification	41
3.9.2 Architectural System design	41
3.9.3 Detailed design.....	41
3.10 System implementation	41
4.0 CHAPTER 4: ANALYSIS AND DESIGN	42
4.1 Introduction	42
4.1.1 User requirements	42
4.1.2 System requirements	42
4.2 Agent programs.....	43
4.2.1 Registration Agent	43
4.2.2 IRA Data Access Agent.....	43
4.2.3 External System Agents	43
4.2.4 RMA, DF and AMS agents	43
4.2.5 Persistence Agent.....	43
4.2.6 WSIG Agent	43
4.2.7 Platform Monitor Agent	44
4.3 Analysis using PROMETHEUS concepts.....	44
4.3.1 User interfaces	46
4.3.2 Analysis Model and View	46
4.4 System Specification.....	47
4.4.1 Analysis Overview Diagram.....	48
4.4.2 System description	48
4.4.3 Goals.....	48
4.4.4 Overall Goal.....	49

4.4.5 Goal Diagram.....	49
4.4.6 Use Case Scenarios	49
4.4.7 Identification of the Systems Interface to the Environment	50
4.4.8 Actors.....	50
4.4.9 System Functionalities	51
4.4.9.1 Health Insurance Registration Functionality	51
4.4.9.2 Sending Alerts.....	52
4.4.9.1 Searching Existing Health Insurance Firms Records.....	52
4.5 Architectural Design.....	54
4.5.1 Data coupling diagram	55
4.5.2 Agent Descriptors.....	55
4.5.3 Interaction Diagram of the System	57
4.5.4 Agent-Role Grouping	58
4.5.5 Agent Acquaintance	58
4.5.6 System Overview Diagram.....	59
4.6 Detailed Design.....	60
4.6.1 Agent Capabilities	61
4.6.2 Capability Descriptors.....	61
4.6.3 Agent Overview diagram.....	62
5.0 CHAPTER FIVE: IMPLEMENTATION AND RESULTS.....	63
5.1 Implementation tools.....	63
5.2 System testing and evaluation	63
5.2.1 Component Testing	64
5.2.2 Integration testing	64
5.2.3 System Testing	64
5.2.4 Acceptance Testing.....	65
5.2.5 Performance Testing.....	65
5.2.6 Volume Testing.....	65
5.2.7 Regression Testing	65
5.2.8 Evaluation Testing.....	65
5.3 Tests for the Developed Systems	66
5.3.1 IRA MAS.....	66
5.3.2 Tests for IRA System Login screen.....	67
5.3.3 Test Search for Policy Holders named containing letter “m	69

5.3.4 Test Search Result for Files or Folders named containing letter “m”	69
5.4 Agent Communication and Interaction Protocol	69
5.5 Evaluation of the multi-agent based system	70
5.6 Discussion of Results	71
6.0 CHAPTER SIX: CONCLUSION AND FUTURE WORK	72
6.1 Conclusion	72
6.2 Recommendation for Future Work.....	72
6.3 Challenges	72
APPENDICES	72
Appendix I References.....	72
Appendix II Installation Manual.....	77
Appendix III Introduction Letter to IRA.....	79
Appendix IV Introduction Letter to NHIF	80
Appendix V Introduction Letter to Jubilee Insurance.....	81
Appendix VI Introduction Letter to UAP Insurance.....	82
Appendix VII Authority Letter from NHIF.....	83
Appendix VIII Authority Letter from IRA.....	84
Appendix IX Hospital Claim Form P1	85
Appendix X General Claim Form.....	87
Appendix XI Code Samples	88
Appendix XII Screen Shots.....	99

LIST OF FIGURES

Figure 2.1: Driving forces of Health Insurance Interoperability	10
.....	
Figure 2.2: System Architecture for an Agent-based Multimedia Data Sharing Platform ..	17
Figure 2.3: Agents and the Environment	19
Figure 2.4: The Architecture of Health Insurance Businesses	22
Figure 2.5: Layered Architecture.....	26
Figure 2.6: Agent Genealogies (James Odell (2005)).....	28
Figure 2.7: Agent Oriented Analysis Model	29
Figure 2.8: Phases of Prometheus Development	30
Figure 2.9: Conceptual Framework	31
Figure 2.10: WSIG Architecture.....	33
.....	
Figure 2.11: FIPA Reference Model of an Agent Platform	36
Figure 2.12: The FIPA Platform.....	37
Figure 4.1: System Specification Phase Diagram.....	46
Figure 4.2: Analysis Overview Diagram.....	47
Figure 4.3: Goal Diagram.....	49
Figure 4.4: Use Case Scenario 1	49
Figure 4.5: Use Case Scenario 2.....	50
Figure 4.6: Use Case Scenario 3.....	50
Figure 4.7: Architectural design Phase Diagram.....	54
Figure 4.8: Data Coupling Diagram.....	55
Figure 4.9: Interaction Diagram of the system	57
Figure 4.10: Agent-Role Grouping Diagram.....	58
Figure 4.11: Agent Acquaintance Diagram.....	58
Figure 4.12: System Overview Diagram.....	59
Figure 4.13: Detailed Design Phase Diagram	60
Figure 4.14: Agent Overview Diagram.....	62
Figure 5.1: IRA MAS Registered Agents	66
Figure 5.2: Screen Shot-Login to IRA System.....	67
Figure 5.3: Search for Policy Holders results.....	68
Figure 5.4: Agent Communication and Interaction Protocol	69

LIST OF TABLES

Table 4.1: Exposing Agent Services as Web Services.....	35
.....	
Table 4.2: Health Insurance Registration Functionality.....	52
Table 4.3: Sending Alert Messages.....	52
Table 4.4: Searching Existing Health Insurance Firms Records	52
Table 4.5: Subscription.....	52
Table 4.6: Agent Configuration	53
Table 4.7: DF to WSDL Converter	54
Table 4.8: UDDI Repository Configuration	54
Table 4.9: UDDIj Configuration.....	54
Table 4.10: Ontologies	54
Table 4.11: Agent Descriptors	55
Table 4.12: Agents Capabilities	61
Table 4.13: Capability Descriptors	61
Table 5.1: Tests for the Developed systems	66
Table 5.2: Search results table	69

LIST OF ABBREVIATIONS

ACC: Agent Communication Channel
ACL: Agent Communication Language
AMS: Agent Management System
AOSE: Agent Software Oriented Engineering
AP: Agent Platform
API: Application Programming Interface
ARB: Agent Resource Broker
Authority: Insurance Regulatory Authority
CA: Communicative Act
CORBA: Common Object Request Broker Architecture
DCOM: Distributed COM
DF: Directory Facilitator
FIPA: Foundation for Intelligent Physical Agents
FP: Feasibility Precondition
GUID: Global Unique Identifier
HAP: Home Agent Platform
HTTP: Hypertext Transmission Protocol
IDL: Interface Definition Language
IIOP: Internet Inter-ORB Protocol
Industry: Health Insurance industry
IRA: Insurance Regulatory Authority
JADE: Java Agent Development
JDK: Java Development Kit
JSP: Java Server Pages
MAS: Multi-agent System
MEP: Message Exchange Pattern
MTP: Message Transport Protocol
MTS: Message Transport Service
NHIF: National Hospital Insurance Fund
OMG: Object Management Group
ORB: Object Request Broker
PY: python

RE: Rational Effect

RMI: Remote Method Invocation, an inter-process communication method embodied in Java

SL: Semantic Language

SMTP: Simple Mail Transfer Protocol

SOA: Service-Oriented Architecture

SOAP: Simple Object Access Protocol

Tcl: Tool Command Language

[Tk](#): GUI toolkit

TCP / IP: Transmission Control Protocol / Internet Protocol

TILAB: Telecom Italia Lab (TILAB)

UDDI: Universal Description Discovery and Integration

UAP: UAP Insurance

W3C: World Wide Web Consortium

WSDL: Web Services Description Language

WSIG: Web Services Integration Gateway

XML: Extensible Markup Language

LIST OF SYMBOLS

Entities

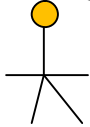
Action

An action is what the agent does that effects the environment. It is denoted by a shape as shown below.



Agent

An agent is denoted by a shape as shown below.



Capability

A capability is different roles of an agent grouped. It can be considered as a part of an agent. It is denoted by a shape as shown below.



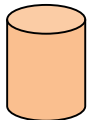
Message

A message which goes from one agent to another is denoted by a shape as shown below. Messages are usually added in a top down order. For example, if you create a message in an agent overview diagram you cannot use this particular message in the system overview diagram. You can create a new message of the same type in the system overview diagram and if you create an edge between the agent and the message a new message (related to the one in the system overview diagram) will automatically be created in the agent.



Data

A data store is used to store beliefs of an agent. It is denoted by a shape as shown below. (If data is added within an agent, and then later a decision is made to include it as shared data, shown within the System Overview, then the copy within the agent will need to be manually removed to avoid having 2 copies on the diagram, of the same underlying object).



Role

A role is denoted by a shape as shown below.



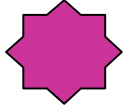
Goal

A goal is denoted by a shape as shown below.



Percept

A percept is the input coming from the environment to the agent. It is denoted by a shape as shown below.



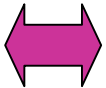
Plan

A plan is denoted by a shape as shown below.



Protocol

A protocol which is a specification of allowable agent interaction sequences, within a given conversation, is shown as below.



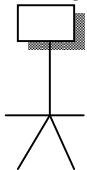
Scenario

A scenario, which is an abstract description of a particular sequence of steps within the system, is denoted by a shape as shown below.



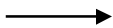
Actor

An actor which is an entity (human or software/hardware) external to the system being designed is denoted by a shape as shown below.



Edge

An edge which is used to



1.0 CHAPTER ONE: INTRODUCTION

The Insurance Regulatory Authority works collectively and individually with health insurance industry players and in line with the Insurance Act, the functions of IRA are to ensure compliance by insurance/reinsurance companies and intermediaries with legal requirements and sound business practices so an updated database of policy holders has wide statistical applications in business, social issues and government institutions to promote efficient, fair, safe and stable markets. Many organizations dealing with health insurance such as NHIF, UAP Insurance, Jubilee Insurance, among others capture data that is relevant to their operations and hardly create interfaces to other stakeholder databases. They constantly add new clients and remove dormant ones in order to keep their databases up to date. This research project focused on an agent-based platform, the system allows IRA to receive and exchange data from distributed sources. This interoperability system makes public and private health insurance data accessible to other registered insurance firms. It optimizes the management and timely dissemination of relevant information and data-driven services to citizens, professional and internal Government audiences, in self-service mode and across multiple channels. This research project involves theory review, evaluation and prototype development, it also reviews pure theory and evaluate its applicability in various or different contexts. This study involves methodical use of a conceptual/explanatory framework towards solving a clearly defined problem. The interoperability system proposed an improvement of the conceptual framework as evidenced from data. This study involved development of a prototype system that implements the proposed solution/conceptual framework to demonstrate its application in a real-world setting.

1.1 BACKGROUND INFORMATION

The use of ICT in public and private organizations has increased significantly. People in these sectors appreciate the usefulness of computers in terms of information processing and dissemination. This improves service delivery, and productivity. The health insurance sector is embracing the use of ICT in its operations.

Interoperability system is a rapid growing industry. The Information Technology industry has been very dynamic for the last couple of decades. Applications have been developed using different methodologies, and technologies. These applications cannot be integrated, which becomes an issue to interoperability in health insurance. The IT industry has worked out different tools to try to solve the interoperability issues. However, these tools do not turn out to be very successful, as

they cannot solve the interoperability requirements of an open system. Multi-agents system is characterized by abstraction, interoperability, modularity and dynamism. These qualities are particularly useful to promote open system. The agents' modularity allows a complex system to be broken down into simpler subsystems, so that processes inside the subsystems are simple, and that the processes between subsystems are manageable. The architecture of modularity also provides the benefits of flexibility, compatibility, and accessibility, and thus improved the interoperability between subsystems. The abstraction of agents gives the architect the abilities to easier design, troubleshoot problems. This project proposal provides a list of health insurance interoperability evaluation criteria on the categories of process, data and application architecture, technical architecture, and technology. The health insurance business is used as a case study to implement multi-agents approach. In this proposal, different agents are used to represent different functional areas. The reactivity, proactive, sociability characteristics of multi-agents achieves health insurance interoperability and accomplishes the health insurance business requirements. In this study, multi-agent based software technology will be used to implement the proposed multi-agent based Interoperability system in health insurance.

1.2 PROBLEM STATEMENT

Health insurance is not new in Kenya context and is slowly catching up with the consumers. Consumers understand the objective of health insurance and its offering to cover the ever rising medical expenses. Health insurance is available to both individual and groups; there are many insurance companies in Kenya and there is no central database to extract stratified information about them. Attempts by many scholars to develop an interoperable system have not been successful largely due to lack of expertise, resources and the inherent complex nature of the health insurance system. There is also difficulty in accessing the required data by different people due to the fact that the applications of these insurance firms are independent. They are designed in any manner, and can use different technologies.

In Kenya, there is no system which has the ability to accept and allow other system of different insurance firms to use the parts or equipment of another system, or the ability of two or more systems or components to exchange information and to predictably use the information that has been exchanged. Lack of a common central database server or a common platform to assure these insurance firms that access to sensitive or critical information is not lost, destroyed, misappropriated or corrupted and to assure that continuity of operations can be maintained subsequent to a deliberate or natural catastrophic event.

1.3 PURPOSE OF THE PROJECT

The research project intends to build a fully automated prototype of an interoperability system using multi-agent technology. The main goal of the system is to provide a fully automated system for health insurance providers. This research will explore an agent based interoperability system for data sharing for health insurance providers. To provide cost-effective health insurance system integration solution, this paper presents a Graphical User Interface state model for automatically exchanging information with existing health insurance software through their GUIs with no modifications needed to them. This paper proposes a framework that helps us to share not only the data in these health insurance institutions but also the data management platform in order to promote collaboration across multiple institutions. This research project evaluates the Multi-Agent based approach as an appropriate methodology to achieve health insurance interoperability.

1.4 OBJECTIVES

- i. To find out existing data sharing methods and their problems
- ii. To understand how a data sharing platform/interoperability system can be implemented
- iii. To find out if software agents can be used to implement a data sharing platform
- iv. Try to create a prototype model for experimentation
- v. To develop an agent-based interoperability system that provides data sharing, data exchange capabilities using agents

1.5 RESEARCH QUESTIONS

In this Agent-based Interoperability System in Health Insurance, the following two research questions would be addressed

- i. What is the feasibility of using an agent-based approach to build an interoperable system in health insurance?
- ii. What is the nature of the system built in terms of its effectiveness?

1.6 Reasons for using Multi-agent system to provide the solution

Nowadays, agent-based technologies are considered the most promising means to deploy enterprise wide and world wide applications that often must operate across corporations and continents and inter-operate with other heterogeneous systems. It is because they offer the high-level software abstractions needed to manage complex applications and because they were invented to cope with distribution and interoperability. Health insurance is widely distributed and complex hence the need for multi-agent system that is suitability for distributed applications. Data received from various health insurance providers by the Insurance Regulatory Authority

(IRA) does not depend on other that is ability to act autonomously. The use of multi-agent system will promote reusability hence reduce cost during registration and approval of health insurance institutions. The integration of different system from these insurance firms makes the system to be complicated hence the suitability of multi-agent systems. Automatic online registration of services by health insurance firms requires reasoning which can be best achieved by agents that is the ability to work co-operatively in teams. High-level representation of behavior, a level of abstraction above object-oriented constructs is considered most appropriate. There is also need to provide flexibility, which is pro-activeness and reactive behavioral characteristics. The prototype requires enhancement of real-time performance.

1.7 SIGNIFICANCE OF STUDY

This study proposes a framework that helps to share not only the data in these health insurance institutions but also the data management platform in order to promote collaboration across multiple health insurance institutions. For health professionals, the study improves access to health record data and health information anytime, anywhere. For patients, this study improves quality and safety of care by improving data exchange, the quality of data flow and access to information by health professionals thereby potentially reducing errors. For health managers, the study improves data collection and facilitates statistical and economic analysis and for health researchers, this research study improves and increases the availability of medical data.

1.8 RESEARCH OUTCOMES

The deliverables for this study have provided the solutions to the stated research problems. The existing system in Insurance Regulatory Authority is long and is prone to human errors; these errors are due to some instances where these procedures are handled manually by human. The system will be a prototype that will help the public and private sectors in Kenya to share data and data management platforms. In this study the documentation and publishing of processes in various health insurance firms will be done. The deliverables of the study included contribution of knowledge and a well published paper.

1.9 ASSUMPTIONS AND LIMITATIONS

This study assumed that there was a functional database in each and every health insurance institution that could be queried to find records and to extract statistics. This research study also assumed that there were computers and wide area network in every insurance firm. This study included the major health insurance firms in Kenya. The study was limited to methodical use of

a conceptual/explanatory framework towards solving a clearly defined problem. The interoperability system proposes an improvement of the conceptual framework as evidenced from data. This study involved development of a prototype system that implemented the proposed solution/conceptual framework to demonstrate its application in a real-world setting.

1.10 SCOPE OF THE STUDY

This study concentrated on analysis, design and implementation of a multi-agent system for Insurance Regulatory Authority. The application has been restricted to only health insurance.

1.11 DEFINITIONS OF IMPORTANT TERMS

The important terms in this project proposal include the following:-

Health insurance is popularly known as Medical Insurance or Mediclaim is a type of insurance that covers your medical expenses.

Agent is a computer system capable of autonomous action in some environment in order to meet its design objectives (Wooldridge 2002)

A Multi-agent system is one that consists of a number of agents, which interact with one another. In most general scenarios, agents will be acting on behalf of users with different goals and motivations. To successfully interact, they will require the ability to cooperate, coordinate and negotiate with each other, much as people do (Wooldridge, 2002).

Interoperability is the capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units. (IEEE, USA)

Hybrid agent is an approach to agent design that incorporates both the deliberative and reactive components (Wooldridge, 2002)

Coordination is how agents interact, which includes methods of communication and cooperation between agents when achieving their tasks.

Control-driven coordination model, this is where the agents interact with external world by generating and handling events on well defined input and output ports.

Data-driven coordination model, this is where the coordinables interact with external world by exchanging data structures through the coordination media.

The Actor, Role and Coordinator(ARC) model, this uses active objects to model asynchronous and distributed computations

JADE –This is Java Agent Development Framework, is a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middle-ware that complies with FIPA specifications and through a set of graphical tools that support the debugging and deployment phases.

FIPA- Foundation for Intelligent Physical Agents. The FIPA 97 specification is the first output of the Foundation for Intelligent Physical Agents. It provides specification of basic agent technologies that can be integrated by agent systems developers to make complex systems with a high degree of interoperability. FIPA specifies the interfaces of the different components in the environment with which an agent can interact, i.e. humans, other agents, non-agent software and the physical world. FIPA Agent Communication specifications deal with Agent Communication Language (ACL) messages, message exchange interaction protocols, speech act theory-based communicative acts and content language representations.

SOAP (Simple Object Access Protocol), is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks

UDDI –(Universal Description, Discovery and Integration) is a platform-independent framework for describing services, discovering businesses, and integrating business services by using the internet.

Java Agent Development Framework -is a software framework for multi-agent systems, to develop distributed agent-based applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems. The JADE platform allows the coordination of multiple FIPA-compliant agents and the use of the standard FIPA-ACL communication language in both SL and XML. It is a software framework to write agent applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems.

Action -A basic construct which represents some activity which an agent may perform. A special class of actions is the communicative acts.

ARB Agent -An agent which provides the Agent Resource Broker (ARB) service. There must be at least one such an agent in each Agent Platform in order to allow the sharing of non-agent services.

Agent -An Agent is the fundamental actor in a domain. It combines one or more service capabilities into a unified and integrated execution model which can include access to external software, human users and communication facilities.

Agent Communication Language (ACL) -A language with precisely defined syntax, semantics and pragmatics that is the basis of communication between independently designed and developed software agents. ACL is the primary subject of this part of the FIPA specification.

Agent Communication Channel (ACC) Router -The Agent Communication Channel is an agent which uses information provided by the Agent Management System to route messages between agents within the platform and to agents resident on other platforms.

Agent Management System (AMS) -The Agent Management System is an agent which manages the creation, deletion, suspension, resumption, authentication and migration of agents on the agent platform and provides a “white pages” directory service for all agents resident on an agent platform. It stores the mapping between globally unique agent names (or GUID) and local transport addresses used by the platform.

Agent Platform (AP) -An Agent Platform provides an infrastructure in which agents can be deployed. An agent must be registered on a platform in order to interact with other agents on that platform or indeed other platforms. An AP consists of three capability sets ACC, AMS and default Directory Facilitator.

Communicative Act (CA) -A special class of actions that correspond to the basic building blocks of dialogue between agents. A communicative act has a well-defined, declarative meaning independent of the content of any given act. CA's are modelled on speech act theory. Pragmatically, CA's are performed by an agent sending a message to another agent, using the message format described in this specification.

Content -That part of a communicative act which represents the domain dependent component of the communication. Note that "the content of a message" does not refer to "everything within the message, including the delimiters", as it does in some languages, but rather specifically to the domain specific component. In the ACL semantic model, a content expression may be composed from propositions, actions or IRE's.

Conversation -An ongoing sequence of communicative acts exchanged between two (or more) agents relating to some ongoing topic of discourse. A conversation may (perhaps implicitly) accumulate context which is used to determine the meaning of later messages in the conversation.

Software System -A software entity which is not conformant to the FIPA Agent Management specification.

CORBA -*Common Object Request Broker Architecture*, an established standard allowing object-oriented distributed systems to communicate through the remote invocation of object methods.

Directory Facilitator (DF) -The Directory facilitator is an agent which provides a “yellow pages” directory service for the agents. It stores descriptions of the agents and the services they offer.

Feasibility Precondition (FP) – The conditions (i.e. one or more propositions) which need be true before an agent can (plan to) execute an action.

Ontology – ontologies are the vocabularies that are used to ensure that agents use the same terms with the same meanings. The definition of terms by ontologies is necessary for the agent to interpret the content of a single message.

Ontology sharing problem -The problem of ensuring that two agents who wish to converse do, in fact, share a common ontology for the domain of discourse.

Interaction Protocol (IPs) – (also called conversation policies in the multi-agent systems research community) are a tool to help agent developers model agent communication.

2.0 CHAPTER TWO: LITERATURE REVIEW

Interoperability is the capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units. In the context of information systems, interoperability is the ability of different types of computers, networks, operating systems, and applications to work together effectively, without prior communication, in order to exchange information in a useful and meaningful way (Inproteo 2005).

According to ISO/IEC 2382-01, *Information Technology Vocabulary, Fundamental Terms*, interoperability is defined as follows: *"The capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units"*.

The effective management and sharing of information across agency boundaries will result in information being used more efficiently and effectively. This will provide significant benefits, including:

- reduced costs of information collection and management through streamlined collection, processing and storage;
- improved decision making for policy and business processes, resulting in more integrated planning and enhanced government service delivery;
- improved timeliness, consistency and quality of government responses –information will be easily accessible, relevant, accurate, and complete;
- improved accountability and transparency for citizens;
- reduced costs and added value for government through reusing existing information, sharing infrastructure and designing integrated, collaborative methods of delivering services;
- improved national and jurisdictional competitiveness; and improved national jurisdictional security. Abbas Rajabifard (2010)

Interoperability is about integration. Institute of Electrical and Electronic Engineers (IEEE) defined interoperability as “the ability of two or more systems or components to exchange information and to use the information that has been exchanged.” It is the ability of the systems in an enterprise to integrate with other systems, which can be internal or external to the enterprise”.

Vassilios Karakoidas et al.(2004) have identified four driving forces that affect enterprise interoperability.

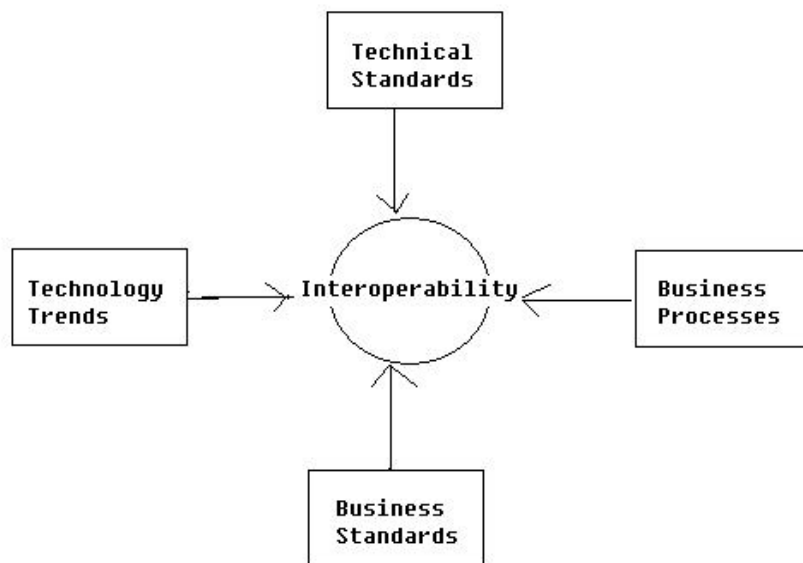
Technical Standards – In order to achieve interoperability, the enterprise has to develop standards for the technologies used in the enterprise. In most enterprises, the technical technologies may be from the old legacy systems still in use in the enterprise, to all other newer applications. In e-commerce systems, the technical standards of the business partner will also have to be considered.

Technology Trends – The enterprise will have to consider the technology trends to achieve interoperability today and in the future. This is particular important, if an enterprise needs to stay integrated with their business partners in an e-commerce environment.

Business standards – Business standards are common language between businesses. It is the common understanding of how the business will communicate. It is not about technologies. The business components in an enterprise have to adapt the same business standards in order to be interoperable. One difficult part is to adopt business standards with the business partners.

Business processes – The business processes is very important to achieve interoperability in an enterprise. The business processes will include manual process and processes in the technologies used in the enterprise. Without clear and well-defined business processes, an enterprise will never integrate. Again, the business processes with the business partners are important to stay integrated in e-commerce. **The figure 1**below shows the driving forces of health insurance Interoperability.

Figure 2.1



The healthcare area is extremely complex due to the new means of diagnostic and therapeutic, new procedures as well as the existence of various professional groups, each one with different characteristics, requirements and ways of working. All of this affects the way information systems in healthcare are implemented and used (Lammintakanen et al, 2010). For a better decision-making process, especially in a critical area as healthcare, fast and reliable access of a patient’s medical history is of utmost importance (Maass et al, 2008; Revere et al, 2007), even

though the information is located in a different information system geographically distant from the information system in use in a given situation. There is an urgent need for the creation of integration mechanisms among the different health information systems that exist today in different healthcare units, like what happens in other areas of activity (Manpaa et al, 2009). A connection exists between the quality of care provided to a patient and the exchange of information among different levels of healthcare (Maass et al, 2008), like between primary care and differentiated care in the Portuguese National Health Service. According to the European Commission (2008), the interoperability between health information systems is of utmost importance and essential for a better health service management, public health, quality and safety of care to patients and clinical research.

Interoperability biggest advantage in a critical and important area as healthcare is increased information available in one of the most critical times if not the most critique and the decision making process (Cho et al, 2010). The bigger and better the amount of information in a given situation, the better the care provided by health professionals and the lower the number of errors (Manpaa et al, 2009).

2.1 Health insurance in Kenya

Health insurance (popularly known as Medical Insurance or Mediclaim) protects you and your dependents against any financial constraints arising on account of a medical emergency. It sometimes includes disability and long term medical needs. In Mediclaim, you pay a premium and in return the insurer commits to pay a predetermined sum of money to meet the claims. Health insurance is not new in Kenya context and is slowly catching up with the consumers. Consumers understand the objective of health insurance and its offering to cover the ever rising medical expenses. Health insurance is available to both individual and groups. However, premium for individual policy is costlier than that of the group policy. An individual is the owner of his personal policy. Whereas in group plans, the sponsor is the owner of the policy and the registered members are covered by the policy. You can take advantage of group health insurance to overcome the shortage of your individual insurance. People with no policy or are uninsurable due to one or the other reason can take good advantage of the group plans and be covered.

2.2 Benefits of Health insurance

Benefit depends on the policy you choose and the coverage it provides. Here is a list of basic coverage provided by most of the health policies.

1. It helps securing a better future by paying a fraction as an expense today called the premium.
2. It reduces saving huge amount of financial losses, risk of financial breakdown in case of expensive medical and post-illness care.
3. It definitely induces a sense of security to the insured.
4. It provides financial security to the family members.
5. It covers your hospitalization and medical bills.
6. It also covers disability and custodial bills.
7. You can avail tax benefits on the premium paid under section 80d of the Income Tax Act.
8. The best factor, you can also opt for health insurance policies even after the age of 60.

The Institute of Electrical and Electronics Engineers (IEEE, USA) defines interoperability as: "the ability of two or more systems or components to exchange information and to use the information that has been exchanged". [IEEE-USA].

In Europe, IDABC - Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens - offers the following similar definition (edited for clarity): "Interoperability means the ability of information and communication technology (ICT) systems, to exchange data and enable the sharing of information and knowledge." [IDABC] The National Alliance for Health Information Technology (NAHIT, USA) expands a little on the above definitions: "In healthcare, interoperability is the ability of different information technology systems and software applications to communicate, to exchange data accurately, effectively, and consistently, and to use the information that has been exchanged. " [NAHIT]

2.3 Health Insurance Institutions

In Kenya there are many insurance firms offering health insurance policies, they include the following:-

2.3.1 UAP Insurance

UAP has two important products [1]Afyaimara which is an inpatient and outpatient health insurance for you and your family, in UAP Insurance the Afyaimara has the following features:

- i. No excess for inpatient
- ii. Covers pre-existing conditions

- iii. Cover for chronic conditions & HIV/AIDS
- iv. Countrywide provider network

Another product of UAP insurance is the [2] Individual Health Insurance, this individual MAXIMED is a comprehensive and flexible inpatient medical insurance product developed to meet healthcare needs of an individual or a family through delivery of quality, convenient user-friendly services. The cover provides comprehensive and enhanced benefits to cater for hospitalization and approved day-care treatment costs.

2.3.2 National Hospital Insurance Fund (NHIF)

NHIF is a State corporation that was established in 1966 as a department under the Ministry of Health. The original Act of Parliament that set up this Fund in 1966 has over the years been reviewed to accommodate the changing health care needs of the Kenyan population, employment and restructuring in the health sector.

Currently an NHIF Act No 9 of 1998 governs the Fund. The transformation of NHIF from a department of the Ministry of Health to a state corporation was aimed at improving effectiveness and efficiency. The Fund's core mandate is to provide medical insurance cover to all its members and their declared dependants (spouse and children). The NHIF membership is open to all Kenyans who have attained the age of 18 years and have a monthly income of more than Ksh 1000. NHIF has more than 30 fully autonomous branches across the country. Each of these branches offers all NHIF services including payment of benefits to hospitals or members or employers. Smaller satellite offices and service points in district hospitals also serve these branches.

2.3.3 Insurance Regulatory Authority

This is a statutory government agency established under the Insurance Act (Amendment) 2006, CAP 487 of the Laws of Kenya to regulate, supervise and develop the insurance industry. It is governed by a Board of Directors which is vested with the fiduciary responsibility overseeing operations of the Authority and ensuring that they are consistent with provisions of the Insurance Act. The Authority is a precursor to the then Office of the Commissioner of Insurance that came into existence with the enactment of the Insurance Act, CAP 487 in 1986. (<http://www.ira.go.ke>)

Prior to this, insurance regulation was based on the UK legislation under the Companies Act 1960. In executing their mandate, the IRA adheres to the core principles of objectivity,

accountability and transparency in promoting not only compliance with the Insurance Act and other legal requirements by insurance/reinsurance companies and intermediaries but also sound business practices. IRA therefore practices regulation and supervision that enables health insurance industry players to be innovative and entrepreneurial. Bearing in mind industry differences in terms of size, extent and complexity, necessitating changes in operating and investment decisions helps cut down on compliance costs. Since in the long run, this has impacts on productivity and growth of the insurance sector, the IRA deploys significant resources in monitoring market behaviors, compliance and solvency issues.

Functions of IRA

The Authority works collectively and individually with health insurance industry players and in line with the Insurance Act, the functions of IRA are to:

- i. Ensure compliance by insurance/reinsurance companies and intermediaries with legal requirements and sound business practices;
- ii. Promote voluntary compliance;
- iii. Set clear objectives and standards of intervention for insurance/reinsurance companies and intermediaries or type of intervention;
- iv. Protect consumers and promote high degree of security for policyholders;
- v. Promote efficient, fair, safe and stable markets;
- vi. Maintain the confidence of consumers in the market;
- vii. Ensure insurance/reinsurance companies and intermediaries remain operationally viable and solvent; and
- viii. Establish a transparent basis for timely, appropriate and consistent supervisory intervention, including enforcement.

2.3.4 IRA Services

Regulation

As an Authority, IRA recognizes that regulation as a means to an important end is essential for the proper functioning of the insurance industry. To ensure that health insurance industry reaps the benefits from a globally competitive financial services sector, the sector has to remain efficient, flexible and responsive to emerging trends i.e. effective in addressing identified problem and efficient in maximizing benefits to the economy. While issuing guidelines and circulars to the health insurance industry, IRA is often alive key drivers for enhancing industry productivity such as technological change (adopting new knowledge), investment in human and physical capital.

Regulations used for the insurance industry in Kenya include the Insurance Act Cap 487 and its accompanying Schedules and Regulations. Circulars and Guidelines are normally issued by the Commissioner of Insurance/Chief Executive Officer of IRA detailing provisions that insurance/re-insurance companies and intermediaries need to comply with. These rules are issued with an expectation of compliance that is done through surveillance and inspections.

Supervision

IRA has the primary responsibility for supervising the insurance industry. Supervision entails the enforcement of rules and standards guiding the conduct of insurance business in Kenya.

This is done mainly through surveillance and compliance sections of the Technical Division. IRA approach to insurance supervision is normally through off-site and on-site inspections. To gain deeper insights, IRA undertakes due diligence of the companies through on-site inspections and surveillance by visiting company offices. In such situations, companies present documents required by IRA for inspection on demand.

The aim of such inspections is ascertain compliance levels with regulations in order to ascertain financial and economic status of the companies or intermediaries.

In addition, not only are insurers and reinsurers expected to submit accounts for public scrutiny and information but also submit reports to IRA for inspection as well as for statistical and policy purposes.

IRA approach to supervision is evolving to be in tandem with developments in the industry locally, regionally and internationally through benchmarking. It is therefore more intensive and intrusive. The purpose of this platform is to ensure that the decisions taken by insurance/re-insurance companies and intermediaries do not pose risks to the statutory objectives.

Consumer Education

IRA recognizes that policy holders and potential policy holders have a right to deal with honest, trustworthy and knowledgeable insurers and intermediaries. Considering that insurers and intermediaries have a greater knowledge of insurance issues than the majority of policy holders, flow of information may be distorted. This is because by the very nature of insurance business, consumers may not be able to detect contracts that could be biased in favour of insurers or which may be interpreted to favour the insurer or simply fail to meet their needs. In addition, marketing methods could place potential policyholders under pressure to make a purchase decision. Through consumer education programmes, this data sharing platform seeks to sensitize the public on the importance of insurance, the reasons as to why one should have an insurance cover, types of products available on the market. Providing this information through this

interoperability system will not only improve insurer, intermediary and consumer relationships but also grow the insurance industry by strengthening consumer confidence.

2.4 Benefits of Interoperability

For IRA: It will ease the services like regulation, supervision, consumer education and protection, launching of claims by policy holders, knowing number of policy holders, ensuring compliance by health insurance companies with legal requirements and sound business practices, promote voluntary compliance.

For health professionals: It improves access to health record data and health information anytime, anywhere.

For patients: Improves quality and safety of care by improving data exchange, the quality of data flow and access to information by health professionals thereby potentially reducing errors.

For health managers: Improves data collection and facilitate statistical and economic analysis.

For health researchers: It improves and increases the availability of medical data.

For the healthcare technology industry: It improves access to the healthcare market for more companies (SMEs in particular who may be limited in their ability to provide technologies which can integrate with an organization's legacy systems). (Brailer DJ. Interoperability: the key to the future health care system. Health Aff (Millwood). 2005)

2.5 Challenges of Interoperability systems

The challenge associated to the creation of interoperable health information systems is the representation of medical information that is very technical and complex (Raghupathi and Umar, 2008) so that the different systems may understand one another, e.g. how to represent the result of a procedure, or how to represent a given medical situation or pathology. One solution is the use of recommendations/standards such as the CEN/ISO EN13606, Health Level 7 (HL7) and the International Classification of Diseases (ICD) (Nores et al, 2011; Wright and Sittig, 2008).

Other important challenges are: knowing if the information is reliable, if it comes from the right place, who can view it, if the information is delivered to those who requested it, which security policies are created and applied (Gritzalis and Lambrinoudakis, 2004). The contextualization of the information that the system tries to obtain is another challenge. How to formally define in which context a given information, is more important than another. How to filter all the information available is another problem in the creation of interoperable systems.

Another problem is the use of mobile devices to access private and sensible information, such as medical information. The device must have hardware and software capabilities to support the security requirements needed to access the different systems, e.g. encryption protocols. To

implement complex business systems, like medical information systems (Liu et al, 2009) and in order to enhance the above challenge mentioned earlier, one option to consider is an SOA (Service Oriented Architecture), this will be seen in the next paragraph

2.6 Review of related Work

Anson Lee, Michael Lyu, Irwin King (2001), Agent-based Multimedia Data Sharing Platform, proposed this agent-based platform which allowed users to receive and exchange multimedia data from distributed sources e.g. digital libraries, image databases and video databases, however, the multitude, diversity and the dynamic nature of multimedia data on the internet made it difficult for them to access any specific piece of multimedia data, in this study the platform was extended easily to support a large number of concurrent client connections and applications.

This platform was a multi-agent system in which three semi-autonomous agents interacted or worked together to perform a user's goal. They were the Server Agent (SA), Query Agent (QA) and Database Agent (DA). The figure 2 below shows the complete system architecture and the relationships between the platform and the client computers.

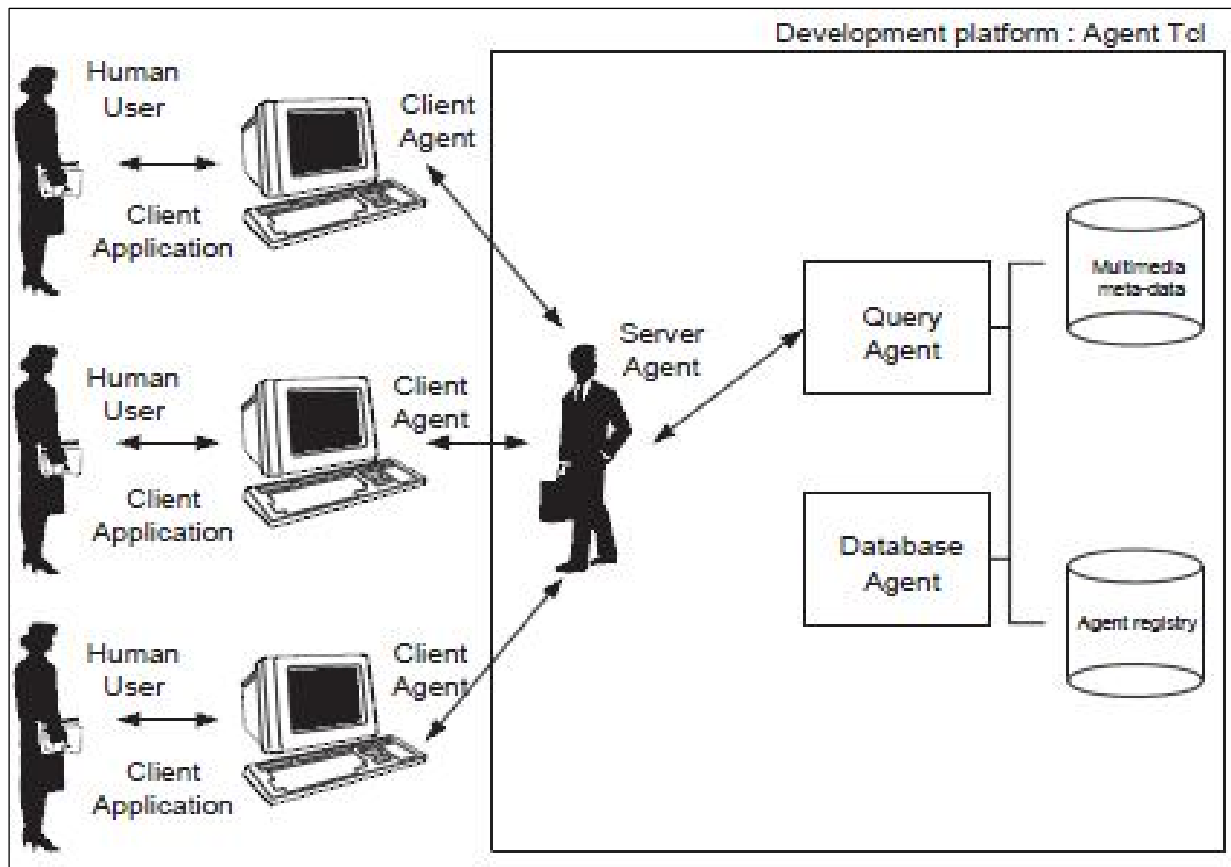


Figure 2.2: System Architecture of the Agent-based Multimedia Data sharing Platform

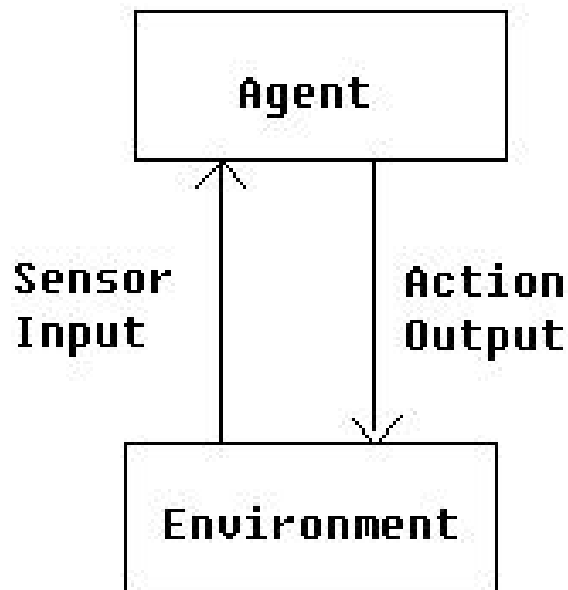
Paul Ho (October, 2008), Agent-based Approach to Interoperability; Using their evaluation criteria, it could be shown that the multi-agent approach was better than object approach to achieve enterprise interoperability. In this study the health insurance business was used as a case study to implement multi-agents approach. In this case study, different agents were used to represent different functional areas in the business. The reactivity, proactive, sociability characteristics of multi-agents achieved enterprise interoperability and accomplished the health insurance business requirements. Using their evaluation criteria, this essay shows that the health insurance agent system was flexible to the changing business needs, very easy to interact with outside organizations, and provided timely and quality information. The study came up with the architecture of Health Insurance business as shown in figure 2 below. The architecture had nine agents; Subscriber personal agent, Enrollment agent, Adjudication agent, Underwriting agent, Planning agent, Monitoring agent, Corporate profile agent, Approval agent, and Staff personal agent.

Yuk-Hei Lam, Zili Zhang and Kok-Leong Ong (2000), Insurance Services in Multi-agent Systems, in this study, the system emphasized that in a multi-agent environment, there was often need for an agent to cooperate with each other so as to ensure that a given task was achieved timely and cost-effectively, in our study, the prototype ensured that agents cooperated with each other so as to achieve some tasks in a timely and cost-effectively manner and also to maximize this through mechanisms such as trust and risk assessments.

2.7 What are Agents

Wooldridge and Jennings defined an agent as ‘a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives’. Notice that his definition of agents involved two important parameters: autonomous and environment (figure 3 below). The agents are autonomous that they can act according to their will. Agents understand what to do based on the environment. The agent’s action will influence the environment. In most cases, the agents’ actions only have partial control (not complete) control on its environment. The most important concept is that agents will decide on the actions themselves based on their understanding on the environment. These actions are taken without any direct influence from human or any other agents (autonomous).

Figure 2.3: Agents and the Environment



However, we need agents to be more intelligent than just to be autonomous. Agents that are able to possess this intelligence are said to be intelligent agents. Intelligent agents are defined by Wooldridge to possess the following capabilities;

Reactivity: the intelligent agents can perceive the environment, and respond (in a timely fashion) to changes that occur in order to satisfy the design objectives.

Proactive: Intelligent agents are able to exhibit goal-directed behaviour by taking initiative in order to satisfy their design objectives.

Social ability: Intelligent agents are able to interact with other agents to satisfy their design objectives. That means the agent must be able to interact with other agents. In this case, we called that the agents are working in a multi-agent environment.

2.8 Multi-Agent Environment

A multi-agent system has the following characteristics

1. Each agent has incomplete information, or capabilities for solving the problem
2. There is no global system control
3. Data is decentralized
4. Computation is asynchronous.

In order for agents to work in a multi-agent environment, the agents have to overcome two challenges. One challenge is to find the agents to work with, and the other challenge is these agents must be able to interoperate. To overcome the first challenge of finding the agents, this study

recommended the use of middle agents. Each agent then advertises their capabilities to the middle agent, so that other agents can discover. The other challenge of agents in a multi-agent system is how these agents would interoperate and interact with each other. The agents can communicate through a common format for the content of communications, and a shared ontology. The agents sharing the same ontology are able to interpret the communication interactions, mutual understanding and to predict behaviors. One other agent's interaction is the use of mobile agent technology. Mobile agent is the ability of the agents to migrate from one system in a network to another in the same network. This mobility will allow the agent to move to a system that contains the object which the agent wants to interact. The agent mobility can transport the state and code of the agents. A mobile agent system can provide the benefits of reduce network load, overcome network latency, encapsulate protocol, execute asynchronously and autonomously, adapt dynamically, and are more fault tolerant.

2.9 Agent approach for interoperability in Health Insurance

Let us review why IT industry has turned to multi-agents approach for interoperability.

- i. Agents are autonomous and not tightly coupled with other agents or other components; this increased the autonomous and dynamic of the applications.
- ii. Agents do not require the other agents or modules to be using the same technologies. In an agents' approach, the only important thing is the communication message format must be synchronized and common ontology. Web services also working to this direction. However, agent is better than web services in that the agents are intelligent than web services. The agents has notion of fitness in its environment. It is reactive to the environment. If an agent decides that a request for service is not appropriate to response, the agent can decide what to react. This deciding factor can be response time, throughput, capacity, availability, reliability, stability, security etc.
- iii. Agents have concern on semantic interoperability. The enterprises to be integrated can have different information semantic. It is only required the agents to communicate with other agents or modules in other enterprise to share the same semantics. This greatly reduced the semantic interoperability complexities.
- iv. Agents allow the applications to be reactive according to environment. In addition, agents allow proactive actions to be taken in case environment changes. This is not available in any other interoperable technologies.
- v. Agents' modularity allows decomposition of complex systems into subsystems, and thus reduced the complexities in troubleshooting, integration, and testing when integrating with other enterprise.

- vi. Agents provide abstraction in design or troubleshooting. In troubleshooting, we do not have to review the whole systems for problems, only the agents involved. In enhancement of functionalities, only the agents involved will be impacted. This greatly increased the dynamism of the systems.

2.10 Multi-Agent Approach in Health Insurance Business (NHIF)

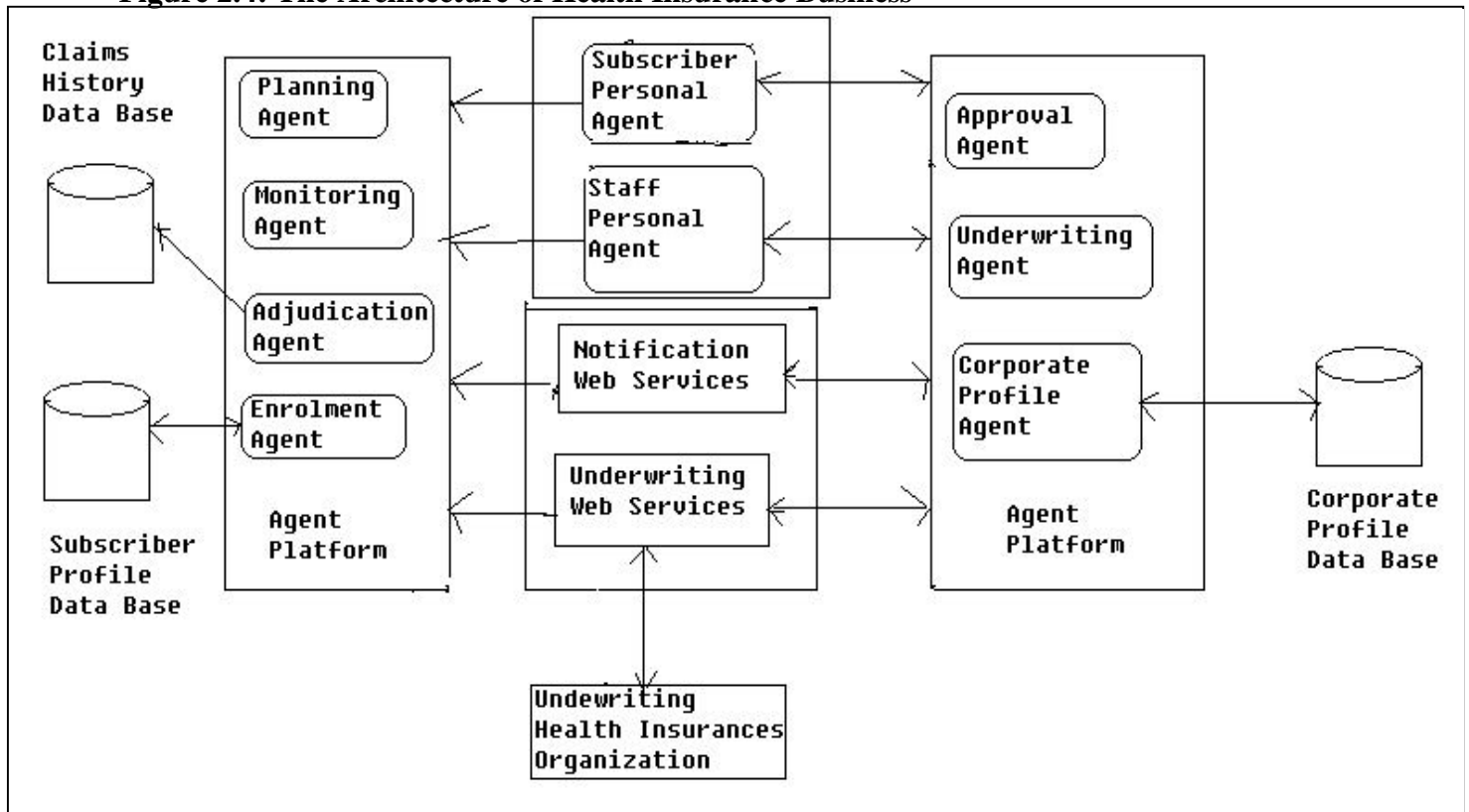
This research project tried to use Health Insurance business as a case study to implement the multi-agent approach to achieve interoperability. A health insurance business at a high level is composed of the following:

- i. Provide enrolment of subscribers into the health insurance policy. This enrollment can be achieved by people enrolling in-person at one of the offices. Also, the subscribers can enroll themselves on the internet.
- ii. Adjudicate the health insurance claims of the subscribers. Subscribers can submit their claim through mail, which would then be keyed in by the staff of the organization. At the same time, the subscribers can submit their claims on line through internet, which would be adjudicated on-line.
- iii. Provide quality insurance planning according to the needs of the subscribers. The subscribers can plan their insurance plans before enrolling in the insurance plans. This planning of plans can be done in-person, or through internet.
- iv. Link up with other health insurance organization for insurance underwriting. It has the following purposes:
 - o Underwrite coverage to other health insurance organization. Not all health insurance organization would insure all plans, underwriting some plan coverage to other organization is a common insurance practice.
- v. Provide pre-approval to health providers before a medical treatment is provided, for example, dental treatment pre-approval.
- vi. Provides system configuration, user interface, and data communication etc to serve all system including the subsystem agents.

2.11 Multi-agent and Service Oriented Architecture of Health Insurance Business

It has nine agents, two web services, and three data bases (Lin, F., Holt, P., Leung, S. and Li, Q. (2006).

Figure 2.4: The Architecture of Health Insurance Business



It has the following agents:

- **Subscriber Personal agent.** The personal agent provides a graphical interface with the subscribers. It also allows the graphical interface to be accessible through internet. This interface is used to introduce the requirements of a search or to show the results of a query to the user. It allows the subscribers to enroll for insurance plans, on-line submit of health claims, request pre-approval, and request for insurance plan advice.
- **Enrolment agent.** This agent will keep a user profile database, allowing enrolling of subscribers. The enrolment has the following major functions:
 - o It will accept/reject subscriber enrolment request from the personal agent or staff personal agent, with the help of the corporate profile agent.
- **Underwriting agent.** The underwriting agent is working with the underwriting web services, and is used to control the underwriting business with other health insurance organizations.
- **Adjudication agent.** The agent is mainly adjudicating the claims of the subscribers. It receives and replies the claims from subscriber personal agent, staff personal agent, and/or underwriting agent. It will retrieve the information of the subscribers through enrolment agent, and decide whether the claim is approved or denied. The results of the adjudication are written to a claim history data base.

- **Planning agent.** It will generate an appropriate health benefit plan to the potential subscribers. This decision is based on subscribers' information received from the personal agent, and communicates with the corporate profile agent to decide on the best insurance plan available for the potential subscribers. It receives and replies the request from subscriber personal agent, staff personal agent, monitoring agent and/or underwriting agent. It retrieves the corporate plan available (through corporate profile agent), and decide on the best plan available for the subscribers.
- **Monitoring agent.** The agent will monitor the enrolment profile database and corporate profile database. When there is a change in subscriber enrolment information, the monitoring agent will work with planning agent to work out a new plan and inform the subscribers through the notification web services. When there is a change in corporate insurance plans, monitor agent will inform enrolment agent to modify the subscriber enrolments impacted, and also work with the planning agent to work out a plan and inform the subscribers through the notification web services. The agent will inform underwriting agent when the subscribers impacted are other health insurance organization.
- **Corporate Profile Agent.** The agent will keep a Corporate Profile database. This database has all the plans available in the organization. It has functions as follows:
 - o Provide corporate profile information to enrolment agent and planning agent.
- **Approval agent.** The main purpose of this agent is providing pre-approval to claims.
 - o It would provide pre-approval as requested by subscriber personal agent, or staff personal agent.
- **Staff personal agent.** It provides user interfaces to the staff of the organization to perform:
 - o Add/change/term of subscribers (local or underwriting), and inform the enrolment agent, underwriting agent

2.12 Agents

An agent (Brenner et al (1998)) is a software or software/hardware that is autonomous (act relatively independently) or software-based computer system that enjoys the following characteristics:

- i. **Autonomy:** acting independently and exercise control over their internal state in an environment.
- ii. **Reactiveness:-** reactive system interacts with its environment; responds to changes that occur in it.
- iii. **Social ability:** Agents have the ability to interact with other agents through Agent Communication Language, the popular ACL being KQML and FIPA.

- iv. **Mobility:** ability to move around electronic network platforms.
- v. **Pro-activeness:** not just being driven by events but recognizing opportunities and talking the initiative.
- vi. **Veracity:** avoid communicating false information knowingly
- vii. **Benevolence:** conflicting goals; agents always try to do what she is asked.
- viii. **Rationality:** agent will act in order to achieve its goals subject to beliefs, and will not act in such a way as to prevent its goals being achieve, at least in so far as its beliefs
- ix. **Learning/adaptability:** agents improve performance over time.
- x. **Personality:** Agent has distinct personality, behavior, name and role

2.13 Agent Architectures and Agent Modeling

This section tries to explain how are agents internally modeled and constructed. **Maes** defines agent architecture as a particular methodology for building agents. It specifies how the agent can be decomposed into the construction of a set of component modules and how these modules should be made to interact. Architecture encompasses techniques and algorithms that support this methodology.

Kaelbling considers an agent architecture to be a specific collection of software (or hardware) modules, typically designated by boxes with arrows indicating the data and control flow among the modules. A more abstract view of architecture is as a general methodology for designing particular modular decompositions for particular tasks.

2.13.1 Types of Agent Architecture

There are three types of agent architecture; these include the following

- i. Symbolic/logical
- ii. Reactive
- iii. Hybrid

Symbolic agents

Agents build with Artificial Intelligence are symbolic agents. They make decisions on what to do using symbolic reasoning. Originally, (1956-1985) all agents built with AI were symbolic reasoning agents. They were supposed to use pure logical reasoning to decide what to do. They were however difficulty to build which led to emergence of reactive agents from 1985 to present.

Some key issues with symbolic agents include:

The transduction problem: that of translating the real world into an accurate, adequate symbolic description, in time for that description to be useful.

The representation/reasoning problem: that of how to symbolically represent information about complex real-world entities and processes, and how to get agents to reason with this information in time for the results to be useful.

Decision making assumes a *static* environment: *calculative* rationality.

Decision making using first-order logic is *undecidable*!

Even where we use *propositional* logic, decision making in the worst case means solving co-NP-complete problems.

Typical solutions to the problems above include

- i. Weaken the logic;
- ii. Use symbolic, non-logical representations;
- iii. Shift the emphasis of reasoning from *run time* to *design time*.

Reactive Agents

The reactive architectures make no reference to their history, and also base their decision making entirely on the present, without any reference to the past. They implement a direct mapping of situation to action and are based on a stimulus– response mechanism triggered by sensor data.

We call such agents purely reactive: action: $E \rightarrow Ac$. The many unsolved problems associated with symbolic AI led to the development of reactive architectures.

Rodney Allen Brooks (criticism of mainstream AI) — behavior languages

Brooks has put forward three theses:

- i. Intelligent behavior can be generated without explicit representations of the kind that symbolic AI proposes.
- ii. Intelligent behavior can be generated without explicit abstract reasoning of the kind that symbolic AI proposes.
- iii. Intelligence is an emergent property of certain complex systems.

Hybrid Agents

Many researchers have argued that neither a completely deliberative nor completely reactive approach is suitable for building agents. They propose *hybrid* systems. *Hybrid* systems attempt

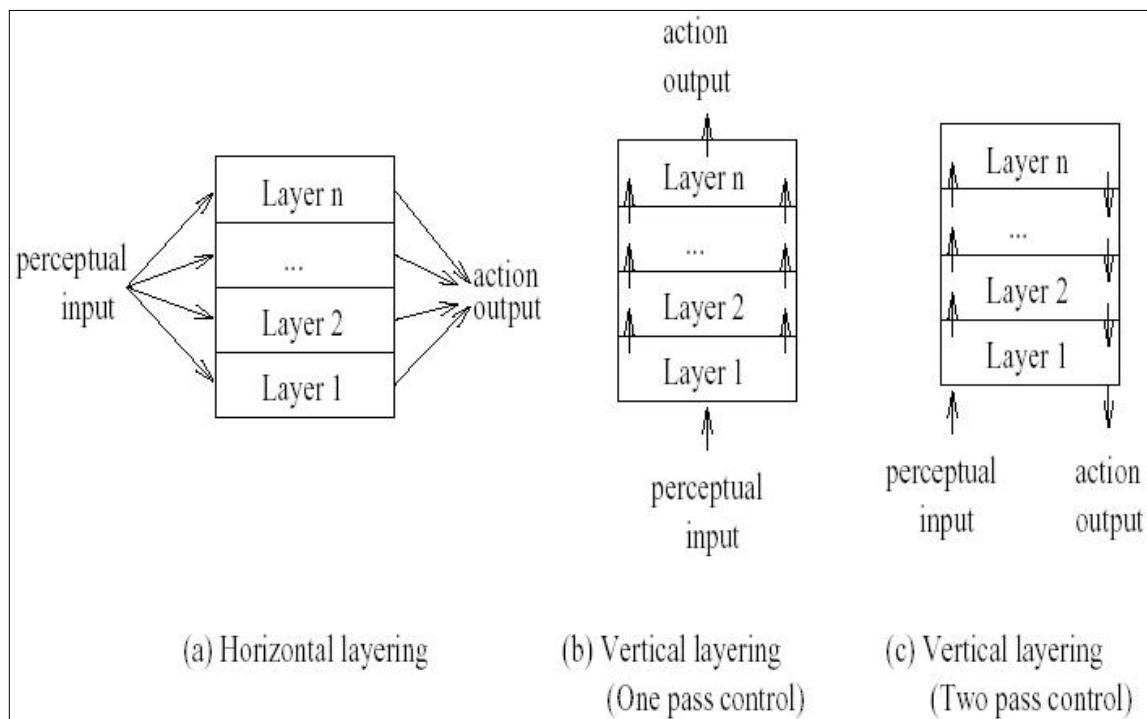
to marry classical and alternative approaches. The approaches may be to build an agent out of two subsystems:-

Deliberative one, containing a symbolic world model, which develops plans and makes decisions in the way proposed by symbolic AI; and Reactive one, which is capable of reacting to events without complex reasoning. Often, the reactive component is given some kind of precedence over the deliberative one. (Michael Wooldridge M, Wiley. J, 2002). This kind of structuring leads naturally to the idea of a layered architecture: With layered architecture, an agent's control subsystems are arranged into a hierarchy, with higher layers dealing with information at increasing levels of abstraction. The layering can be horizontal or vertical as shown in figure 5 below.

Horizontal layering: Layers are each directly connected to the sensory input and action output.

Vertical layering: Sensory input and action output are each dealt with by at most one layer each.

Figure 2.5: Layered architecture



2.14 Multi-Agent based systems technology

This proposal will use agent-based systems technology which is a new paradigm for conceptualizing, designing, and implementing software systems. This technology is an attractive technology for building software systems for environments that are distributed, open and complex. In the recent decades the agent systems were developed and built with one agent. As complexity of these systems increased and agent technology matured incorporation of several

agents has been adopted to counter the limitations of a single agent namely, limitation of computational resources and risk of failure.

Multi-agent system is a system of agents which interact with one another through cooperation, competition, coordination or negotiation [usually to accomplish some goal]. Wooldridge (2002) According to Sycara (1998), Multi-agent system is a system of several agents (implied), and according to Georgini et al.(2001), MAS is an organization of coordinated autonomous agents that interact in order to achieve common goals.

2.14.1 Characteristics of MAS

Agents act on behalf of the user with different goals and motivation. For agents to reliably interact, they require the ability to cooperate, coordinate, and negotiate with each other, much as people do. In comparison to a single agent or a centralized system, MAS has several advantages which include the following:

- i. MAS lead to the realization of increased speed and efficiency.
- ii. Robustness and reliability- no single point of failure; ability to solve problems that are too large for a single agent. Implementation of many agents increases robustness and reliability. If one agent fails, another one can take over.
- iii. Allows for the interconnection and interoperation of multiple existing legacy systems. Legacy systems increasingly fail to interoperate due to differences in syntax and semantics. In order to keep them relevant and useful, MAS technology can be used to incorporate them in to an agent community where they can interact with other systems. This can be achieved by building wrappers, which inject code in a program to allow it to communicate (Genesereth and Ketchpel ,1994).
- iv. Scalability and flexibility- the agents can enter or exit
- v. Reusability and cost – single agent developed, roles assigned; replicable, MAS is also suitable for distributed environment.

2.15 Multi-Agent System (MAS) Methodologies

A methodology is a body of methods employed by a discipline; it is also a method or a procedure for attaining something (Paolo Girogini (2005)). Cases where agent solution is appropriate include the following:-

- i. Open, dynamic, uncertain or complex environment.
- ii. Agents are natural metaphors- such as in organizations with distributed functions, intelligent interfaces.

iii. Data, control or expertise is distributed- such as database systems with different autonomous ownership.

iv. Legacy systems- where interfaces to old systems are important.

A MAS Methodology generally encompasses a set of concepts, notations for modeling aspects of the software and processes that follow in order to build the software. The agent-oriented methodologies have multiple roots; this is shown in the Figure 4 below

2.15.1 Features of Agent Methodology

- i. Should provide sufficient abstractions to fully model and support agents and MASs arguably.
- ii. Should focus on an organized society of agents playing roles within an environment
- iii. Support MAS, where agents interact according to protocols determined by the agents' roles
- iv. Should be agent-oriented in that it is geared towards the creation of agent-based software

2.15.2 Agent Genealogies - (James Odell (2005))

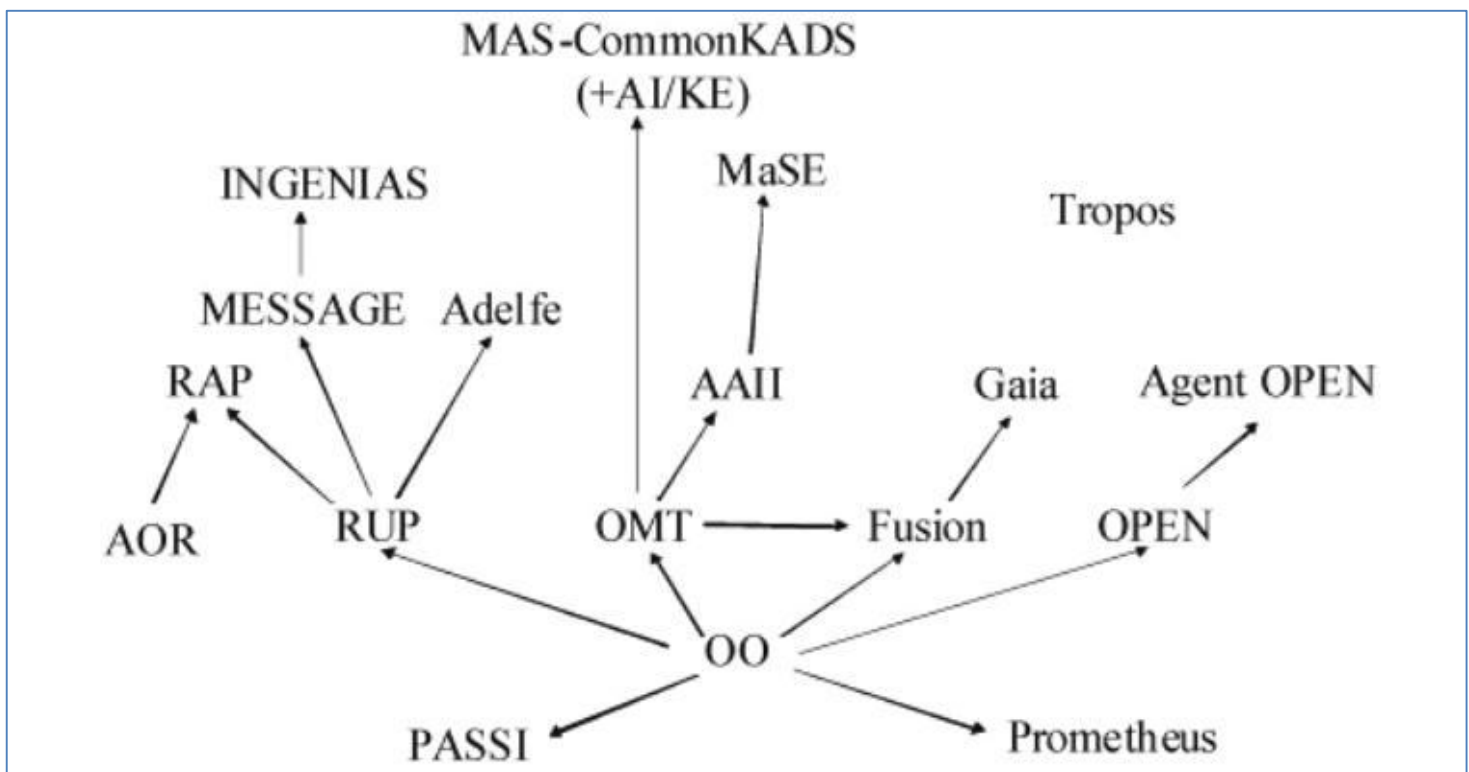


Figure 2.6: Agent Genealogies - (James Odell (2005))

The above figure depicts the influences of Object Oriented Methodologies on Agent Oriented Methods (Henderson- Sellers & Giorgini, 2005)

2.15.3 Agent Oriented analysis model (Burmeister-1996)

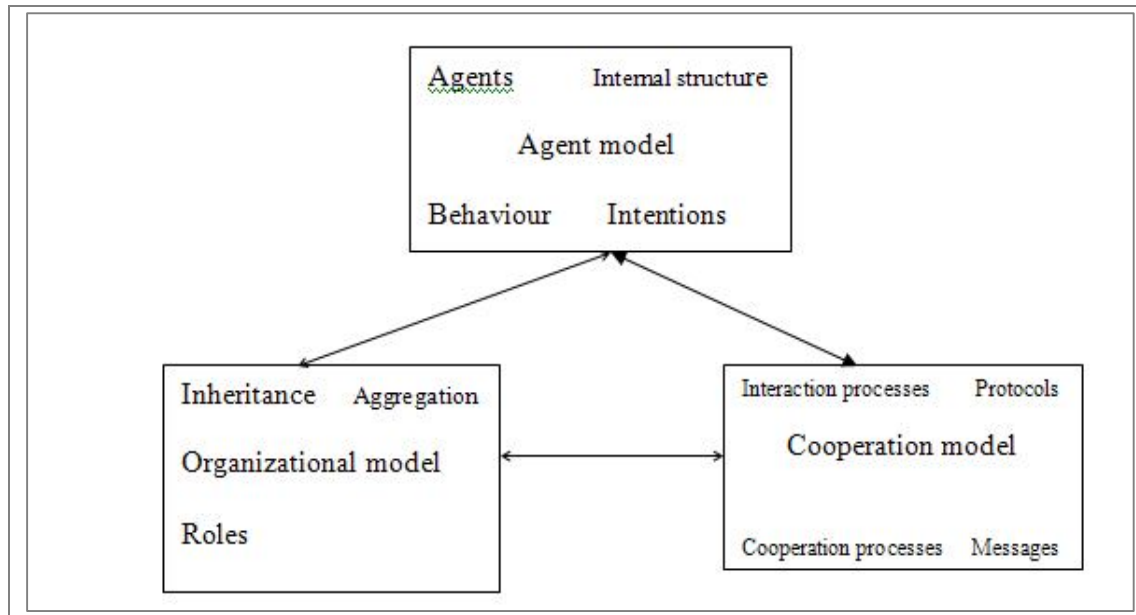


Figure 2.7 shows the agent oriented analysis model

- i. Agent model- this is the actual agents
- ii. Organizational model- this is the static relationships between agents
- iii. Cooperation model- this is the interaction and cooperation between agents

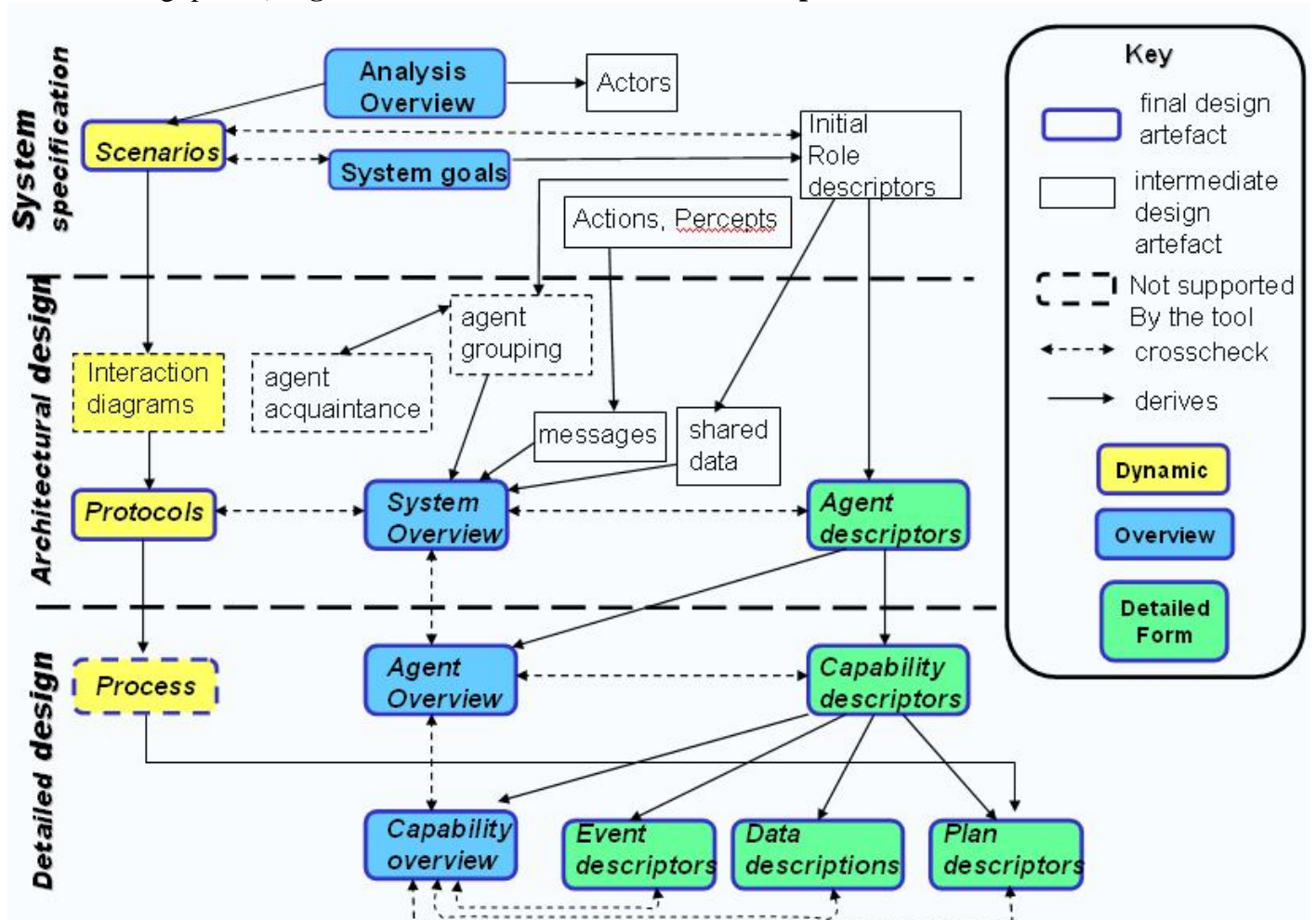
2.16 Prometheus methodology

The study identifies the use of Prometheus methodology to specify, design and implement the system. The Prometheus methodology consists of three phases:

- i. **System Specification:** where the system is specified using goals and use case scenarios; the system's interface to its environment is described in terms of actions, percepts, and external data; and functionalities are defined.
- ii. **Architectural design:** where agent types are identified; the system's overall structure is captured in a system overview diagram; and use case scenarios are developed into interaction protocols.
- iii. **Detailed design:** where the details of each agent's internals are developed and defined in terms of capabilities, data, events, and plans; process diagrams are used as a stepping stone between interaction protocols and plans.

2.16.1 Stages of Prometheus Methodology

The figure 2.8 below shows the various stages or phases of Prometheus Development. (From: Lin Padgham, Michael Winikoff (2005). Prometheus: A Practical Agent-Oriented Methodology, in Brian Henderson-Sellers, Paolo Grogini (2005). *Agent oriented methodologies*. Idea Group Publishing, p. 126) **Figure 2.8: Phases of Prometheus Development**

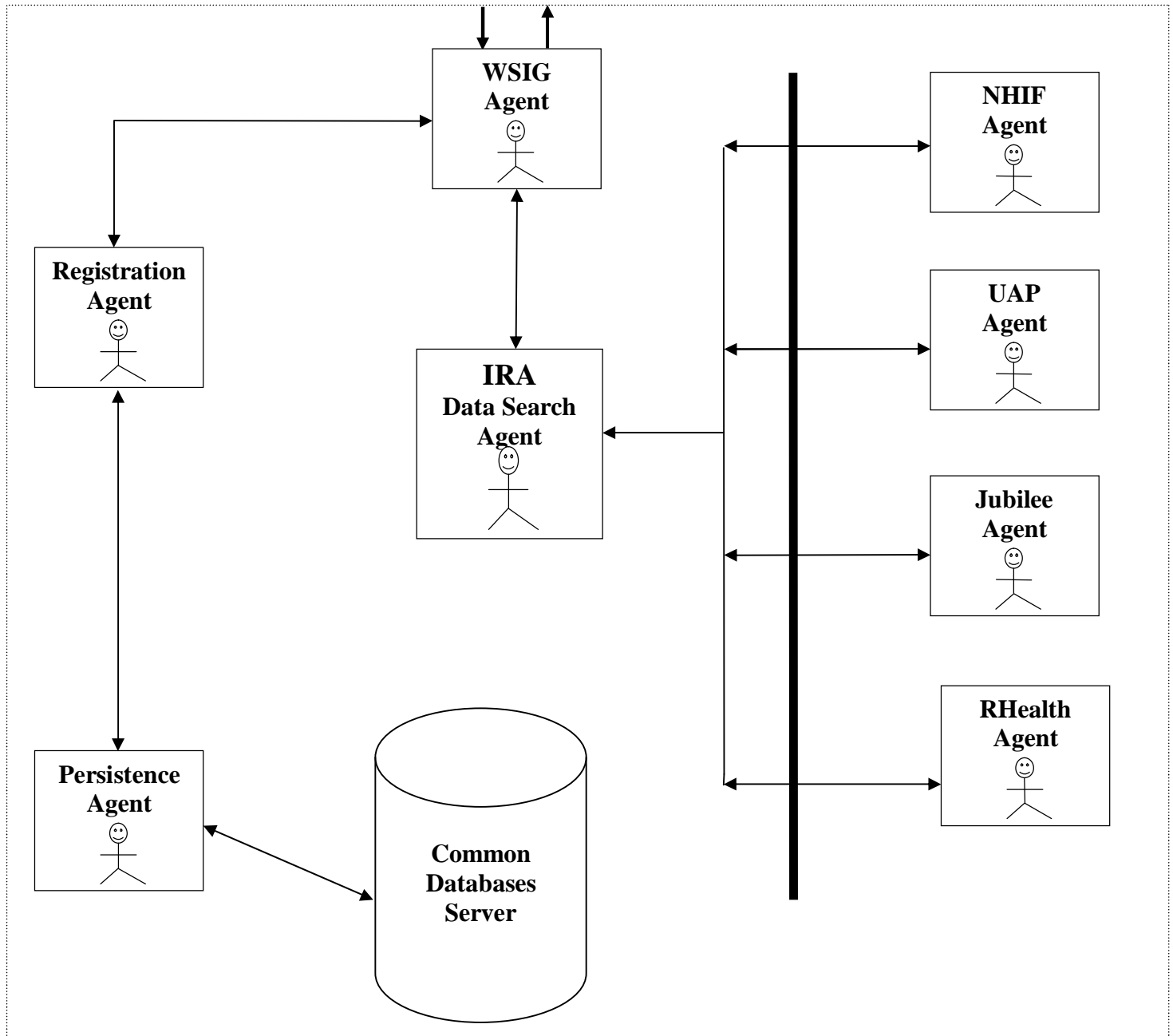


2.17 Conceptual framework

The proposed multi-agent based system will provide data in a timely manner to the users. It will also update the central database of the system. The framework will have the following five main agents.

- i. WSIG agent –web services Integrations Gateway
- ii. Registration agent
- iii. Data Search Agent
- iv. External System Agents
- v. Persistence Agent

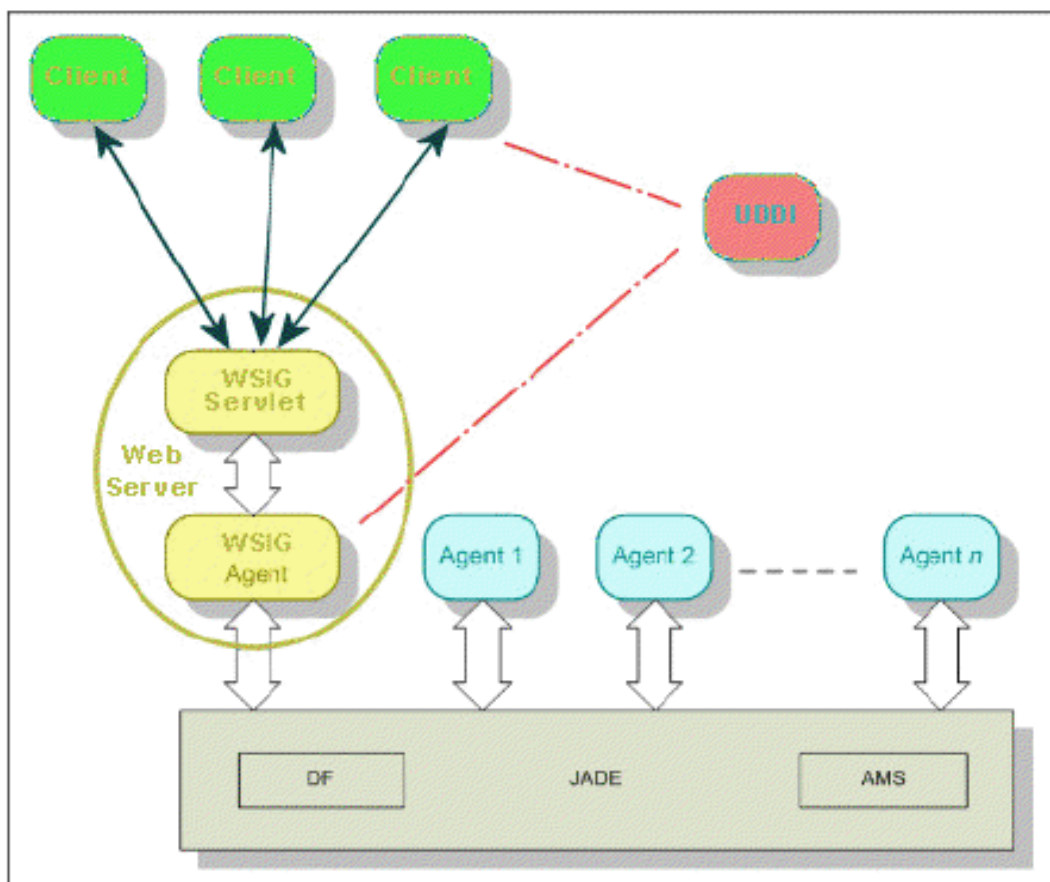
Figure 2.9: The Conceptual Framework



WSIG agent

This is the web services Integration Gateway add-on that provides support for invocation of JADE agent services from web service clients. It is the link between the agents and the web world which handles registration of external agents as web services in this platform; the WSIG agent receives web requests, converts them into the appropriate object specified in the ontology and forwards these requests to the appropriate agents.

This agent converts results of agent processing into a soap response that is converted into a HTTP response and sent back to the client.



This **figure 2.10** above shows the WSIG Architecture, the WSIG add-on supports the standard web services stack, consisting of WSDL (Web Service Definition Language) for service descriptions, SOAP (Simple Object Access Protocol) message transport and a UDDI(Universal Description, Discovery and Integration) repository for publishing web services using tModels (technical Model). This diagram depicts that WSIG is a web application composed of two main elements:

i. **WSIG Servlet**

WSIG Servlet is the front-end towards the internet world and is responsible for serving incoming HTTP/SOAP requests, extracting the SOAP message and also preparing the corresponding agent action and passing it to the WSIG Agent. Moreover once the action has been served

- Converting the action result into a SOAP message
- Preparing the HTTP/SOAP response to be sent back to the client

ii. **WSIG Agent**

The WSIG Agent is the gateway between the web and the Agent worlds and is responsible for forwarding agent action received from the WSIG Servlet to the agents actually able to serve them and getting back responses, the WSIG Agent is also responsible for subscribing to the

JADE DF to receive notifications about agent registrations/deregistration, is also responsible for creating the WSDL corresponding to each agent service registered with DF and publish the service in a UDDI registry if needed.

The two main processes are continuously active in the WSIG web application

- i. The process responsible for intercepting DF registrations/deregistrations and converting them into suitable WSDLs. This process is completely carried out by the WSIG Agent.
- ii. The process responsible for serving incoming web service requests and triggering the corresponding Agent actions. This process is carried out jointly by the WSIG Servlet (performing the necessary translations) and the WSIG Agent (forwarding requests to agents able to serve them)

Platform Monitor Agent

This agent initializes the platform and agents. It creates a link to the platform database and it ensures that all the required parts of the platform are running, these may include database and all the required agents.

Registration Agent

This agent handles registration to and deregistration from the platform by external system. It also validates the registration requests, approves or denies registration. It also validates existing registration details.

Data Search Agent

This agent receives client requests for data from the WSIG agent. It instructs the appropriate external system agents to search for the required data and return the results. This agent also compiles the results from the different external systems and sends them back to the client.

External System Agents

These agents provide access to the external systems registered onto the platform. There are as many of them as external systems registered on the platform. They perform search for requested data on their respective platforms under instruction from the Data Access Agent. They wrap data from different sources into one standard format understood by all external systems.

RMA, DF and AMS agents

These are system agents, shipped with the JADE distribution; HostMonitor can monitor remote networks using Remote Monitoring Agents (RMA). RMA is small application that accepts

requests from HostMonitor, performs test (or action) and provides information about test result back to HostMonitor.

JADE implements a Directory Facilitator (DF) agent as specified by FIPA. The DF is often compared to the "Yellow Pages" phone book. Agents wishing to advertize their services register with the DF. Visiting agents can then ask (search) the DF looking for agents which provide the services they desire. Jade creates multiple containers for agents, each of which can be on the same computing system or different systems. Together, a set of containers forms a *platform*.

Each platform must have a **Main Container** which holds two specialized agents called the AMS agent and the DF agent. The **AMS** (Agent Management System) agent is the authority in the platform. It is the only agent that can create and kill other agents, kill containers, and shut down the platform. The **DF** (Directory Facilitator) agent implements a yellow pages service which advertises the services of agents in the platform so other agents requiring those services can find them.

Persistence Agent

This handles database operations for the platform.

Goal

The services web, also known as WEB SERVICES, are becoming most important in the panorama of software development and a sort of de facto standard for interconnecting different applications. The objective of WSIG is to expose services provided by agents and published in the JADE DF as web services with no or minimal additional effort, allowing for the enough flexibility to meet specific requirements then may have.

The process involves the generation of a suitable WSDL for each service-description registered with the DF and possibly the publication of the exposed services in a UDDI registry.

Requirements

The WSIG add-on requires Java JRE v5.0 (<http://java.sun.com/javase/>), JADE v3.5 or later and a Servlet container such as Jakarta Tomcat (<http://jakarta.apache.org/tomcat/>). Furthermore, if the automatic UDDI publication feature is turned on, a suitable UDDI registry must be available.

The WSIG makes use of the following third party libraries **already included in the WSIG distribution**:

Apache Axis v1.4 (<http://ws.apache.org/axis/>)

Apache Commons (<http://jakarta.apache.org/commons/>)

UDDI4J v2.0.5 (<http://uddi4j.sourceforge.net/>)

WSDL4J v1.6.2 (<http://sourceforge.net/projects/wsdl4j>)

Eclipse EMF v2.3.0 (<http://www.eclipse.org/emf/>)

2.18 Exposing Agent Services as Web Services in this Platform

JADE agents publish their services in the DF (Directory Facilitator) providing a structure called DF-Agent-Description and defined by the FIPA specification. A DF-Agent-Description includes one or more Service-Description each one actually describing a service provided by the registering agent. A Service-Description typically specifies, among others, one or more ontologies that must be known in order to access the published service. The actions the registering agent is actually able to perform are those defined in the specified ontologies.

In order to expose an agent service as a web service it is sufficient to set the `Wsig` property to `true` in the properties of the Service-Description at DF registration time as below:-

Table 4.1

```
.....  
ServiceDescription sd = new ServiceDescription();  
.....  
sd.addProperties(new Property("wsig", "true"));  
.....
```

2.19 Agent-based Framework for Integration in Multi-agent Systems

Multi-agent systems are often created from scratch by researchers and developers. There are however various frameworks that are available that can be used to implement common standards such as Zeus and FIPA (jade.tilab.com) . A Framework for agent systems saves developers time and also helps in standardization of MaS development. Actually, advances in agent technology depend on improving the frameworks. In this research study, the FIPA framework was used to implement the multi-agent based system for sharing data in health insurance firms. The FIPA standard specifies the agent abstract architecture, agent management system, agent message transportation and agent communication

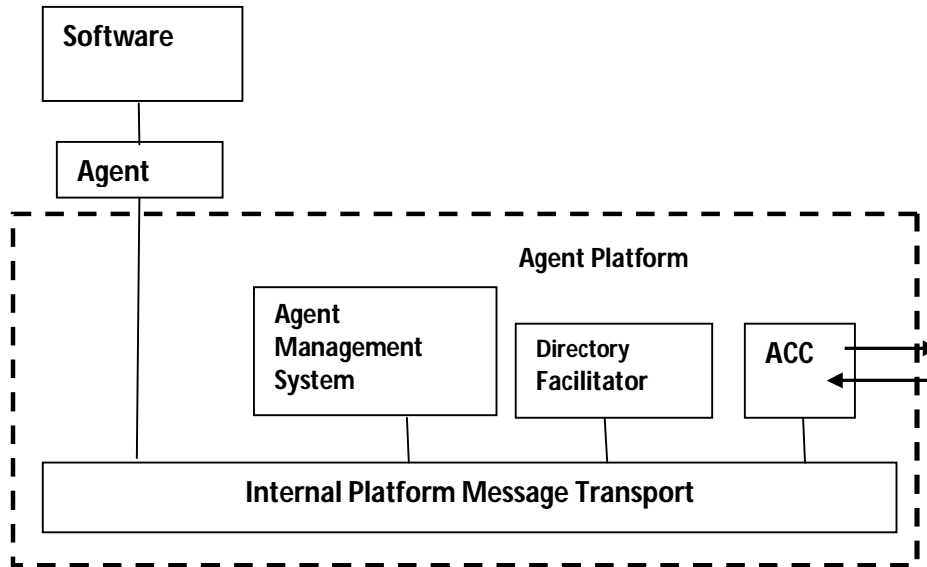


Figure 2.11: – FIPA reference model of an Agent Platform

JADE

JADE is a Java agent framework to develop agent applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems. FIPA standards ensure that systems have a high degree of interoperability. They do this by ensuring that only the external behavior of system components are specified, leaving implementation details and internal architectures to agent developers. The JADE FIPA-compliant Agent Platform includes:-

1. AMS (Agent Management System). This is the agent that exerts supervisory control over access to and use of the platform; it is responsible for authentication of resident agents and control of registrations.
2. The DF (Directory Facilitator), the agent that provides a yellow page service to the agent platform.
3. The Message Transport System, also called Agent Communication Channel (ACC), is the software component controlling all the exchange of messages within the platform, including messages to/from remote platforms. It must support interoperability between different agent platforms.

NOTE: The above three components are automatically activated at the Agent platform start-up.

2.20 The FIPA Platform

i. Agent Management System (AMS)

- The AMS provides white-page and life-cycle service, maintaining a directory of agent identifiers (AID) and agent state.
- Each agent must register with an AMS in order to get a valid AID.

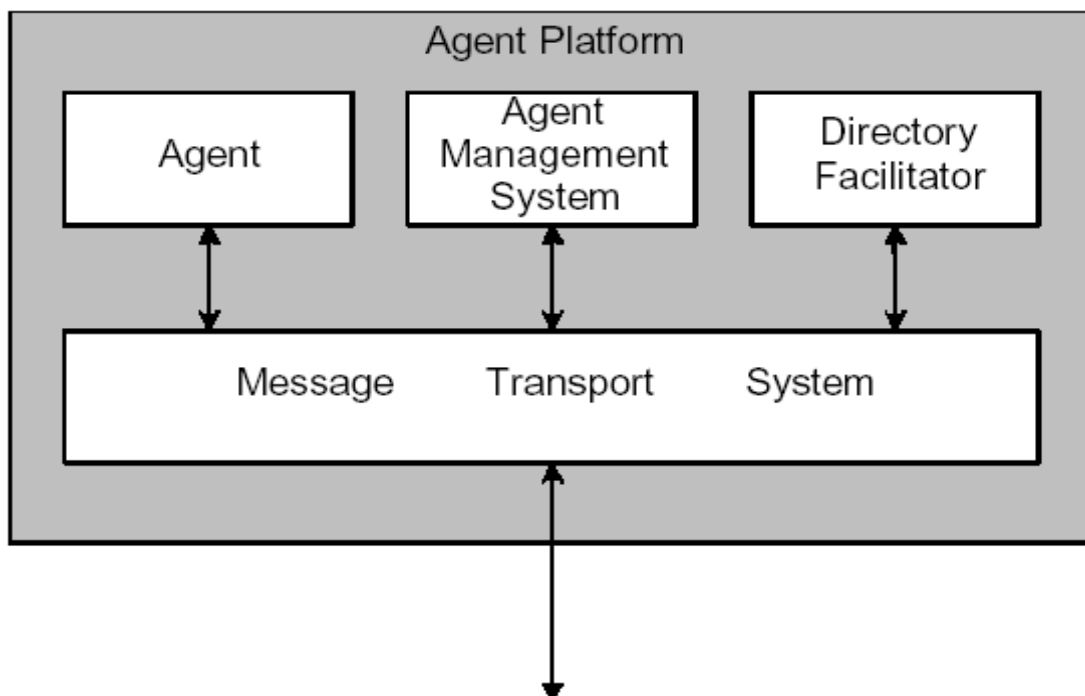
ii. Directory Facilitator (DF)

- the agent who provides the default yellow page service in the platform.

iii. Message Transport System (MTS)

- the software component controlling all the exchange of messages within the platform, including messages to/from remote platforms.
- also called Agent Communication Channel (ACC)

Figure 2.12 The FIPA Agent Platform



3.0 CHAPTER THREE: METHODOLOGY

3.1 Introduction

In order to realize the success of the proposed study, the following steps were taken into consideration

- i. Requirements gathering
- ii. Analysis and design of the data sharing platform for IRA
- iii. Implementation and results
- iv. Testing and evaluation of the platform
- v. Documentation and submission of the final report

3.2 Requirement gathering

This has been done by studying the existing manuals together with health insurance documentation. Some information has been obtained using interviews and questionnaires. The interviews were conducted with some stakeholders. Where interviews were not possible, FAQs and observation were applied.

3.3 Data collection process

The objective of data collection was to obtain evaluation data. Primary and secondary data was also collected. To choose a representative sample, a systematic sampling technique was used.

3.4 Data collection Tools

Primary data was collected using questionnaires. A list of questions with choices and options were prepared and distributed to the respondents by email and others by hand. Secondary data was obtained from reports and statistical publications by Kenya Government Ministries.

3.5 Analysis and design of the data sharing platform

Analysis and design of this platform was done using multi-agent system (MAS) methodology. In this case PROMETHEUS was specifically used. The details of analysis and design of this platform are covered in next chapter. Data once collected was first processed to detect errors, omissions and to classify it. Analysis was carried out to draw comparisons and validate conclusions. The tools for analysis of primary data from questionnaires included Microsoft Excel worksheet and statistical software (SPSS). Secondary data was also analyzed with Microsoft Excel worksheet.

3.6 Coding and debugging

Coding is the process of translating the system specifications into program code statement that are the compiled and executed to produce results or outputs to the user. A written program can refuse to compile or can compile and refuse to execute. A program code can compile and execute successfully, but fail to produce correct results. The error that prevents the program from compiling, executing and producing the correct results is known as a bug. For the user requirements to be met or achieved, any detected bug must be removed, this process of detecting the bug is called testing but debugging is the process of removing the detected bug to enable proper compilation, execution and generation of correct or desired outputs.

In order to implement this platform, the Java Agent Development (JADE) framework was used. JADE has been used in Java development kit (JDK) version 6.1 environment

3.7 Testing

Testing is the process of detecting errors in a software program. The detected errors must be corrected to ensure that the software meets its specifications. Due to the sensitivity of health insurance data, it was not possible to use live data for testing purposes. The simulated data was therefore used to test the operation of the developed system. The details of this system testing and evaluation are covered properly in chapter five “5.2 System Testing and Evaluation’.

3.8 Evaluation of the multi-agent based system

Test results will determine the behavior of the system when subjected to load. The results shall be analyzed and interpreted to establish the correct operation of the system. Based on the interpretation of the results, further study is going to be recommended on the proposed system. The evaluation of the multi-agent based system was done with the users in various health insurance firms so as to get comparative views of the existing system and the multi-agent based system. Evaluation shall involve data collection and data analysis.

3.9 Documentation (Report Writing)

This step started from the beginning of the project up to the end of the project. All pieces of documentation have been refined to come up with a report for submission to the panel of examiners. The MS office 2007 was used for documentation and report writing.

3.10 Methodology

A brief literature review of the health insurance system was carried out with a view to gain thorough understanding of the underlying needs, challenges and desired operation level of health

insurance interoperability systems. The review guided the process of compiling the requirements document. A study on multi-agent systems technology was also conducted to justify its appropriateness in providing a solution to problems in health insurance. The study identified the use of Prometheus methodology to specify, design and implement the system.

In order to achieve the overall objectives set out in this research project, a number of activities were carried out as outlined below:

- i. Review of Related work on health insurance systems and Multi-agent technology
- ii. System specification
- iii. System design
- iv. System implementation
- v. Testing
- vi. Evaluation

3.11 The Prometheus methodology

Prometheus methodology was used in the system specification, system design and implementation. The Prometheus methodology is a detailed process for specifying, designing, and implementing intelligent agent systems. This methodology was used in this study because it distinguishes itself from other methodologies by supporting the development of intelligent agents, providing start-to-end support, having evolved out of practical industrial and pedagogical experience, having been used in both industry and academia, and, above all, in being detailed and complete. Prometheus is also amenable to tool support and provides scope for cross checking between designs.

It is preferred over other MaS methodologies in this study due to the fact that it is a complete methodology that is, from start to end. It has detailed and elaborate process for system specification, implementation, testing and debugging. It is also supported by Prometheus Design Tool that allows the user to design overview diagrams. Further, it has examples which help to better understand what is required in each stage of development. It has three phases which were implemented to accomplish activities set out in section 3.0

The methodology consists of three phases: system specification, architectural design, and detailed design. An overview of the methodology, including its phases, deliverables, and intermediate products, is depicted below. Although the phases are described in a sequential fashion it is acknowledged that like most Software Engineering methodologies, practice involves revisiting earlier phases as one works out the details. The following include the description of the phases and deliverables.

3.11.1 System specification

This is the first step in Prometheus methodology. The following activities were undertaken.

- a) Identification of system goals and sub-goals because proactiveness was desired. What was the system being built for? What were the subsidiary goals?
- b) Developed use case scenarios. Scenarios show a particular instance of a system. There was no branching. Scenarios consist of a sequence of steps which were themselves scenarios.
- c) Identified the agent system's interface to the environment in terms of actions, percepts, and external data.
- d) Identified the agent functionalities.
- e) Identified data read and written by functionalities
- f) Prepared functionality schemas (name, description, actions, percepts, data used/produced, interaction (with other functionalities), and goals)

3.11.2 Architectural System design

In this phase, the system specification artefacts were used to build the system architecture. The following steps were followed:

- a) Specification of agent types.
- b) Defining agent types and developing agent descriptors
- c) Produce a system level overview diagram describing the overall structure of the system
- d) Develop interaction protocols from use case scenarios (via interaction diagrams)

3.11.3 Detailed design

This is the phase where the details of each agent's internals were developed and defined in terms of capabilities, data, events, and plans. The process diagrams were also used as a stepping stone between interaction protocols and plans.

3.12 System implementation

During the Detailed Design phase agent overview diagram was produced and the agent capabilities and interactions were also produced.

4.0 CHAPTER 4: ANALYSIS AND DESIGN

4.1 Introduction

This system has been developed for IRA for regulation, supervision, consumer education and consumer protection among the health insurance firms in Kenya. It has been developed using multi-agent system methodology. The IRA can provide various services requiring health insurance; this system will allow different system to be integrated from various health insurance firms without changing their current systems.

4.1.1 User requirements

User requirements specify what the user wants to achieve from the proposed system. These are the problems which the proposed system should solve. The following are the user requirements

- i. Provision of secure access to the system
- ii. Interaction with the System Administrator and users using the GUI interface
- iii. Accessing number of policy holders in each and every health insurance firm
- iv. Regular premiums payment
- v. Policy holders' details e.g. dependants
- vi. Public Consumer education dates
- vii. Treatment given to policy holders
- viii. Condition / Diagnosis
- ix. Statement of accounts
- x. Date of Admission
- xi. Date of Discharge
- xii. Insurance Structure
- xiii. Analysis of application and registration forms
- xiv. Turnover –submission of data, Monthly, Quarterly and Annually
- xv. Application –state how to apply for the health insurance products

4.1.2 System requirements

System requirements (specifications) include the description of

- i. The inputs to the process output
- ii. The operations the system performs for each input
- iii. The output obtained for the corresponding input

Agent-based Interoperability System in Health Insurance has been developed using the following agent programs

- i. External Systems Agents (UAP, NHIF, IRA, Jubilee Agents)
- ii. Persistence agent
- iii. Registration Agent
- iv. Data Access Agent
- v. Platform Monitor Agent
- vi. RMA, DF and AMS Agents
- vii. WSIG Agent**

4.1.2i Registration Agent

This agent handles registration to and deregistration from the platform by external system. It also validates the registration requests, approves or denies registration. It also validates existing registration details.

4.1.2ii IRA Data Access Agent

This agent receives client requests for data from the WSIG agent. It instructs the appropriate external system agents to search for the required data and return the results. This agent also compiles the results from the different external systems and sends them back to the client.

4.1.2iii External System Agents

These agents provide access to the external systems registered onto the platform. There are as many of them as external systems registered on the platform. They perform search for requested data on their respective platforms under instruction from the Data Access Agent. They wrap data from different sources into one standard format understood by all external systems.

4.1.2iv RMA, DF and AMS agents

These are system agents, shipped with the JADE distribution; HostMonitor can monitor remote networks using Remote Monitoring Agents (RMA). RMA is small application that accepts requests from HostMonitor, performs test (or action) and provides information about test result back to HostMonitor.

4.1.2v Persistence Agent

This handles database operations for the platform.

4.1.2vi WSIG Agent

The WSIG Agent is the gateway between the web and the Agent worlds and is responsible for forwarding agent action received from the WSIG Servlet to the agents actually able to serve them and getting back responses, the WSIG Agent is also responsible for subscribing to the

JADE DF to receive notifications about agent registrations/deregistration, is also responsible for creating the WSDL corresponding to each agent service registered with DF and publish the service in a UDDI registry if needed.

4.1.2vii Platform Monitor Agent

This agent initializes the platform and agents. It creates a link to the platform database and it ensures that all the required parts of the platform are running, these may include database and all the required agents.

4.1.3 Analysis using PROMETHEUS concepts

The analysis of this system has been done using multi-agent system development known as PROMETHEUS which is used for designing intelligent software agents and agent systems. The methodology provides detailed guidance in terms of processes as well as notations. The analysis models for the system are represented using the concept and symbols for PROMETHEUS methodology. The following is a list of various agents which constitute the system, the service/tasks to be performed, goals to be achieved and the collaborators.

4.1.3i External System Agents

Tasks/services

- Register services to IRA
- provide access to the external systems registered onto the platform
- perform search for requested data on their respective platforms under instruction from the IRA

Data Access Agent

Main Goal

- Wrap data from different sources into one standard format understood by all external systems.

Collaborators

1. Data Access Agents
2. Registration Agents

4.1.3ii Registration Agent

Tasks/Services

- handles registration to and deregistration from the platform by external system.
- validates the registration requests, approves or denies registration.

Main Goal

-validates existing registration details.

4.1.3iv WSIG Agent

Tasks/Services

- gateway between the web and the Agent worlds
- responsible for forwarding agent action received from the WSIG Servlet to the agents actually able to serve them and getting back responses,
- responsible for subscribing to the JADE DF to receive notifications about agent registrations/deregistration,

Main Goal

Have two main goals

- creates the WSDL corresponding to each agent service registered with DF
- Publishes the service in a UDDI registry if needed.

Collaborators

- IRA Data Access Agents
- Registration Agents
- External System Agents

4.1.3v Platform Monitor Agent

Tasks/Services

- Initializes the platform and agents.
- Creates a link to the platform database

Goal

- Ensures that all the required parts of the platform are running, these may include database and all the required agents

Collaborators

- WSIG Agent
- Data Access Agent
- Registration Agent
- Persistence Agent

4.1.3vi Persistence Agent

Task/Service and Goal

- handles database operations for the platform

4.1.4 User interfaces

Each agent in the system should know the identity of other agents to interact with in pursuit of its goals. For this agent-based interoperability system in health insurance, the identities of other agents must be configured during system installation or modification. Configuration of the agent identities will ensure that the right information is delivered and received by the right agent. For example, if a new external system agent joins the existing service, the existing agents will have to include its identity to their databases, at the same time, the new external system agent will have to configure the identities of the existing external system agent.

All these configurations and modifications are done by the systems Administrator. In order to facilitate system configuration and modification, each agent program has at least a configuration interface. This interface is either command-based or graphical user interface.

Apart from system configuration and modification done by system administrators, other system users would also wish to obtain some information. The interaction between the system users and the system can also be provided by using command-based or GUI. For ease of user interaction, user interfaces are provided using GUIs. The feedback from the system due to user's query shall also be in the form of GUIs.

4.1.5 Analysis Model and View

Analysis was carried out to produce system specification. The specification was a model which consisted of a collection of views of the system to be developed and its environment. It facilitates communication among the people involved in the system development. The Agent-based Interoperability System in Health Insurance has been analyzed accordingly to the following modal views defined in the PROMETHEUS using and developing subsets of entity and relationship concepts.

Prometheus defines a proper detailed process to specify, implement and test/debug agent-oriented software systems. It offers a set of detailed guidelines that includes examples and heuristics, which help better understanding what is required in each step of the development. This process incorporates three phases.

The system specification phase identifies the basic goals and functionalities of the system, develops the use case scenarios that illustrate the functioning of the Prometheus system, and specifies which are the inputs (percepts) and outputs (actions). It obtains the scenarios diagram, goal overview diagram, and system roles diagram.

The architectural design phase uses the outputs produced in the previous phase to determine the agent types that exist in the system and how they interact. It obtains the data coupling diagram, agent-role diagram, agent acquaintance diagram, and system overview diagram.

The detailed design phase centers on developing the internal structure of each agent and how each agent will perform his tasks within the global system. It obtains agent overview and capability overview diagrams. Finally, Prometheus details how the entities obtained during the design are transformed into the concepts used in a specific agent-oriented programming language (JACK); this supposes, in principle, a loss of generality.

4.2 System Specification

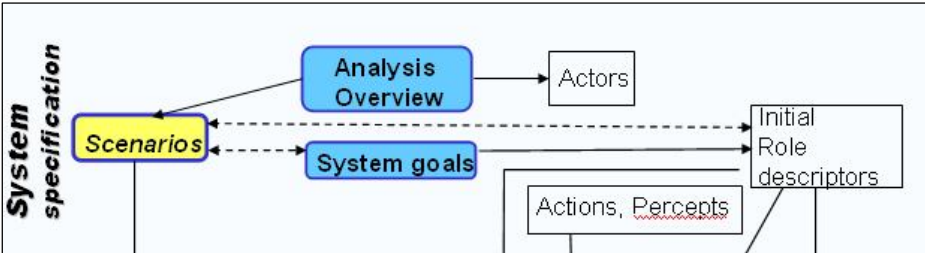


Figure 4.1 System Specification diagram. This is the initial phase of Prometheus methodology for building MAS.

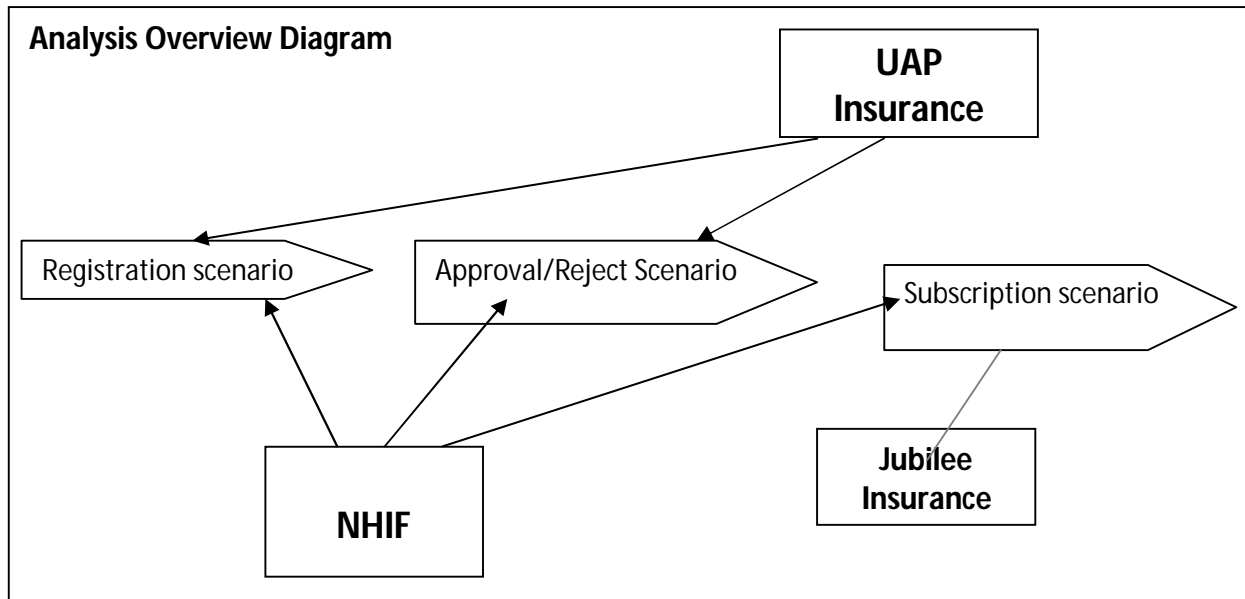
The phase involved specifying the system requirements using goals and scenarios and identifying the actions and percepts. Use of case scenarios were developed together with the agents interface to the environment.

During the research study this phase was first phase of Prometheus methodology which was considered as the pre-design stage, the diagram shows the overview diagram of System specification and its design artifacts. It is not unusual for the initial ideas for a system to be captured very briefly, possibly in a few paragraphs. During System Specification this description must be elaborated and explored, to provide a sound basis for system design and development. In this research study, an Agent-based Interoperability system in Health Insurance was described as a system with four distinct agents who would make the platform to functionally operate with different systems.

Typically, using Prometheus, the development of the System Specification began with identifying the external entities which were referred to as *actors* that used or interacted in some way with the system, and the key *scenarios* around which interaction occurred.

In the figure below the following UAP Insurance, NHIF, Jubilee Insurance among others (referred to as External system Agents) were identified as the entities that would interact with the system. They were associated with some main scenarios which corresponded to the main functionality of the system.

Figure 4.2: Analysis Overview Diagram



After coming up with the above diagram, percepts as input to each scenario were identified and the actions produced by the system for each scenario linking them to the appropriate actors were also identified. For example, UAP insurance registers its services to IRA Data Access System as a percept (input) to the system and the system performs an action of sending an acknowledgement back to UAP insurance. The analysis overview diagram thus defines the *interface* to the system in terms of the percepts (inputs) and actions (outputs).

The next step was to specify the details of the scenarios that we identified in the analysis overview diagram. A scenario is a sequence of *structured steps* where each step can be one of: *goal, action, percept, or (sub) scenario*. Each step also allows the designer to indicate the *roles* associated with that step, the *data* accessed, and a *description* of the step. These preliminary goals, roles and data that are identified are used to automatically propagate information into other aspects of the design. As steps are defined, the relevant entities are created if they do not yet exist.

4.2.1 System description:

Numbers of policy holders in health insurance, regulation of monthly premiums are required by IRA for health planning, and population statistics. The goals of the system and use cases were used to analyze the requirements of the system.

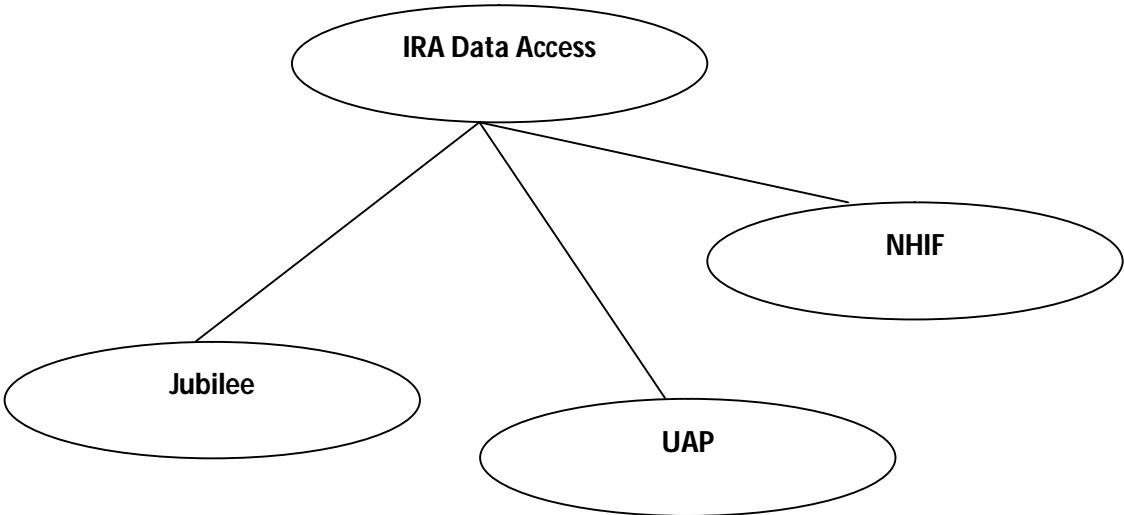
4.2.2 GOALS

What is the system built for?

Overall Goal

The overall goal of the system is to regulate all these health insurance firms to ensure insurance/reinsurance companies and intermediaries remain operationally viable and solvent; and establish a transparent basis for timely, appropriate and consistent supervisory intervention, including enforcement, in a timely and accurate way and make the information available to stakeholders and interested individuals.

Figure 4.3: Goal Diagram



4.2.3 Use Case Scenarios

A use case is a view of the functionality (or use) of a system from the user’s perspective. Use case scenarios represent a particular instance of the system without branching. Use case analysis has been used for requirements specification of the multi-agent system and the behaviour specification.

(Michael Papasimeon and Clinton Heinze, 2000)

Use case1: The health insurance firm logs on to the system and proceeds to register her services with this data sharing platform. The service can either be subscription details or requesting for approval of previous application. If a new Health insurance firm, they must first register their services with the platform before getting permission to receive the services.

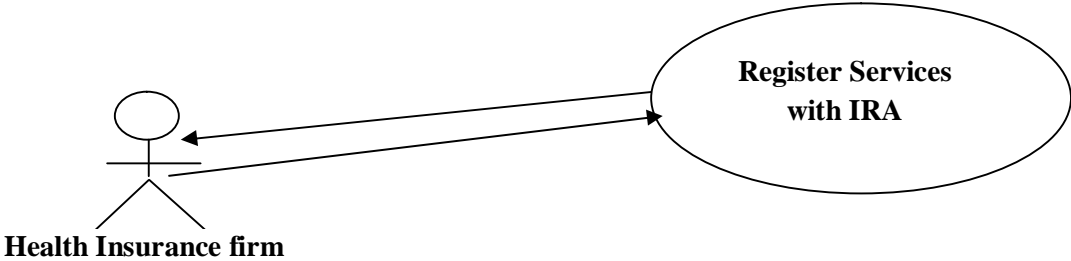


Figure 4.4: Use case 1

Use case2: Subscription: the entities who wish to get updates on registration of new health insurance firm must subscribe into the system filling the online form. The subscribers get acknowledgement whenever a new health insurance firm is registered in the system. Subscribers can turn off alerts by unsubscribing.

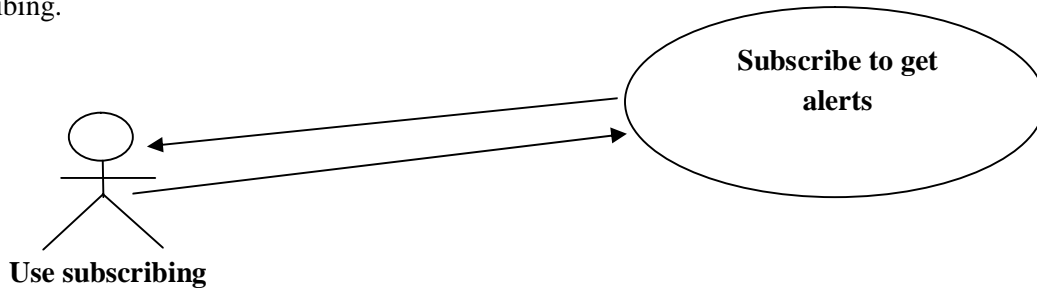


Figure 4.5: Use case 2

Use case 3: Search for existing registered health insurance firm. A user who is subscribed searches the system for a particular record using specified parameters.

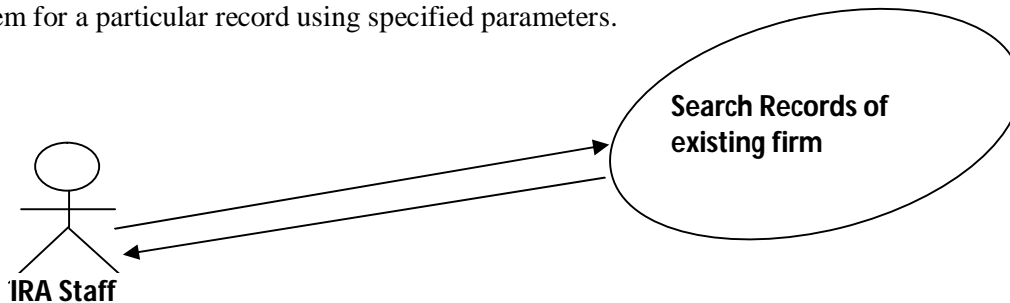


Figure 4.6: Use Case 3

4.2.4 Identification of the Systems Interface to the Environment

The systems interface to the environment was defined in terms of actors, precepts and external data.

4.2.5 Actors

The main actors in the IRA health insurance system are :

i. Health Insurance Companies

A health insurance firm approaches the IRA with information about a new registration occurrence. The IRA personnel records the information in the system. The information captured includes names, Registration Number, date of registration, Amount payable, Statement of Accounts and probable reason for registration.

ii. IRA

IRA as the main user of the system, give information regarding the reasons why these firms should register their services with the platform. The information captured includes Number of

policy holders, Premium payable, Dependants of policy holders, date of Admission and date of discharge of the policy holder in case they launch claims, hospital where admitted.

iii. Subscribers

The subscribers to the system get alerts whenever new services are registered. The subscribers get the name, date of registration. They also make queries to the system about existing firms.

iv. Percepts and Actions

The multi-agent based system for Registration of health insurance firms precepts and events include the IRA Personnel in Technical department receiving a new application and entering it into the agent system, users searching the system for various information, users subscribing / unsubscribing from the system. Actions include sending alerts to subscribed users, sending short message, alerts to the subscribed users and processing subscriptions to the system.

v. External data

New registration report submitted by the IRA. This may be citizens or Government agencies.

4.2.6 System Functionality

- i. The system functionalities identified are: registration of health insurance firms, sending alerts to subscribers, and registration of subscribers, sending emails, searching of existing health insurance firm, searching of policy holders, conditions and treatment given to policy holders.

a) Health Insurance Registration Functionality

<p>NAME: Health Insurance Registration</p> <p>Description: Register a new health insurance firm in the system</p> <p>Percepts/events/messages: Services registered</p> <p>Message send : Registration completed successfully ; customized message</p> <p>Actions : display customized message</p> <p>Data used: Data supplied by the person registering the firm</p> <p>Interactions : data reception agent via communication agent</p>
--

Table 4.2

b) SENDING ALERTS

<p>Name : Sending alerts to subscribers</p> <p>Description: Alert subscribers that there is a new registration.</p> <p>Percepts/events/messages: new Health insurance firm record DB(customized message)</p> <p>Message sent: new registration occurrence (customized message)</p> <p>Actions: Display the record</p> <p>Data used: Registration DB, Subscribers' details</p> <p>Interactions: Communications manager via sms alerts</p>

Table 4.3

D) Searching the Existing Health Insurance Firms Records

Table 4.4

<p>Name: Search record of registered Firms</p>
<p>Description: perform search</p>
<p>Percepts/events/messages: record found in DB (customised message)</p>
<p>Messages sent: search results</p> <p>Actions: Display record</p> <p>Data used: Registration DB, subscribers' details.</p> <p>Interactions: Communications manager via search</p>

E) SUBSCRIPTION

Table 4.5

<p>Name: subscription</p> <p>Description: register users in the system so as to get alert messages</p> <p>Percepts/events/messages: successful (un)subscription (customised message)</p>
<p>Messages sent: subscription activated.</p> <p>Actions: subscribe/activate subscription/deactivate</p> <p>Data used: subscribers' details.</p> <p>Interactions: Communications manager via subscription</p>

F) WSIG configuration file

i. Agent Configuration

Table 4.6

host	Localhost
port	1099
Container-name	WSIG
Local-port	1200
wsig-agent	Com.tilab.wsig.agent.WSIGAgent
wsig.uri	http://localhost:8084/ws
wsig.console.uri	http://localhost:8084/
wsig.uri	http://localhost:8084/ws
wsig.timeout	30000

ii. DF to WSDL converter

Table 4.7

wSDL.localNamespacePrefix	impl
wSDL.style	document
wSDL.writeEnable	false
wSDL.directory	wSDL

iii. UDDI repository configuration

Table 4.8

uddi.enable	false
uddi.queryManagerURL	http://localhost:8084/juddi/inquiry
uddi.lifeCycleManagerURL	http://localhost:8084/juddi/publish
uddi.businessKey	8C983E50-E09B-11D8-BE50-DA8FBF3BDC61
uddi.userName	jdoe

iv. UDDI4j configuration

Table 4.9

org.uddi4j.logEnabled	true
org.uddi4j.TransportClassName	org.uddi4j.transport.ApacheAxisTransport

v. Ontologies

onto.multi-agent-mahiip-ontology	com.research.mls.onto.MAHIIPOntology
---	--------------------------------------

Table 4.10

4.3 Architectural Design

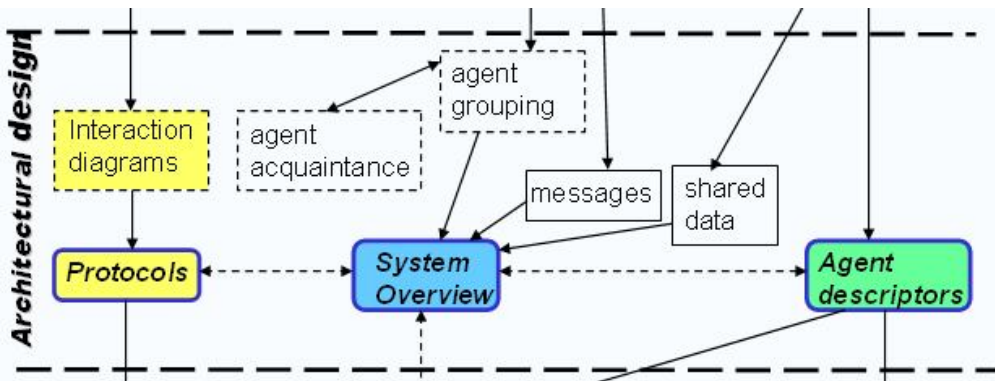


Figure 4.7 Architectural Design phase diagram

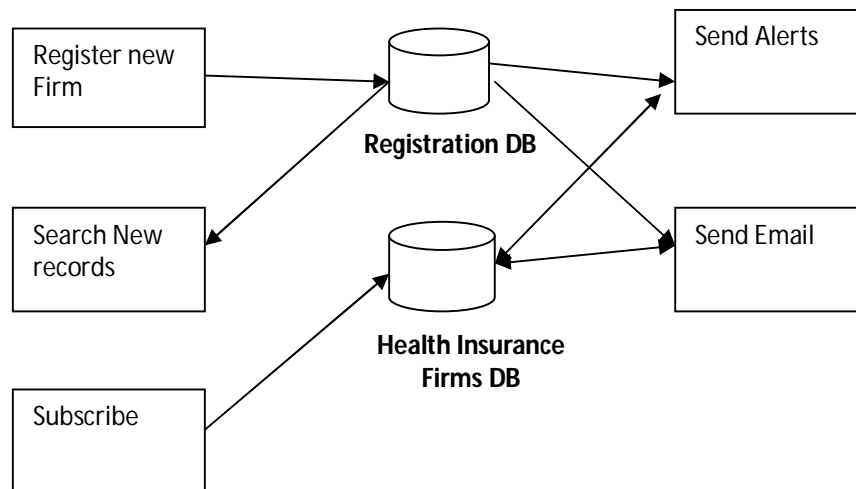
In the architectural design phase, the focus was on:

- i. Deciding on the agent types in the system: where agent types were identified by grouping functionalities based on considerations of coupling; and these were explored using a coupling diagram and an agent acquaintance diagram. Once a grouping was chosen the resulting agents were described using agent descriptors.
- ii. Describing the interactions between agents using interaction diagrams and interaction protocols: where interaction diagrams were derived from use case scenarios; and these were then revised and generalized to produce interaction protocols.
- iii. Designing the overall system structure: where the overall structure of the agent system was defined and documented using a system overview diagram. This diagram captures the agent types in the system, the boundaries of the system and its interfaces in terms of actions and percepts, but also in terms of data and code that is external to the system.

This is the second phase in Prometheus Methodology, during this stage the following activities

were performed:- identification of which agents should belong to the MAS, identification of groups of agents which share the same functionalities, identification of the agent acquaintance diagram which defined the links among interacting agents, definition of the agent descriptors, characterized by name, description, cardinality, functionalities, reads data, writes data, interacts with definition of the events, messages and shared data objects, identification of the system overview diagram which ties together agents, events and shared data objects and definition of the interaction diagrams and interaction protocols using Agent Unified Modelling Language (AUM). The system functionalities identified in the previous chapter are: registration of health insurance, sending alerts to subscribers, registration of subscribers, sending emails, searching of existing health insurance firm. The agents required to achieve these functionalities include, registration agent, persistence agent, IRA Data Access agent, and WSIG agents.

Figure 4.8: Data coupling diagram



4.3.1 Agent Descriptors

The **agent descriptors** for the identified agents are described below.

Table 4.11

<p>Name: WSIG – Container:control Description: receive all requests from users of the web interface and sends the responses back after agents have worked on them. Directs the requests to the appropriate agent(s). Lifetime : instantiated when system starts. Initialization: Reads user input Demise: Closes requests Functionalities included: Registration, search, sms sending , email sending Uses Data: Registration DB, SubscribersDB, usersDB</p>	<p>Name: Data Access Agent Description: receive data from the WSIG control container Lifetime : instantiated when a new registration is entered into the system Initialization: receives data from WSIG container Demise: Closes requests Functionalities included: Registration, Uses Data: registration data captured Goals: receive message from WSIG , return response to WSIG,</p>
---	--

<p>Goals: respond to web requests, direct requests to the appropriate agents, output results.</p> <p>Events responded to: new registration record,search request, sms and email alerts.</p> <p>Actions: display customized responses to search requests and successful entry of new records.</p> <p>Interacts with: data reception</p>	<p>Events responded to: new registration record</p> <p>Actions:</p> <p>Interacts with:WSIG, Data processing agent</p>
<p>Name: Data Persistence Agent</p> <p>Description: Receives data to be persisted into DB. Does data validation ;Enters data into db;Initiates communication to subscribers upon successful data persistence;Generates serial numbers.</p> <p>Lifetime: instantiated on receipt of new record.</p> <p>Demise: Application closes</p> <p>Initialisation: receives data to be persisted in DB.</p> <p>Functionalities included: registration</p> <p>Uses data: registrations records,</p> <p>Produces data: DB record</p> <p>Goals: enter data into db,</p> <p>Events responded to: new registration record;</p> <p>Actions: conversion</p> <p>Interacts with: communication agent,WSIG agent</p>	<p>Name :Communication agent</p> <p>Description: Responsible for general communication tasks.;SMS and email agents work under instruction from this agent.;Converts data objects into SMS and email objects.;Validates SMS email and data object data;Sends requests to sms and email agents</p> <p>Lifetime: Instantiated on receipt of SMS or Email request. Demise when a user logs out</p> <p>Initialisation: receives request , reads registration DB</p> <p>Demise: on close of DB connections</p> <p>Functionalities included: sending sms, sending email</p> <p>Usesdata: deaths DB,</p> <p>Events responded to: new arrival; customer query; customerpurchase;creditcheckresponsecustomerresponse;</p> <p>Actions: convert data into sms and email objects.</p> <p>Interacts with: sms sending, email sending , record searching</p>

4.3.2 Interaction Diagram of the System

Figure 4.9i

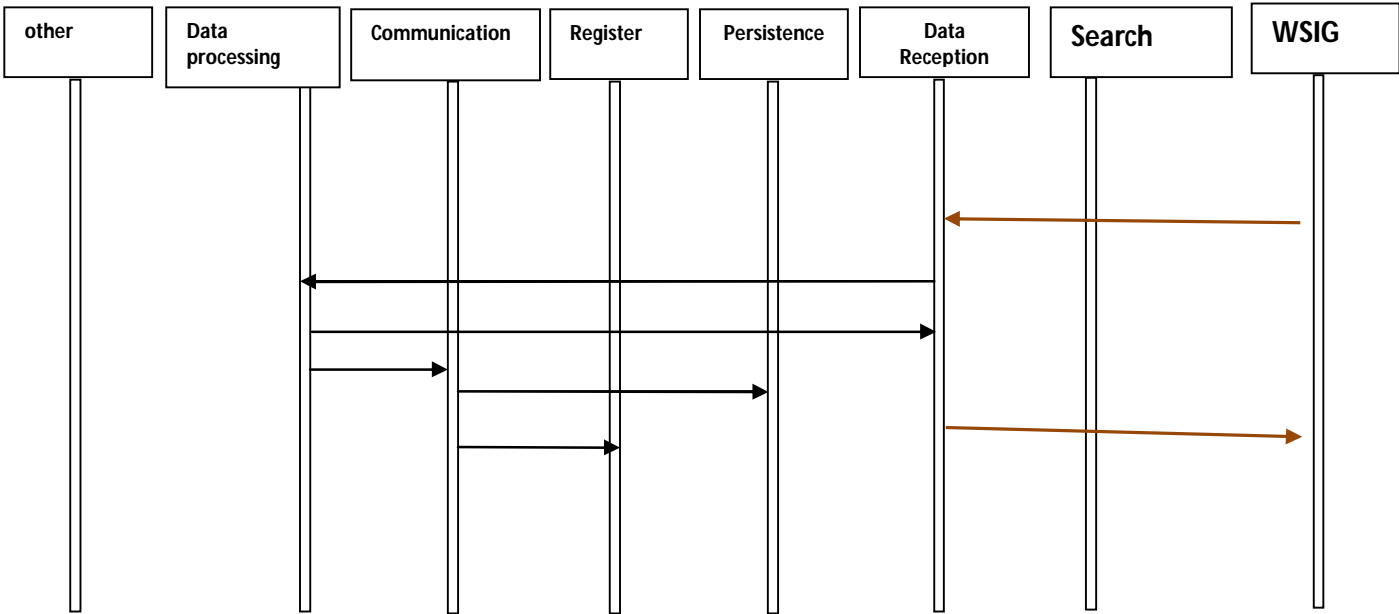
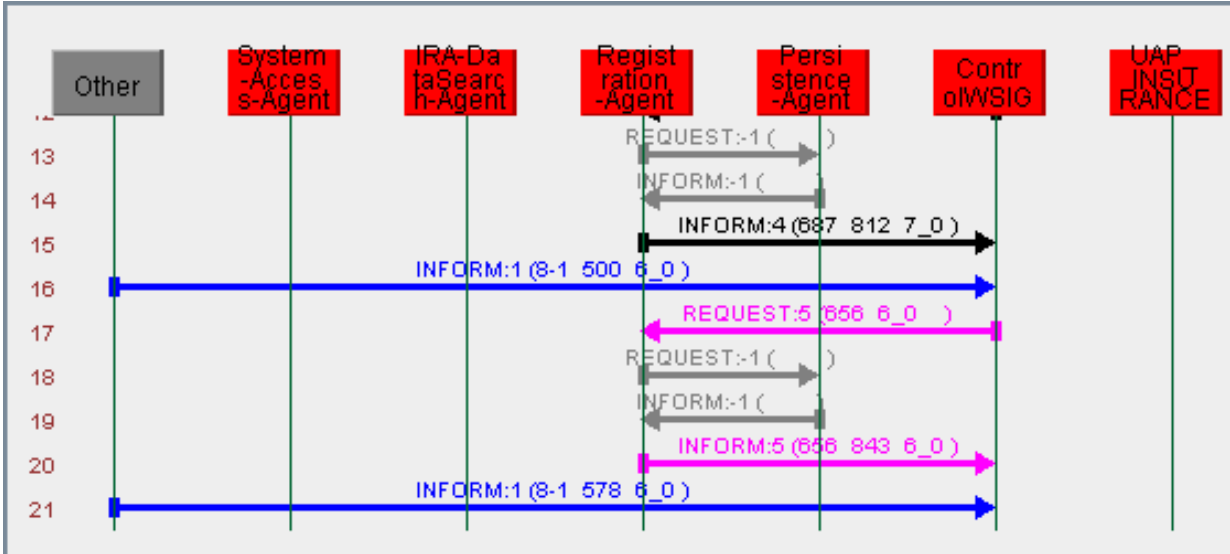


Figure 4.9: IRA Interaction Diagram



4.3.3 Registration Agent -This agent handles registration to and deregistration from the platform by external system. It also validates the registration requests, approves or denies registration. It also validates existing registration details.

4.3.4 IRA Data Access Agent -This agent receives client requests for data from the WSIG agent. It instructs the appropriate external system agents to search for the required data and return the results. This agent also compiles the results from the different external systems and sends them back to the client.

4.3.5 External System Agents -These agents provide access to the external systems registered onto the platform. There are as many of them as external systems registered on the platform. They perform search for requested data on their respective platforms under instruction from the Data Access Agent. They wrap data from different sources into one standard format understood by all external systems.

4.3.6 RMA, DF and AMS agents -These are system agents, shipped with the JADE distribution; HostMonitor can monitor remote networks using Remote Monitoring Agents (RMA). RMA is small application that accepts requests from HostMonitor, performs test (or action) and provides information about test result back to HostMonitor.

4.3.7 Persistence Agent -This handles database operations for the platform.

4.3.8 WSIG Agent -The WSIG Agent is the gateway between the web and the Agent worlds and is responsible for forwarding agent action received from the WSIG Servlet to the agents

actually able to serve them and getting back responses, the WSIG Agent is also responsible for subscribing to the JADE DF to receive notifications about agent registrations/deregistration, is also responsible for creating the WSDL corresponding to each agent service registered with DF and publish the service in a UDDI registry if needed.

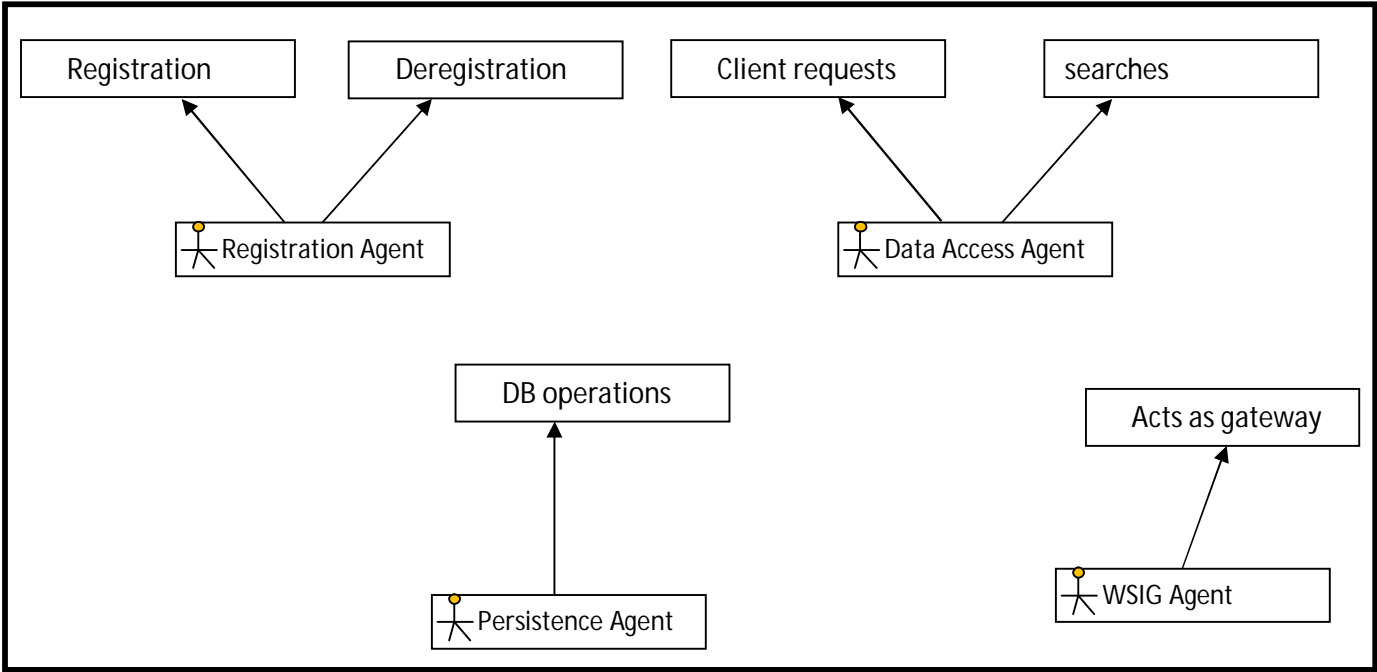
4.3.9 Platform Monitor Agent -This agent initializes the platform and agents. It creates a link to the platform database and it ensures that all the required parts of the platform are running, these may include database and all the required agents.

4.3.10 Agent-Role Grouping Diagram

In the below diagram, we grouped the roles into agent types. Decisions regarding how to group depended on role similarity, as well as analysis of data usage. The agent-role coupling diagram showed the group of roles that came under an agent. The agents were connected to their roles using edges. An agent role coupling diagram contained the following entities in the toolbar:

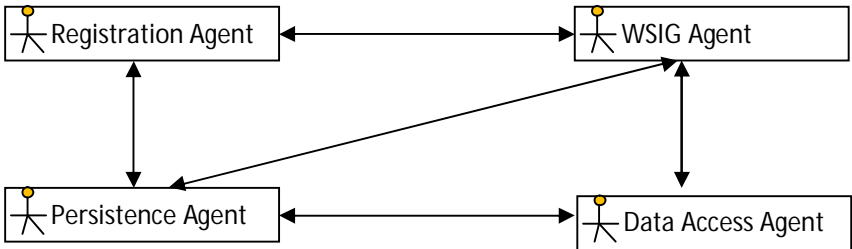
- 1. Agent
- 2. Role
- 3. Edge

Figure 4.10: Agent Role Grouping Diagram



4.3.11 Agent Acquaintance Diagram

In the agent acquaintance diagram you could see all agents within the system and which agents interact. An agent acquaintance diagram contains only one type of entity; an agent. **Figure 4.11**

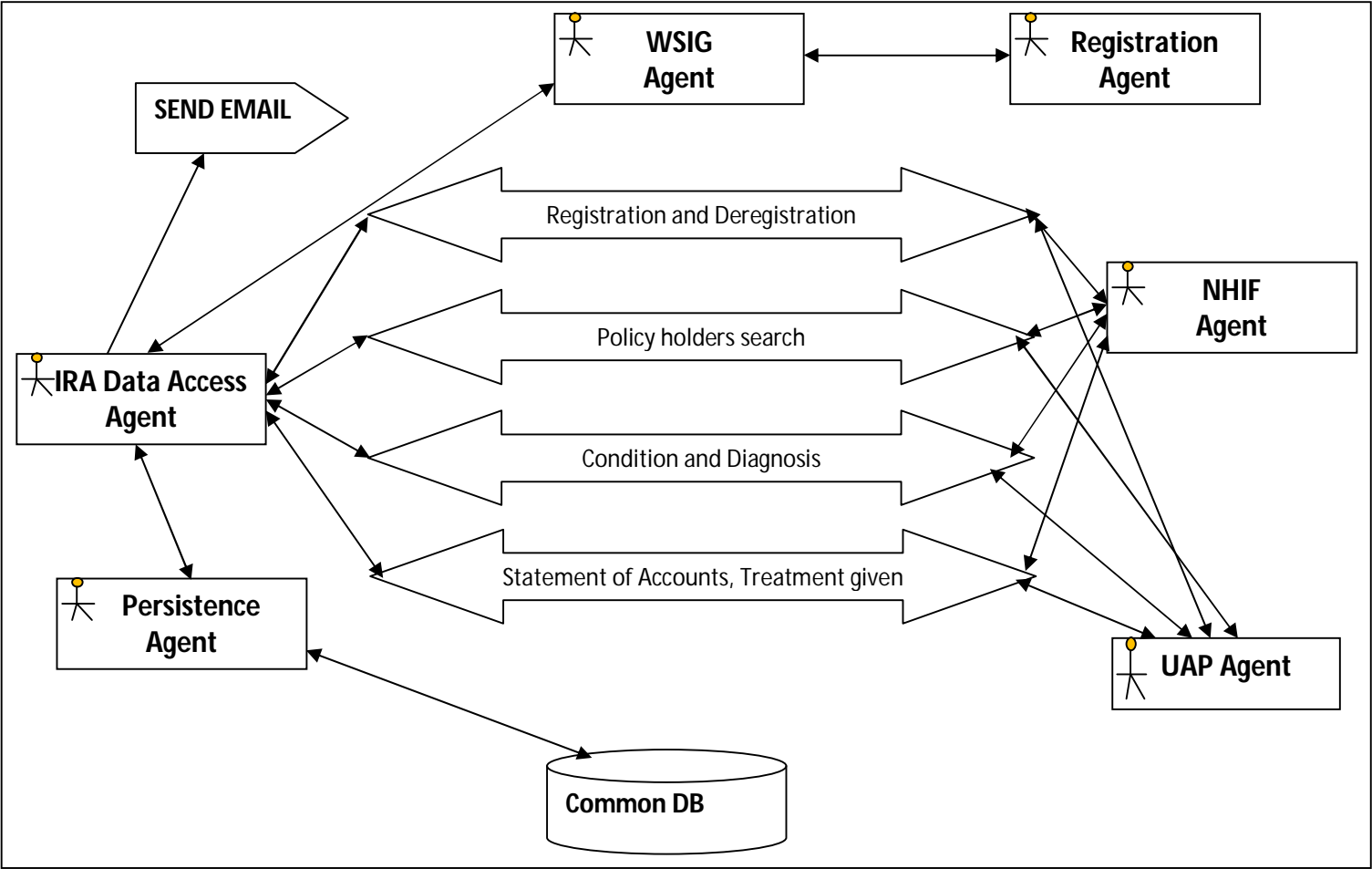


4.3.12 System Overview Diagram

In the system overview diagram you could see all agents in the system, along with their interface and interactions. This diagram was the central diagram of the entire system design. It was developed directly, and inherited the agents from the Agent-Role Coupling diagram. Percepts and actions were inherited from the System Specification phase. Protocols were linked to agents directly within the diagram. The links were inferred from the specification of the protocols. A system overview diagram contains the following types of entities:

1. Agent
2. Action
3. Data
4. Message
5. Percept
6. Protocol
7. Edge

Figure 4.12: System Overview Diagram



4.4 Detailed Design

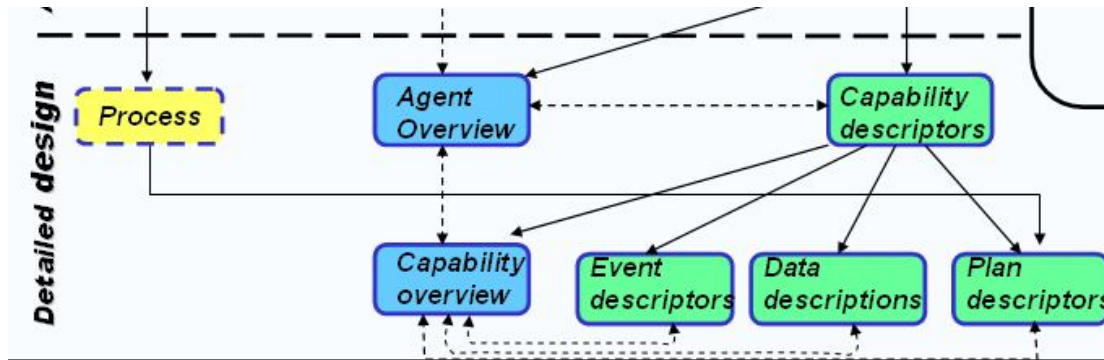


Figure 4.13 Detailed Design Phase diagram

Detailed design focused on developing the internal structure of each of the agents and the behavior of the agents. The agent's internal events, plans and detailed data structures. This phase also consisted of developing the internals of agents, in terms of capabilities and, in this case directly in terms of events, plans and data. This was done using agent overview diagrams and capability descriptors, developed process diagrams from interaction protocols.

Developed the details of capabilities in terms of other capabilities as well as events, plans and data. This was done using capability overview diagrams and various descriptors. A key focus was developing plan sets to achieve goals and ensuring appropriate coverage.

Capabilities are a structuring mechanism akin to modules. A capability contained plans, data, and events. It also contained other capabilities allowing for a hierarchical structure.

In identifying the capabilities that each agent type contained, the study usually started by considering a capability for each functionality that was grouped in the agent type. This initial detailed design was then refined by merging capabilities that were similar and small, splitting capabilities that were too large and adding capabilities that corresponded to common library code.

The structure of each agent was depicted by an agent overview diagram. This was similar to the system overview diagram except that it did not contain agent nodes and does not usually contain protocol nodes. However, the agent overview diagram does usually contain capability nodes and plan nodes.

4.4.1 Agent Capabilities

The capabilities of each agent were identified as follows:

Table 4.12

Agent	Capabilities
WSIG – Container	Interface between the web and the other agents
Registration Agent	This agent handles registration to and deregistration from the platform by external system. It also validates the registration requests, approves or denies registration. It also validates existing registration details
Persistence Agent	This handles database operations for the platform. Ensure data is persisted into the DB and initiate communication with subscribers.
Platform Monitor Agent	This agent initializes the platform and agents
IRA Data Access Agent	This agent receives client requests for data from the WSIG agent. It instructs the appropriate external system agents to search for the required data and return the results. This agent also compiles the results from the different external systems and sends them back to the client.
Data reception agent	Receive data entered into the database and validates it.
RMA, DF and AMS agents	These are system agents, shipped with the JADE distribution; HostMonitor can monitor remote networks using Remote Monitoring Agents (RMA)
Communication agent	For general communication and send requests to sms and email agents.
Search agent	Perform data search

4.4.2 Capability Descriptors

The capability descriptors have been generated below:

Table 4.13

Name: handle search requests

External interface to the capability: user enters search string

Natural language description: produce search results

Interaction with other capabilities: Transport capability

Data used/produced by the capability: search results

Inclusion of other capabilities: None

Name: Registration

External interface to the capability: new record found.

Natural language description: send alerts msg to subscribed users.

Interaction with other capabilities:

Data used/produced by the capability: alerts object

Inclusion of other capabilities: None

Name: data access

External interface to the capability: Data entry

Natural language description: enter data into the deaths data base

Interaction with other capabilities: communication agent

Data used/produced by the capability: database record

Inclusion of other capabilities: None

4.4.3 Agent Overview diagram

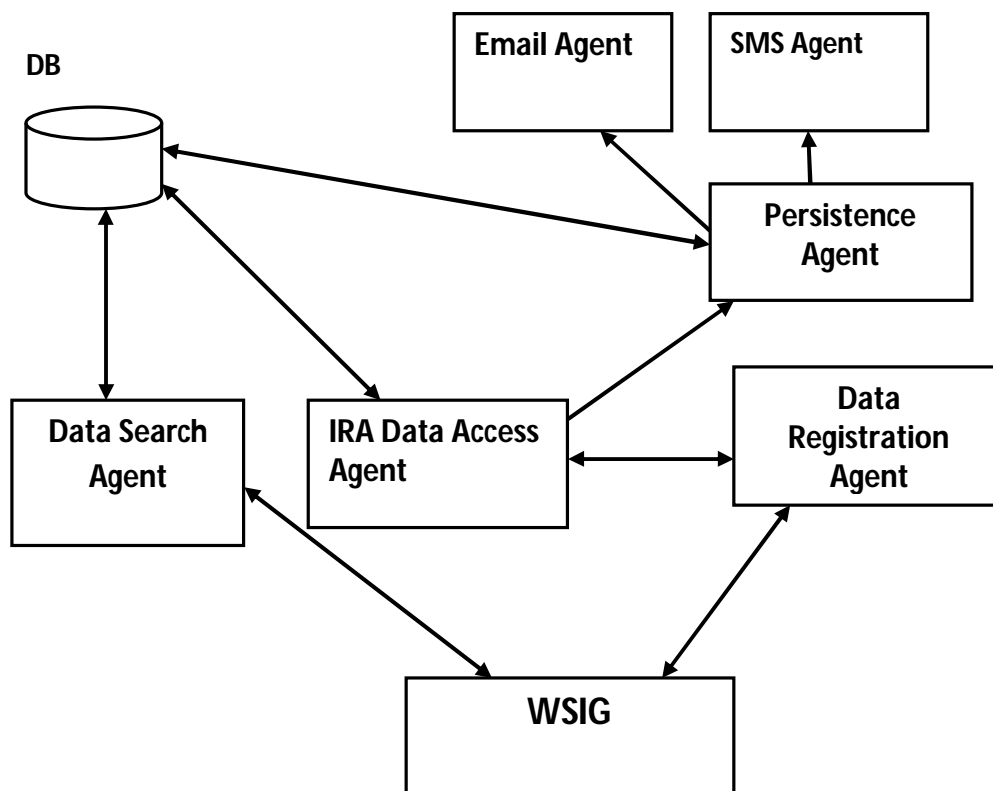


Figure 4.14: Agent Overview Diagram

5.0 CHAPTER FIVE: IMPLEMENTATION AND RESULTS

5.1 Implementation tools

The multi-agent based system for registration of health insurance firm services has been implemented using a number of analysis and development tools.

MySQL : for database management.

SQLite3: for database management

PhpMyAdmin 3.4.9 : a graphical interface for Mysql database (NHIF Client Application).

WampServer: a Windows web development environment which allows us to create web applications with Apache2, PHP and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your databases.

Java EE using Netbeans IDE (for IRA System and IRA MAS) JSP (Java Server Pages.

Apache Tom cat 5.5.35Servlet Container

Apache, php webserver

Python application for the webserver application (UAP Client Application).

5.2 System testing and evaluation

Testing is a process of detecting errors in a software program. The detected errors must be corrected to ensure that the software meets its specifications and user requirements. The testing which was applied during this study was static and dynamic. The static testing process involved reading and checking documents concerning requirements, reading and checking of code without running the system. The dynamic testing process involved running the code to check its output. The dynamic testing included two approaches; black box testing and white box testing. In this study the black box testing treated the system as a black box that received inputs and produce outputs; it was based on assessment of requirements and functionality of the system. Also in this study the white box testing involved the coverage of code statements, branches, paths and condition of execution, during white box testing, the internal logic of the platform's code was tested. The system was tested with random records. Due to the sensitivity of health insurance data, it was not possible to use live data for testing purposes. The simulated data was therefore used to test the operation of the developed system. The system was able to transmit updates to subscribed users. The study covered only types of dynamic testing and evaluation of results. In order to conduct a proper testing process, a test plan was required. The contents of the plan included the following:-

- i. The identity of the component to be tested
- ii. The purpose of the test

- iii. Condition for carrying out the test
- iv. Test data to be used (correct and incorrect data)
- v. Expected results

5.2.1 Component testing

The study used this type of testing to test each individual module in a suite of programs. Each module was tested with correct and incorrect data to reveal loopholes in the code. This testing often takes place in parallel with coding to provide intermediate results that could reveal logic, functionality and error handling of various units in the system. This testing was done at the source or code level for language, specifically for identifying programming errors such as bad syntax, logic errors or to test particular functions or code modules.

5.2.2 Integrating Testing

Specific functional user requirements were tested to ensure that the system meet these requirements and ensured that the developer has developed the right product. Each user specific function was thoroughly tested to ensure that they perform their tasks properly. The test tried to find errors within the inputs and outputs of these functions. The results display and format would be closely examined to make sure that it meets user requirements. These tests included the following:-

- i. The interfaces to ensure that they are functioning well by checking if the interface components are properly integrated and behaving as expected especially verifying that the appropriate action is associated with correct interaction method of designed mouse click events, key strokes, etc.
- ii. The interaction of the system's graphical user interface and the backend database. In this study, data would be inserted from the front end using data entry forms and check if each entry corresponds to the correct database table fields, also check if data inserted is in the desired format.
- iii. The form interaction design would also be tested to ensure that the information presentation to the user is in a manner that the users can easily find and/or perform the operation they want. The design should be an abstract representation of the abstract physical forms that the system employees use.

5.2.3 System Testing

The platform was properly tested to verify its functionality. This was achieved by ensuring that relevant tests were performed such as integration tests, unit tests, functional tests and acceptance tests. This test would be used as the overall testing of the system after the system has fully been developed just before it is taken to the client, where the client will perform acceptance testing

with the developer and raise any adjustments they would like to be included at different system levels. This type of testing would be designed to ensure that the system meets all functional and non functional user requirements. The testing would also focus on the behavior of the system once it is subjected to different inputs. It would also focus on how different components of the system interact with each other for optimal performance of the system (i.e integration testing). Different user demand would be tested against the system which would include the error message testing and several screen mappings.

5.2.4 Acceptance Testing

This is done by the system client and after all other types of testing have been performed. The system is taken to the IRA Technical Department then to the Commissioner to test whether the system is adhering to the agreed upon functional requirements in System requirements Specification Document (SRS). The client (IRA) also uses black box testing to evaluate the system functionality. Mission critical errors in functional requirements are corrected immediately and regression testing performed to ensure that changes made do not affect other system components or functions. Upon successful acceptance testing, the system will then be implemented; a process that will see the developed system is in place and working as intended.

5.2.5 Performance Testing

This type of testing relates to the expected level of the processing time of the transaction or response time especially when the system is being fully utilized.

5.2.6 Volume Testing

This type of testing ensures that the system can handle the expected number of users or transaction's processing speed

5.2.7 Regression Testing

This type of testing involves ensuring that the system corrections do not result in the same or other errors as corrections are being made.

5.2.8 Evaluation Testing

For accurate results on testing, once alpha test is finished an independent body will be assigned to carry out a beta test. This will guarantee software quality since the body does not have idea on the system structure. Therefore, they will use different test data which may help in discovering more errors. Their recommendations will be considered and errors corrected accordingly. Later the system will be released to the end user. For the purposes of software maintenance and documentation, all the test records and test logs will be recorded and documented for future reference both for alpha and beta tests.

5.3 Tests for the Developed Systems Table 5.1

No.	Component Tested	Purpose	Condition	Test Data	Expected Results
1	CollectionUser	Correct Login to display main GUI	Login GUI	Username and Password	Main menu form display
2	PasswordChange	Successful password change	Change password GUI	Username, old password and new password	Old password replaced by new password
3	Systemmenu	Correct access to all menu items	Main menu form	Selected menu items	Display of the selected menu item
4	NewUser	Successful creation of new user	Create new_user	userId and name	Creation and storage of new user
5	RegistrationAgent	Fire registrationAgent	GUI commandline interface	Java jade. Boot_host register: registrationagent	Login GUI displays
6	DeleteUser	To Confirm deletion of the selected user	Delete_User GUI	UserId	Successful removal of the selected user data from the user table
7	AddProvider	Confirm successful addition of new provider	Add new_provider GUI	provider_details	Addition of and storage of new provider
8	SearchPolicyHolder	Search for existing policyholder	Type a letter or word	Policy_holder name	Display policy holder details

5.3.1 IRA MAS

First, we run the Insurance Regulatory Authority Multi-Agent System, so that the registered and active agents would be displayed, the successful test run is as shown below:-

```
Agent container Interoperability Agents@nhif-1b6cf0abd2 is ready.
-----
Registration-Agent: Registered: registration
IRA-DataSearch-Agent: Registered: data-search
System-Access-Agent: Registered: system-access
Persistence-Agent: Registered: persistence
[EL Info]: 2013-07-30 13:51:14.562--ServerSession(20823247)--EclipseLink, version:
Eclipse Persistence Services - 2.3.2.v20111125-r10461
[EL Info]: 2013-07-30 13:51:16.093--ServerSession(20823247)--file:/C:/project/IRA
MAS/build/classes/_IRA_MAS_PU login successful
[EL Warning]: 2013-07-30 13:51:16.265--ServerSession(20823247)--Exception
[EclipseLink-4002] (Eclipse Persistence Services - 2.3.2.v20111125-r10461):
org.eclipse.persistence.exceptions.DatabaseException
Internal Exception: com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException: Table
'platformconnection' already exists
```

Figure 5.1



5.3.2 Login to IRA System

This is a web application platform where all the External agents would be registering their services.



The screenshot shows the login interface for the Insurance Regulatory Authority (IRA). At the top left is the IRA logo, consisting of three overlapping diamond shapes in shades of green and gold, with the letters 'IRA' below it. To the right of the logo, the text 'INSURANCE REGULATORY AUTHORITY' is displayed in a bold, gold-colored font. Below this, the text 'Login to continue' is centered in a dark blue font. The login form consists of two input fields: 'Username' and 'Password'. The 'Username' field is empty, and the 'Password' field is also empty. A 'Login' button is positioned to the right of the 'Password' field. The entire form is enclosed in a thin black border.

Figure 5.2: Login to IRA System

The system users, providers (e.g IRA, UAP, NHIF) must be provided with a password and a username in order to access the platform.



This screenshot shows the same login interface as Figure 5.2, but with the input fields filled. The 'Username' field contains the text 'liech|', and the 'Password' field contains five black dots. The 'Login' button remains visible to the right of the password field. The layout, including the logo and header text, is identical to the previous screenshot.

Once the user has logged in the following screen would appear where they would be able to search for policy holders, add providers and even the users of this platform.

The screenshot shows the IRA dashboard with a navigation bar containing buttons for Home, Search Policy Holders, Add Provider, Add User, System information, and Logout. Below the navigation bar is the heading "Registered Health Insurance Providers". The main content area displays the logo for UAP Insurance (a red square with "UAP" in white) and the tagline "Better. Simple. Life.". To the right of the logo, the following information is listed: Name: UAP INSURANCE, State: ACTIVE, and two links: [Edit](#) and [View](#). Below this information is the text "UAP INSURANCE" followed by the website URL: <http://localhost:8000/admin/records>.

5.3.3 Search for Policy Holders named containing letter “m

Figure 5.3

The screenshot shows the IRA dashboard with the same navigation bar as the previous figure. Below the navigation bar is the heading "Search Policy Holders". Underneath the heading is a search input field containing the letter "m" and a "Search" button.

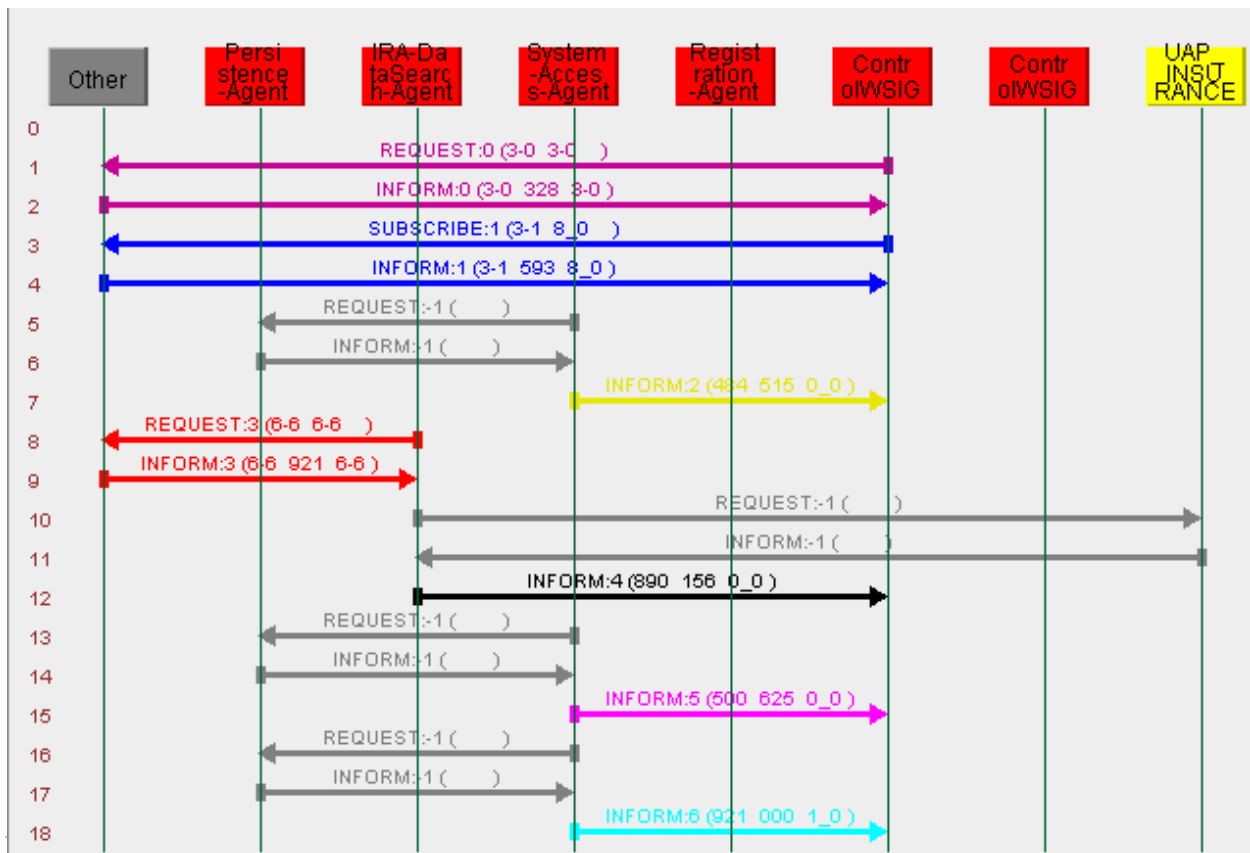
5.3.4 Search Result for Files or Folders named containing letter “m”

This results show that two health insurance policy holders are found in UAP databases, they are Kim John Stamu boy and Muturi Kioko Amsoni. Their fullnames, citizenship, Date of birth, Gender, National Identification, Relation to Contributor and policy could be viewed from this platform. **Table 5.2**

INSURANCE REGULATORY AUTHORITY IRA											
Home Search Policy Holders Add Provider Add User System information Logout											
Search Policy Holders											
<input type="text" value="m"/> <input type="button" value="Search"/>											
1	Full Names	Citizenship	Date of Birth	Gender	National Identification	Relation To Contributor	Policy	Ailments Suffered			
								Condition	Admitted	Discharged	Treatment
2	Kim John Stamu boy	Kenyan	2013-06-10	M		self	Default [UAP_INSURANCE]	1 Common Cold	2013-06-11 17:14:05	N/A	Givern tabs
3	Muturi Kioko Amsoni	Kenyan	1900-06-10	M		child	Children Cover [UAP_INSURANCE]	1 Common Cold	2013-06-10 13:50:43	N/A	Syrup
								2 Malaria	2013-06-10 14:03:07	2013-06-10 14:03:08	AL

8.3.5 Agent Communication and Interaction Protocol

During the search for Files or Folders named containing letter “m”, the agents were seen as exchanging messages, informing and requesting each other as shown in the figure below. **Fig 5.4**



5.4 Evaluation of the multi-agent based system

Test results determined the behavior of the system when it was subjected to load. The results were analyzed and interpreted to establish the correct operation of the developed system. The evaluation of the multi-agent based system was done with the users in various health insurance firms so as to get comparative views of the existing system and the multi-agent based system. Evaluation involved data collection and data analysis.

Testing subscription: The recipients of the alerts first subscribed to receive updates. Those with incorrect details did not receive alerts.

Testing data capture: Only the registered health insurance firm were able to add new records to the database.

System evaluation was carried out to ensure that the developed system met the user requirements and system specification. When the above tests were conducted the functionalities of the system were successfully evaluated. After the evaluation, the system was able to achieve the following:-

- i. To add health insurance providers which enable us to understand how a data sharing platform/interoperability system could be implemented
- ii. To search the existing policy holders from various sources which helped us to conclude that software agents could be used to implement a data sharing platform.
- iii. Provision of secure access to the system
- iv. Interaction with the System Administrator and users using the GUI interface
- v. Accessing number of policy holders in each and every health insurance firm
- vi. Regular premiums payment
- vii. Policy holders' details e.g. dependants
- viii. Public Consumer education dates
- ix. Treatment given to policy holders
- x. Condition / Diagnosis
- xi. Statement of accounts
- xii. Date of Admission
- xiii. Date of Discharge
- xiv. Insurance Structure
- xv. Analysis of application and registration forms
- xvi. Turnover –submission of data, Monthly, Quarterly and Annually
- xvii. Application –state how to apply for the health insurance products

5.5 Discussion of Results

Many organizations dealing with health insurance such as NHIF, UAP Insurance, Jubilee Insurance, among others capture data that is relevant to their operations and hardly create interfaces to other stakeholder databases. They constantly add new clients and remove dormant ones in order to keep their databases up to date. This research project focused on an agent-based platform, the system allows IRA to receive and exchange data from distributed sources.

This interoperability system makes public and private health insurance data accessible to other registered insurance firms. It optimizes the management and timely dissemination of relevant information and data-driven services to citizens, professional and internal Government audiences, in self-service mode and across multiple channels. While the IRA process of carrying out this work is slow and does not produce timely and accurate data, the multi agent based system can efficiently carry out this process and avail data to all interested stakeholders mostly the Government Agencies and Ministry of Health, who require the data for economic reports and health planning.

6.0 CHAPTER SIX: CONCLUSION AND FUTURE WORK

6.1 Conclusion

The health insurance sector is embracing the use of ICT in its operations. Interoperability system is a rapid growing industry. The Information Technology industry has been very dynamic for the last couple of decades. These health insurance applications have been developed using different methodologies, and technologies. These applications cannot be integrated, which becomes an issue to interoperability in health insurance. However, IRA or the government has not been able to come up with a proper technological solution to the problem availability of timely and meaningful data.

This study has seen that this multi-agent based system can provide a solution to inaccurate data, incomplete records and delayed dissemination of health insurance data to stakeholders. The study has also shown that it is very easy to model interactions of agents; also it is very easy to build agents using experiments. Updates are transmitted as soon as they occur and interested users of the data can subscribe to receive updates. The results were analyzed and interpreted to establish the correct operation of the system.

6.2 Recommendation for Future Work

Based on the interpretation of the results, further study is recommended on the developed system. The evaluation of the multi-agent based system was done with the users in various health insurance firms so as to get comparative views of the existing system and the multi-agent based system. Evaluation involved data collection and data analysis.

6.3 Challenges

In this research study, many challenges were encountered which included the following:-

- i. Since it was a health insurance issue, the respondents were reluctant to communicate private and confidential details from their database
- ii. Most of the respondents were very busy and had very little time for detailed discussion; this hindered the data collection process, hence overall system development.
- iii. During system testing process, we had to use simulated data since the health insurance data are sensitive, this made the testing of the system to be difficult and lengthy because we had to take sometime to generate the simulated data.

- iv. Programming and coding the agents was difficult since we had to learn some new techniques in Python, Java, Netbeans, PhP, SQLite3 and others to perform proper system implementation.
- v. Implementing multi-agent systems is still a complex task due to
 - ✓ Lack of maturity in both methodologies and programming tools
 - ✓ Lack of specialized debugging tools
 - ✓ Lack of skills needed to move from analysis and design to code
 - ✓ problems associated with awareness of the specifics of different agent platforms
 - ✓ Problems in understanding the nature of what is a new and distinct approach to systems development

APPENDIX I: References

1. Aameek Singh, Ling Liu; Storage Systems, IBM Almaden Research Center
2. Abbas Rajabifard (2010): Data Integration and Interoperability of Systems and Data
3. Adina Magda Florea (2003): Intelligent agents on the Web (2003), 5th International Workshop on Symbolic and Numeric Algorithms for Scientific Computing Timisoara, Romania, 2003
4. Anson Lee, Michael Lyu, Irwin King (2002): Agent-based Multimedia Data Sharing Platform
5. Asuman Dogac, SRDC Ltd (2012): Interoperability in eHealth Systems
6. Centers for Medicare & Medicaid Services (2011): Guidance for Exchange and Medicaid Information Technology (IT) Systems
7. Christopher S. Sears, Esq., Victoria M. Prescott, Esq., Clement J. McDonald, M.D. (2005),The Indiana Network for Patient Care: A Case Study Of A Successful Healthcare Data Sharing Agreement
8. Claire Broome, M.D., Centers for Disease Control and Prevention (2002): Facilitating electronic sharing of data needed for public health preparedness: the National Electronic Disease Surveillance System (NEDSS)
9. D. Van Den Abeele, J Szymanski , C Gransart, M Berbineau (2010): Innovative Data Sharing Platform for business performance improvement
10. Eric Leclercq, Djamal Benslimane, Kokou Y´etongnon (2011) ISIS: A Semantic Mediation Model and an Agent Based Architecture for GIS Interoperability
11. Fang Cao, Norm Archer and Skip Poehlman: Journal of Emerging Technologies In Web Intelligence, Vol. 1, No. 2, 2009: An Agent-based Knowledge Management Framework for Electronic Health Record Interoperability
12. Fang Cao, Norm Archer and Skip Poehlman (2011): An Agent-based Knowledge Management Framework for Electronic Health Record Interoperability
13. Gabriel Antoniu, Marin Bertier, Luc Boug´e and Eddy Caron (2005) GDS: An Architecture Proposal for a Grid Data-Sharing Service
14. Gokce B. Laleci Erturkmen, Asuman Dogac, Mustafa Yuksel, Sajjad Hussain, Gunnar Declerck, Christel Daniel, Hong Sun, Kristof Depraetere, Dirk Colaert , Jos Devlies, Tobias Krahn, Bharat Thakrar, Gerard Freriks, Tomas Bergvall, Ali Anil Sinaci (2011): Building the Semantic Interoperability Architecture Enabling Sustainable Proactive Post Market Safety Studies

15. Gonçalo Jorge Rodrigues Plácido, Carlos Rompante Cunha and Elisabete Paulo Morais (2011): Promoting Ubiquity and Interoperability among Health Information Systems Using an SOA Based Architecture
16. Hongqiao YANG, Kecheng LIU and Weizi LI (February 2010): Adaptive Requirement-Driven Architecture for Integrated Healthcare Systems
17. MD. Nurul Huda*, Noboru Sonehara, Shigeki Yamada (2009): A Privacy Management Architecture for Patient-Controlled Personal Health Record Systems
18. Michael M. Gorman (2008): A Metadata Architecture For Enterprise-Wide Data Sharing (2008)
19. Michael Ndirangu Mungai (2011), Multi Agent based Traffic Controls
20. Michael Wooldridge (2002): An Introduction to Multi-agent Systems, England, John Wiley & Sons Ltd
21. Paul HO (2008) Agent Based Approach to Interoperability
22. Prasanna Desikan, Kuo-Wei Hsu, and Jaideep Srivastava (2011): Data Mining for Healthcare Management
23. Prof. Deoki Nandan Director, National Institute of Health & Family Welfare, New Delhi, Overview on Models of Public Private Partnership n Nrhm
24. R.Kailar, V.Muralidhar, American Medical Informatics Association (AMIA) Annual Symposium, Chicago, 2007: A Security Architecture for Health Information Networks http://assets1.csc.com/lef/downloads/CSC_Papers_2009_Security_Architecture.pdf
San Jose, CA, USA (2006) SHAROES: A Data Sharing Platform for Outsourced Enterprise Storage Environments. last accessed in October 2012
25. SeoyeonKang, Haewoon Kwak, Keon Jang, Sue Moon (August 15th, 2008 1st CAIDA-WIDE-CASFI Workshop) CASFI Data-sharing Platform
25. Shailendra Singh, Bukhary Ikhwan Ismail, Fazilah Haron, and Chan Huah Yong (2004) Architecture of Agent-Based Healthcare Intelligent Assistant on Grid Environment
26. Syed Mohamed Aljunid, Samrit Srithamrongsawat, Wen Chen, Seung Jin Bae, RPh, ScD5, Raoh-Fang Pwu, Shunya Ikeda, Ling Xu(2012): Health-Care Data Collecting, Sharing, and Using in Thailand, China Mainland, South Korea, Taiwan, Japan, and Malaysia
27. The Boston Consulting Group (2012), The Socio-Economic Impact of Mobile Health
28. Tinghuai Ma, Hao Cao, Donghai Guan and Sung Young Lee (2010): An Efficient Data Sharing systems based on Agents

29. Vagelis Hristidis and Eduardo Ruiz (2009) CADs: A Collaborative Adaptive Data Sharing Platform(2009)
30. Veerle Van den Eynden, Louise Corti, Matthew, Woollard, Libby Bishop and Laurence Horton, First published 2009, Third edition, fully revised, 2011 Managing and Sharing Data
31. Vinay Chavan and Sanjay E. Yedey (2008) Coordination Model for Communication among Agents in Multi-Agent based Intelligent Systems
32. Zablon Ochomo (2009), Hybrid Agents Coordination in a Virtualized Environment
33. Lin, F., Holt, P., Leung, S. and Li, Q. (2006) A multiagent and service-oriented architecture for developing adaptive e-learning systems, *Int. J. Continuing Engineering Education and Lifelong Learning*, Vol. 16, Nos. 1/2, pp.77–91.
34. (Brian Henderson- Sellers, Paolo Girogini (2005). Agent oriented methodologies. Idea Group Publishing)):
35. Wooldridge, Jennings, et al.(1999), A methodology for agent-oriented analysis and design
36. E-health Strategies Database: <http://ehealth-strategies.eu/database/database.html>, last accessed in November 2012
37. World Health Organization: <http://www.who.int>, last accessed in December 2012

APPENDIX II: Installation Manual

NHIF

1. Install and run wampserver
2. Copy the project folder to the www folder of the installation above
3. Create a Mysql database and populate its tables from the sql file: nhif.sql in the same directory as this help file.
4. Adjust database properties appropriately in {PROJECT_ROOT}/fct/db.php
5. Copy the file logo.jpg to the www folder too (C:\wamp\www)
6. Navigate to the project on your browser on the address: <http://localhost/nhif>
7. Installation successful

***** **

UAP

1. Make sure python is installed
2. Install Django framework for python
3. Install SQLite3
4. Copy the UAP folder to your preferred destination
5. Double click the file run.bat to start this application
6. Go to your browser and access the site at <http://127.0.0.1:8000/admin/records/>

IRA MAS

Ensure you have jdk version 7 installed

Ensure Netbeans ide

This is a Netbeans project that uses a Mysql database

Create a Mysql database named mahiip and generate the structure from the file mahiip.sql. The default connection parameters are user=root, password=,host=localhost,port=3301

1. Open the project in Netbeans
2. If your database connection is different, edit the persistence unit file META-INF/persistence.xml appropriately
3. Right-click the project and run.

If you don't want to run in this way, build the project on Netbeans and get the output jar in the project dist/ directory

You can run this jar by navigating to this directory and typing `java -jar <jar-name-without-the-brackets>`

IRA

Ensure you have jdk version 7 installed

Ensure Netbeans ide

This is also a Netbeans project,that depends on the IRA MAS project before you run this, make sure MAS IRA is running

1. Open the project in Netbeans

2. Right click the project and run alternatively, right click the project and build.
From the nbdist/dist folder get the generated war file deploy this war file on a
jboss/tomcat/glassfish/jetty.... server running at the port 8084 and restart/start the server access
the server on your browser at http://localhost:8084/

The above instructions assume that you are running on a windows machine, and using default
settings.

Please tailor them appropriately to suite your environment as all the projects are cross-platform
and will run on any system as long as all their requirements are met

Resources Required and Used

UAP Client Applications

- ✓ Python – for coding purposes
- ✓ SQLite3 – for Database Management

NHIF Client Application System

- ✓ Php – for coding purposes
- ✓ mySQL – for database Management (nhif)

IRA System

- ✓ We use JADE and WSDL (web service Definition Language –this is a WSIG add-ons)
- ✓ Java (JSP –Java server pages) – for coding purposes

IRA MAS

- ✓ Java – JADE Framework (for coding purposes)
- ✓ mySQL – for Database Management (mahiip –multiagent health insurance integration platform)

APPENDIX III: Introduction Letter to Insurance Regulatory Authority



UNIVERSITY OF NAIROBI

SCHOOL OF COMPUTING AND INFORMATICS

Telephone: 4447870/4444919/4446544

Telefax: 4447870

Email: moturi@uonbi.ac.ke

P. O. Box 30197

Nairobi

Kenya

10 April 2013

Insurance Regulatory Authority
Attn: Benard Gitangi -Technical Department

LIECH JAMES GOR P58/63063/2011

MSC RESEARCH PROJECT

The above named is bona fide student in the MSc in Computer Science of this University. As part of the requirement for the programme, the student is carrying out his final project. His project is entitled: **Agent-Based Interoperability System in Health Insurance**. This project involves gathering relevant information from various stakeholders in private and public sector. Your institution has been identified as one such stakeholder. We are therefore requesting that your accord the student the necessary assistance. Your assistance will be highly appreciated.

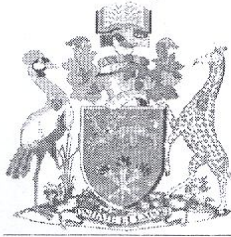
Yours faithfully,

A handwritten signature in black ink, appearing to read 'Christopher A Moturi'.

Christopher A Moturi
Deputy Director
School of Computing & Informatics

School of Computing & Informatics
University of NAIROBI
P. O. Box 30197
NAIROBI

APPENDIX IV: Introduction Letter to National Hospital Insurance Fund (NHIF)



UNIVERSITY OF NAIROBI

SCHOOL OF COMPUTING AND INFORMATICS

Telephone: 4447870/4444919/4446544

Telefax: 4447870

Email: moturi@uonbi.ac.ke

P. O. Box 30197

Nairobi

Kenya

10 April 2013

Manager
HR&D
NHIF

Dear Sir,

LIECH JAMES GOR P58/63063/2011
MSC RESEARCH PROJECT

The above named is bona fide student in the MSc in Computer Science of this University. As part of the requirement for the programme, the student is carrying out his final project. His project is entitled: **Agent-Based Interoperability System in Health Insurance**. This project involves gathering relevant information from various stakeholders in private and public sector. Your institution has been identified as one such stakeholder. We are therefore requesting that you accord the student the necessary assistance. Your assistance will be highly appreciated.

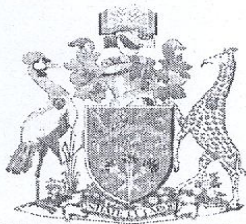
Yours faithfully,

Christopher A Moturi
Deputy Director
School of Computing & Informatics

See HR
Int for file.
[Signature]
07-11/23
151329/3

APPENDIX V: Introduction Letter to Jubilee Insurance

CONTACT CALL: 012
075
Jliech2001



UNIVERSITY OF NAIROBI
SCHOOL OF COMPUTING AND INFORMATICS

Telephone: 4447870/4444919/4446544
Telefax: 4447870
Email: moturi@uonbi.ac.ke

P. O. Box 30197
Nairobi
Kenya

10 April 2013

Jubilee Insurance

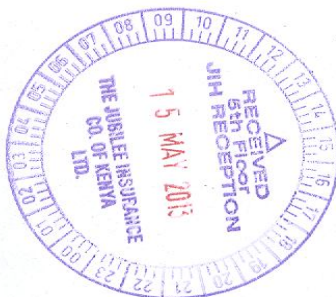
LIECH JAMES GOR P58/63063/2011
MSC RESEARCH PROJECT

The above named is bona fide student in the MSc in Computer Science of this University. As part of the requirement for the programme, the student is carrying out his final project. His project is entitled: **Agent-Based Interoperability System in Health Insurance**. This project involves gathering relevant information from various stakeholders in private and public sector. Your institution has been identified as one such stakeholder. We are therefore requesting that you accord the student the necessary assistance. Your assistance will be highly appreciated.

Yours faithfully,

Christopher A Moturi
Deputy Director
School of Computing & Informatics

School of Computing & Informatics
University of NAIROBI
P. O. Box 30197
NAIROBI



APPENDIX VI: Introduction Letter to UAP Insurance

STUDENT NAME: LIECH JAMES
CONTACT CELL: 0733 126609
0722 635067
0753 270 581
jliech2001@yahoo.com



UNIVERSITY OF NAIROBI
SCHOOL OF COMPUTING AND INFORMATICS

Telephone: 4447870/4444919/4446544
Telefax: 4447870
Email: moturi@uonbi.ac.ke

P. O. Box 30197
Nairobi
Kenya

10 April 2013

UAP Insurance

LIECH JAMES GOR P58/63063/2011
MSC RESEARCH PROJECT

The above named is bona fide student in the MSc in Computer Science of this University. As part of the requirement for the programme, the student is carrying out his final project. His project is entitled: **Agent-Based Interoperability System in Health Insurance**. This project involves gathering relevant information from various stakeholders in private and public sector. Your institution has been identified as one such stakeholder. We are therefore requesting that your accord the student the necessary assistance. Your assistance will be highly appreciated.

Yours faithfully,

Christopher A Moturi
Deputy Director
School of Computing & Informatics



APPENDIX VII: Authority Letter from NHIF to conduct the study



HF/S/1600/129

27th May 2013

James G. Liech - 1600

Thro'

Branch Manager - Ruaraka

Dear Sir,

RE: REQUEST FOR CARRY OUT RESEARCH

We refer to your letter dated 10th April 2013, in which you requested Management to grant you permission to undertake a research on the **Agent-based Interoperability System in Health Insurance**.

We are pleased to inform you that your request has been granted and you will be able to undertake the said research for one (1) month with effect from the date of this letter.

You are advised to report to the Manager HR&D before embarking on the exercise. Upon completion you are requested to submit a copy of your research project report to the HR&D Manager..

If you need more assistance do not hesitate to contact the undersigned and thank you for choosing our organization as your mentor.

J.K. Tonui
for

**J.K. TONU
MANAGER HR&D**

National Hospital Insurance Fund Headquarters, Ragati Road P.O. Box 30443 -00100 Nairobi, Kenya
Tel: (020) - 2723255/6,2723246,2714793/94 Fax: 2714806 E-mail:info@nhif.or.ke Website: www.nhif.or.ke

Appendix VIII: Authority Letter from IRA



INSURANCE REGULATORY AUTHORITY

IRA/RS/1041/2013

19th April 2013

The Deputy Director
School of Computing & Informatics
University of Nairobi

Attn: Liech James Gor P58/63063/2011

RE: REQUEST TO CARRY OUT RESEARCH STUDY

We refer to your letter dated 10th April 2013 requesting the Insurance Regulatory Authority to grant you permission to undertake a research study in our current system for integration to various health insurance firms in Kenya.

In liaison with our Technical department, we are pleased to inform you that the request has been honored and you are requested to visit our offices in Nairobi upper hill to be able to undertake the said research study for a period not exceeding one month with effect from 1st May 2013.

You're also encouraged to report to Mr. Benard Gitangi -Technical department before and during this study.

NB: Upon completion of the study, please submit a copy of the findings to IRA

Thank you

A handwritten signature in black ink is written over the IRA logo. The signature appears to be "James Gor". The logo is the same as the one at the top of the page, with the letters "IRA" printed below it.


HR Division-IRA

APPENDIX VIII: Hospital Claim Form P1

Ref No: 2415122. 1017.

D-3280024
B-361263

Pay OK
03/13



NATIONAL HOSPITAL INSURANCE FUND
 P.O. BOX 30443 00100, NAIROBI
 TEL: 020 2723255/6
 HOSPITAL CLAIM FORM
 email: info@nhif.or.ke
 website: www.nhif.or.ke

SERIAL NO.....

NHIF8
(Revised 2009)

HOSPITAL NAME: JAMU KIPAWA MEDICAL CENTRE CLAIM NO. 2013040 222

HOSPITAL ADDRESS: 54012-00200 Contributor's NHIF No. 0338219

HOSPITAL CODE: 8000949 Contributor's I/D NO. 9801571

PART I: PATIENT'S PARTICULARS

1 Full Names: WELIMSTONE PETER OSAULO

2 Date of Birth: 1967

3 Patient/Contributor's Tel No: 072271107

4 Payment Receipt No:

5 Patient's relationship to contributor: (Self/Spouse/Child)

6 If Child/Dependant, School/College Attended:

Town:

CLAIMS RECEIVED
26 APR 2013
NHIF
TRANSACT SERVICE

PART II: HOSPITALS PARTICULARS

1 Patient's Date of Admission (DOA): 12th April 13

2 Patient's Date of Discharge (DOD): 15/4/13

3 In patient No (IP/NO): 230913 Bed/Cot No. 8

4

Rebate Kshs	No of Days	Invoiced Total Bill Kshs	Amount Payable Kshs
1000	3	5100	3000

PART III: MEMBERS DECLARATION

I hereby certify that I have produced to the hospital Authority my NHIF contribution card No 0338219 duly paid up to date and that the particulars described in part I and II above are correct. I hereby authorize NHIF to reimburse/pay the hospital as stipulated by the NHIF rebate and further give consent to NHIF medical personnel to have unlimited access to my hospital records

Member's Signature [Signature] Date 15/4/13

PART IV: HOSPITAL DECLARATION

I certify that I have inspected card member No 0338219 Valid Receipt No/Certificate No (CCP No)
Please arrange to pay the hospital the sum of KShs. 3000 being the approval rebate of Kshs. 1000 for 3 Days

Name of Authorized official Christine

Designation Nurse Signature [Signature] Date 15/4/13

Official Hospital Rubber Stamp

JAHMII (KIPAWA) MEDICAL CENTRE
P.O. BOX 54012-00200
NAIROBI
TEL: 0770272614

PART V: Attach copy of Discharge summary (with hospital seal)

Disease Code(ICD 10) J42

PART VI: FOR OFFICIAL USE

I hereby confirm that the claim has met all the requirements and promise to ensure confidentiality of the patient's medical records accessed by me.

Quality Assurance Officer (a) Full Name B. Kasandi

(b) Signature [Signature] Date 15/4/13

Compliance Officer (a) Full Name

(b) Signature Date

Branch Manager (a) Full Name

(b) Signature Date

Appendix X: General Claim Form

RKA/19/148/13

NHIF 3 (REVISED 2008)



NATIONAL HOSPITAL INSURANCE FUND

• NHIF Building, Ragati Road • P.O. Box 30443 - 00100, Nairobi, Kenya.
 • Office: +254 020 272 3255/56 • Mobile: +254 (0) 724 566 581, 020 279 xxxx (Extension)
 • Fax: +254-020-272 5752
 Email: info@nhif.or.ke Website: www.nhif.or.ke

GENERAL CLAIM FORM

PA Confirmed
 12/4/13

Part I: Contributor's Particulars

1. Surname/Main name ESENDI JOB MWAMBUKI
2. Other names JOB MWAMBUKI
3. ID No. 5304840 NHIF contribution No. 3649194
4. Mobile No. 0723383515
5. Date when first contribution was due
6. Postal address Box 49506 NAIROBI
7. Name of Employer O.O.D (POLICE DEPT.)
8. Address of Employer BOX 30083 NAIROBI
9. Certificate of contributions paid serial No.

Part II: Patient's Particulars

1. Surname/Main name MUTHARA
2. Other names TERESIA WANJA
3. Date of birth 1980
4. ID No. where patient is self or spouse 22489223
5. Postal address of the patient Box 49506 NAIROBI
6. Patient's relationship to contributor WIFE
 (State whether self, wife/husband or child).



Part III: Hospital's Particulars

1. Hospital's full name Rumani MATERNITY HOSPITAL
2. Hospital's address 42849
3. NHIF approved rate Ksh. 2000 per day.
4. Patient's date of admission 19/11/2011
5. Patient's date of discharge 22/11/2011
6. Nature of treatment Maternity
 (State whether medical, surgical, maternity or therapy)
7. Receipt No. hospital's statement No.



I certify that my contribution to the National Hospital Insurance Fund have been paid to date and I attach my card in evidence. I also certify that to the best of my knowledge and belief that hospitalization in this case does not result from injury or illness in respect of which claim for compensation or damages has been or will be lodged under the Workmen's Compensation Act (Cap. 236).

Date 02/04/2013

E. M. Tende
 Signature

Appendix XI: Code Samples

IRA Platform

```
<%--
  Document   : home.jsp
  Created on : Jun 11, 2013, 7:48:23 AM
  Author    : liech
--%>

<%@page import="java.util.List"%>
<%@page import="com.research.mls.objects.PlatformConnection"%>
<%@page import="com.research.mls.objects.ModelManager"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <% if (session.getAttribute("username") == null) {
    response.sendRedirect("/index.jsp");
  }%>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Home</title>
    <link rel="stylesheet" type="text/css" href="/css/css.css" />
    <link rel="stylesheet" href="wsig.css" />
    <%@include file="no_cache.jspf" %>
  </head>
  <body>
    
    <%@include file="menu.jspf" %>
    <h2>Registered Health Insurance Providers</h2>
    <div class="content">
      <% ModelManager manager = new ModelManager();
      List<PlatformConnection> providers = manager.findEntities(PlatformConnection.class);
      for (PlatformConnection ip : providers) {
        %><div class="bordered"><div class="float_left float"> Logo"/><%
        %><h2><a href="/platform/view.jsp?id=<%=ip.getId()%>"><%
out.print(ip.getName());%></a></h2></div>
        <div class="float_right float">
          <span>Name: <%=ip.getName()%></span><br/>
          <span>State: <%=ip.getActive() > 0 ? "ACTIVE" : "INACTIVE"%></span><br/>
          <span><a href="/platform/edit.jsp?id=<%=ip.getId()%>"> Edit </a></span><br/>
          <span><a href="/platform/view.jsp?id=<%=ip.getId()%>"> View </a></span><br/>
          <span>Website: <a
href="<%=ip.getWebsiteURL()%>"><%=ip.getWebsiteURL()%></a></span>
        </div>
      </div><%          }
        %>
      </div>
    </body>
  </html>
<%--
  Document   : search
```

Created on : Jun 11, 2013, 9:42:43 AM

Author : liech

```
--%>
<%@page import="com.google.gson.stream.JsonReader"%>
<%@page import="java.util.LinkedList"%>
<%@page import="com.research.mls.external.Condition"%>
<%@page import="java.util.HashMap"%>
<%@page import="java.util.List"%>
<%@page import="java.util.ArrayList"%>
<%@page import="com.google.gson.Gson"%>
<%@page import="com.research.mls.external.PolicyHolder"%>
<%@page import="org.apache.commons.lang3.StringEscapeUtils"%>
<%@page import="com.tilab.wsig.soap.SoapClient"%>
<%@page import="com.research.mls.onto.MAHIPOntology"%>
<%@page import="com.research.mls.objects.Constants"%>
<%@page import="com.tilab.wsig.WSIGConfiguration"%>
<%@page import="com.research.mls.external.StringWrapper"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Policy Holder Search</title>
    <link rel="stylesheet" type="text/css" href="/css/css.css" />
    <link rel="stylesheet" href="wsig.css" />
    <%@include file="no_cache.jspf" %>
  </head>
  <body>
    <% boolean ok = true;
    String SOAPResponse = null;
    if (session.getAttribute("username") == null) {
      response.sendRedirect("/index.jsp");
      ok = false;
    }
    String key = request.getParameter("key");
    if (ok && key != null) {
      WSIGConfiguration wsigConfig = (WSIGConfiguration) application
        .getAttribute("WSIGConfiguration");
      // Get parameters
      String SOAPUrl = null;
      if (SOAPUrl == null) {
        SOAPUrl = wsigConfig.getWsigUri();
      }
      StringWrapper lr = new StringWrapper();
      lr.setMessage(key);
      String SOAPRequest = lr.asXML(Constants.SERVER_AGENT_DATASEARCH_AGENT,
        MAHIPOntology.PLATFORM_DATA_SERVICE);
      //System.out.println(SOAPRequest);
      if (SOAPRequest != null && !"".equals(SOAPRequest)) {
        SOAPResponse = SoapClient.sendStringMessage(SOAPUrl,
          SOAPRequest);
      }
    }
    %>
```



```

try {
    SOAPResponse = StringEscapeUtils.unescapeXml(SOAPResponse);
    SOAPResponse = SOAPResponse.split(" xmlns=\"\">", 2)[1];
    SOAPResponse = SOAPResponse.split("</", 2)[0];
//    SOAPResponse = '{' + SOAPResponse + '}';
    System.out.println("=====> " + SOAPResponse);

} catch (Exception e) {
    System.out.println("error: " + SOAPResponse);
    e.printStackTrace();
}
}
}
%>
<div class="content">

<%@include file="menu.jspf" %>
<h2>Search Policy Holders</h2>
<form method="post">
    <input type="search" name="key" required="" <%if (key != null) {
        out.print("value=" + key + "");
    }%>/><input type="submit" value="Search"/>
</form>

<%
    HashMap<PolicyHolder, List<Condition>> map = new HashMap<PolicyHolder,
List<Condition>>());
    if (SOAPResponse != null) {
        try {
            String[] agentResults = SOAPResponse.split(StringWrapper.SEPARATOR_HIGHER);
            for (String agentResult : agentResults) {
                try {
                    String[] split = agentResult.split(StringWrapper.SEPARATOR_LOWER, 2);
                    String agentName = split[0];
                    String placeholdersString = split[1];
                    System.out.println("agentName = " + agentName);
//                    System.out.println("PlaceholderString = " + placeholdersString);
                    String[] ps = placeholdersString.split(StringWrapper.SEPARATOR);
                    Gson gson = new Gson();
                    for (String p : ps) {
                        try {
                            PolicyHolder ph = gson.fromJson(p, PolicyHolder.class);
//                            System.out.println(ph);
                            ph.setPolicy(ph.getPolicy() + "[" + agentName + "]");
                            String conditions[] = ph.getConditions() == null ? new String[0] :
ph.getConditions().split("\\};");
                            List<String> conds = new ArrayList<String>();
                            if (conditions == null || conditions.length <= 1) {
                                for (String c : conditions) {
                                    conds.add(c + "");
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

```

```

    }
    } else {
        for (int i = 0; i < conditions.length; i++) {
            if (i == 0) {
                conds.add(conditions[i] + "");
            } else if (i + 1 == conditions.length) {
                conds.add(conditions[i] + ');
            } else {
                conds.add(conditions[i] + "");
            }
        }
    }
    List<Condition> css = new LinkedList<Condition>();
    for (String ss : conds) {
        System.out.println(ss);
        try {
            css.add(gson.fromJson("" + ss, Condition.class));
        } catch (Exception es) {
            System.out.println(es.getMessage());
//            es.printStackTrace();
        }
    }
    map.put(ph, css);
} catch (Exception eee) {
    eee.printStackTrace();
}
} catch (Exception ee) {
}
}
} catch (Exception e) {
}
}
%>

<%
    int i = 1;
    if (!map.isEmpty()) {
%>
<table ><tr><th ><%    out.print(i++);%></th>
    <th>Full Names</th>
    <th>Citizenship</th>
    <th>Date of Birth</th>
    <th>Gender</th>
    <th>National Identification</th>
    <th>Relation To Contributor</th>
    <th>Policy</th>
    <th>Ailments Suffered</th></tr>
<%
    try {

```

```

        for (PolicyHolder ph : map.keySet()) {
            if (ph == null || ph.getFullNames() == null || ph.getFullNames().equals("null")) {
                continue;
            }
        }
    %>
<tr>
    <td ><% out.print(i++);%></td>
    <td><% out.print(ph.getFullNames());%></td>
    <td><% out.print(ph.getCitizenship());%></td>
    <td><% out.print(ph.getDateOfBirth());%></td>
    <td><% out.print(ph.getSex());%></td>
    <td><% out.print(ph.getNationalId());%></td>
    <td><% out.print(ph.getRelationToContributor());%></td>
    <td><% out.print(ph.getPolicy());%></td>
    <td><% List<Condition> kn = map.get(ph);
        int count = 1;
        if (kn != null && !kn.isEmpty()) {
            %><table style="float: right; width: 100%"> <tr>
                <th></th>
                <th>Condition</th>
                <th>Admitted</th>
                <th>Discharged</th>
                <th>Treatment</th>
            </tr><%
                int index = 1;
                for (Condition cns : kn) {

                    try {
            %>
            <tr>
                <td><% out.print(index++);%></td>
                <td><% out.print(cns.getName());%></td>
                <td><% out.print(cns.getDateAdmitted() == null ? "N/A" :
cns.getDateAdmitted());%></td>
                <td><% out.print(cns.getDateDischarged() == null ? "N/A" :
cns.getDateDischarged());%></td>
                <td><textarea readonly=""><% out.print(cns.getTreatment());%></textarea></td>
            </tr><%
                    } catch (Exception exceptin) {
                    }
                    count++;
                    if (count > 5) {
                        break;
                    }
                }
            %></table><%
                }
            %></td>
        </tr>
    <%
        }
    } catch (Exception ex) {

```

```

    }
    %>
</table>
<%
}
%>
</div>
</body>
</html>

```

Sample Coding

RegistrationAgent.java

```

package com.research.mls.agents;
import com.google.gson.Gson;
import com.research.mls.db.connection.DatabaseConnection;
import com.research.mls.external.PlatformDatabaseConnection;
import com.research.mls.external.SQLiteConnection;
import com.research.mls.objects.Constants;
import com.research.mls.objects.LoginRequest;
import com.research.mls.objects.PlatformConnection;
import com.research.mls.objects.SystemUser;
import com.research.mls.onto.objects.misc.AbstractAgent;
import jade.content.AgentAction;
import jade.content.lang.Codec;
import jade.content.onto.OntologyException;
import jade.content.onto.UngroundedException;
import jade.content.onto.basic.Action;
import jade.core.AID;
import jade.core.Agent;
import jade.core.behaviours.CyclicBehaviour;
import jade.core.behaviours.TickerBehaviour;
import jade.lang.acl.ACLMessage;
import jade.lang.acl.MessageTemplate;
import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author liech
 */
public class RegistrationAgent extends AbstractAgent {

    @Override
    protected void setup() {
        register("registration");
        addBehaviour(new RegistrationRequests());
        addBehaviour(new RegistrationResponses());
    }

    private class RegistrationRequests extends CyclicBehaviour {

        MessageTemplate template =
MessageTemplate.MatchPerformative(ACLMessage.REQUEST);

        @Override

```

```

public void action() {
    ACLMessage msg = receive(template);
    Action actExpr = null;
    PlatformConnection su = null;
    if (msg != null) {
        try {
            actExpr = (Action) myAgent.getContentManager()
                .extractContent(msg);
            AgentAction action = (AgentAction)
actExpr.getAction();
            if (action instanceof PlatformConnection) {
                su = (PlatformConnection) action;
                su.setWsigParams(findWSIGParams(msg));
                actExpr.setAction(su);
                PlatformDatabaseConnection databaseConnection;
                if (su.getDbDriver().matches("(?i).*mysql.*"))
{
                    databaseConnection = new
DatabaseConnection(su);
                } else if
(su.getDbDriver().matches("(?i).*sqli.*")) {
                    databaseConnection = new
SQLiteConnection(su);
                } else {
                    throw new Exception();
                }
                if (!databaseConnection.connect()) {
                    throw new Exception();
                }
                sendMessage(actExpr, ACLMessage.REQUEST, new
AID[] {new AID(Constants.SERVER_AGENT_PERSISTENCE_AGENT,
AID.ISLOCALNAME)});
            }
        } catch (Exception ex) {
            sendResult(actExpr, msg, ACLMessage.INFORM, new
Gson().toJson(su, PlatformConnection.class));
        }
    } else {
        block();
    }
}

private class RegistrationResponses extends CyclicBehaviour {

    MessageTemplate template =
MessageTemplate.MatchPerformative(ACLMessage.INFORM);

    @Override
    public void action() {
        ACLMessage msg = receive(template);
        if (msg != null) {
            try {
                Action actExpr = null;

```



```

<header><h1>Person Information</h1></header>
<?php
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
if (isset($_REQUEST['id'])) {
    require_once 'fct/db.php';
    $result = mysql_query("SELECT * FROM conditions", $db);
    ?>
    <script>
        pageId = <?php echo $_REQUEST['id']; ?>;
        conditions = '<?php
$str = "";
while ($row = mysql_fetch_array($result)) {
    $str.=" . $row['id'] . ":" . $row['name'] . ";";
}
if ($str == "") {
    $str = "-1:None in database";
} else {
    $str = substr($str, 0, strlen($str) - 1);
}
echo "$str";
?>';
    </script>
    <?php
    $obj = new stdClass();
    $result = mysql_query("SELECT * FROM patients where id=" .
$_REQUEST['id'], $db);

    if ($row = mysql_fetch_array($result)) {
        $obj->id = $row['id'];
        $obj->name = $row['full_names'];
        $obj->dob = $row['dob'];
        $obj->sex = $row['sex'];
        $obj->citizenship = $row['citizenship'];
        $obj->relationship = $row['relation_to_contributor'];
        $obj->modified = $row['modified'];
    }

    mysql_close($db);
    ?><div id="content">
        <form>
            <table>
                <thead>
                </thead>
                <tbody>
                    <tr>
                        <th>ID</th>
                        <td><input name="id" value="<?php echo $obj->id; ?>" readonly="true"/></td>
                    </tr>
                    <tr>
                        <th>Name</th>
                        <td><input name="name" value="<?php echo $obj->name; ?>" /></td>
                    </tr>
                    <tr>
                        <th>Date of Birth</th>
                        <td><input name="dob" value="<?php echo $obj->dob; ?>" /></td>

```

```

        </tr>
        <tr>
            <th>Sex</th>
            <td><select name="sex" disabled="">
                <option value="F"<?php echo $obj->sex
== "F" ? " selected='true'" : " "; ?>>Female</option>
                <option value="M"<?php echo $obj->sex
== "M" ? " selected='true'" : " "; ?>>Male</option>
            </select></td>
        </tr>
        <tr>
            <th>Citizenship</th>
            <td><input name="citizenship" value="<?php
echo $obj->citizenship; ?>" /></td>
        </tr>
        <tr>
            <th>Relationship to contributor</th>
            <td><select name="relationship" disabled="">
                <option value="self"<?php echo $obj-
>relationship == "self" ? " selected='true'" : " "; ?>>Self</option>
                <option value="spouse"<?php echo $obj->relationship == "spouse" ? "
selected='true'" : " "; ?>>Spouse</option>
                <option value="child"<?php
echo $obj->relationship == "child" ? " selected='true'" : " ";
?>>Child</option>
            </select></td>
        </tr><tr>
            <th>Last modified</th>
            <td><input name="modified" value="<?php echo
$obj->modified; ?>" readonly="true" /></td>
        </tr>
    </tbody>
</table></form>
<div id="diseases">
    <table id="diseases_table"></table>
    <div id="diseases_table_pagination"></div>
</div>
</div>
<?php
} else {
    die("Person data not found!");
}
?>
</body>
</html>

```

Client Applications UAP ---python
Django settings for uap project.

```

DEBUG = True
TEMPLATE_DEBUG = DEBUG

```

```

ADMINS = (
    # ('Your Name', 'your_email@example.com'),
)

```

```

MANAGERS = ADMINS

```

```

DATABASES = {

```



```
'default': {
    'ENGINE': 'django.db.backends.sqlite3', # Add 'postgresql_psycopg2', 'mysql', 'sqlite3' or 'oracle'.
    'NAME': 'C:\\Users\\tindase\\workspace\\uap\\sqlite.db', # Or path to database file if
using sqlite3.
    'USER': '', # Not used with sqlite3.
    'PASSWORD': '', # Not used with sqlite3.
    'HOST': '', # Set to empty string for localhost. Not used with sqlite3.
    'PORT': '', # Set to empty string for default. Not used with sqlite3.
}
}
```

```
# Local time zone for this installation. Choices can be found here:
# http://en.wikipedia.org/wiki/List\_of\_tz\_zones\_by\_name
# although not all choices may be available on all operating systems.
# In a Windows environment this must be set to your system time zone.
TIME_ZONE = 'Africa/Nairobi'
```

```
# Language code for this installation. All choices can be found here:
# http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = 'en-us'
```

```
SITE_ID = 1
```

```
# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
USE_I18N = True
```

```
# If you set this to False, Django will not format dates, numbers and
# calendars according to the current locale.
USE_L10N = True
```

```
# If you set this to False, Django will not use timezone-aware datetimes.
USE_TZ = True
```

```
# Absolute filesystem path to the directory that will hold user-uploaded files.
```

Appendix XII: Screen shots

Client Application -NHIF

The screenshot displays the NHIF Client Application interface. At the top, there is a navigation bar with the NHIF logo and a 'Home' button, followed by links for 'Policies', 'Guideline', 'Rates', and 'Application'. The main content area is titled 'Patient List' and features a table with the following data:

Id	Full Names	Date of Birth	Sex	Citizenship	Contributor	Modified
7	Lick James	1975-12-12	M	Kenyan	self	2013-06-06 13:27:13
2	Mirka Madio Hiko	1986-09-19	F	Kenyan	child	2013-05-11 19:02:57
6	Swite Pitt	1212-12-12	M	Kenyan	self	2013-04-30 11:26:36
5	Timalai Alox	1898-12-01	M	Kenyan	spouse	2013-04-29 22:59:09

Below the table is a pagination control showing 'Page 1 of 1' and 'View 1 - 4 of 4'. The second section is titled 'Diseases List' and features a table with the following data:

Id	Name	Notes	Modified
7	Swite Pitt	horrible	2013-05-11 19:04:38
6	Laziness	Feels like doing nothing	2013-04-30 11:27:02
5	TB	Tuberculosis bad	2013-04-29 17:07:38
3	Common Cold	coughing	2013-04-29 12:56:17
4	Malaria	blat	2013-04-29 12:56:17

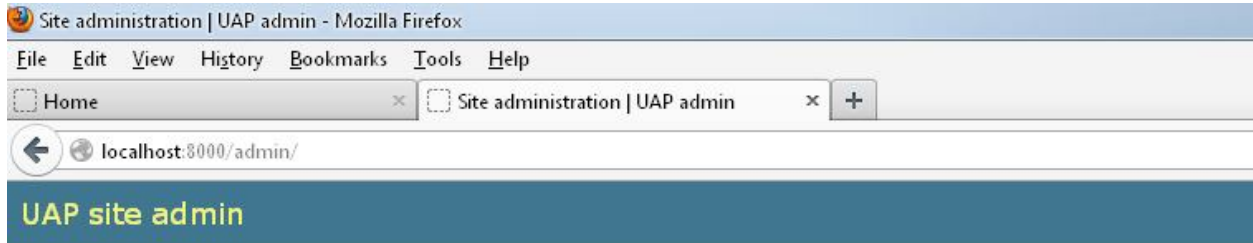
Below the table is a pagination control showing 'Page 1 of 1' and 'View 1 - 5 of 5'. At the bottom of the page, there is a footer with the text 'designed by Lich' on the left and '© 2013 CimpleWave' on the right.

Client Application -UAP

The screenshot shows a browser window with the title 'Log in | UAP admin - Mozilla Firefox'. The address bar displays 'localhost:8000/admin/'. The main content area contains a login form titled 'UAP site admin' with the following fields and buttons:

Username:

Password:

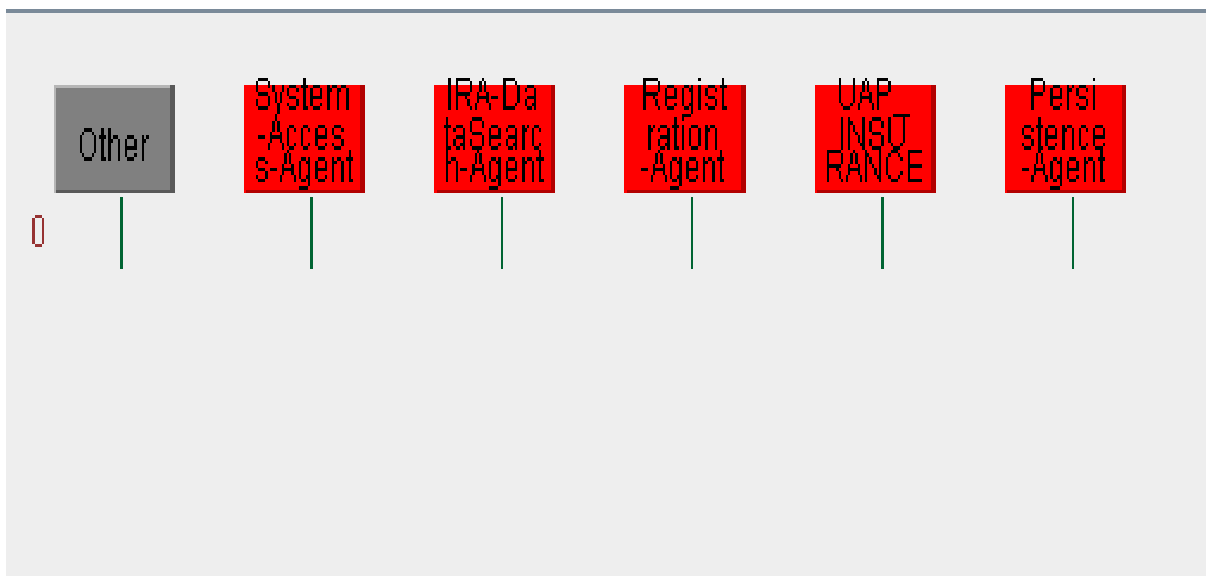


Site administration

Auth	
Groups	+ Add ✎ Change
Users	+ Add ✎ Change
Records	
Conditions/Diagnosis	+ Add ✎ Change
Persons	+ Add ✎ Change
Policies Offered	+ Add ✎ Change
Sites	
Sites	+ Add ✎ Change

Recent Actions
My Actions
+ Prof. Kibuye Paul Imende John Person
✎ Kim John Stamu boy Person
+ Liech James Gor Person
✎ Muturi Kioko Amsoni Person
✎ Kim John Stamu boy Person
+ Pneumonia Condition/Diagnosis
+ Inpatient Policy
✎ Kim John Stamu boy Person
+ Kim Stamu Person
+ Default Policy

IRA MAS



IRA SYSTEM



INSURANCE REGULATORY AUTHORITY
IRA

Login to continue

Username	<input type="text" value="liech"/>
Password	<input type="password" value="••••"/>
	<input type="button" value="Login"/>



INSURANCE REGULATORY AUTHORITY
IRA

[Home](#) [Search Policy Holders](#) [Add Provider](#) [Add User](#) [System information](#) [Logout](#)

Registered Health Insurance Providers

 <p>UAP <i>Better. Simple. Life.</i></p>	<p>Name: UAP INSURANCE State: ACTIVE Edit View Website: http://localhost:8000/admin/records</p>
--	---
