Cellular Network Fault Prediction Using Mobile Intelligent Agent Technology

Okuthe P. Kogeda and Johnson I. Agbinya++

Department of Computer Science (Center of Excellence for IP and Internet Computing)
University of the Western Cape, Private Bag X17, Bellville 7535, Republic of South Africa.
++Centre for Real-Time Information Networks (CRIN), Faculty of Engineering,
University of Technology, Sydney, NSW 2007, Australia.
Email: pkogeda@uwc.ac.za, agbinya@eng.uts.edu.au[++]

Abstract –**Proactive cellular network fault automation models using mobile intelligent agent are presented in this paper. Cellular networks are uncertain and dynamic in their behaviours and therefore we use different artificial intelligent techniques to develop platform independent, autonomous, reasoning, scalable and robust agent that can report on any unforeseen anomaly within the cellular network service provider. The specific design and implementation is done using Java Agent DEvelopment Framework (JADE) [17]. We set up a simple wireless network consisting of four devices for our experiments, which lasted seven days. The partial results obtained from the experiments conducted are presented and discussed in this paper.**

Index Terms: **Mobile Intelligent Agent, JADE software agents, Fault Prediction, Bayesian Network, Cellular networks, architecture and services.**

I. INTRODUCTION

Every cellular network service provider would want to have an edge over the others by offering high quality, reliable and affordable services to its customers as and when required. This objective has become illusive with the complication of the services to be offered, technological development, and increasing number of subscribers. This requires that the operation of the cellular network be at its best all the times to keep the subscribers happy but also to retain them and attract new ones. This can happen with proper maintenance of the network itself.

Cellular network faults prediction is an approach for avoiding the catastrophic failures that may cause network blackout. This paper explores the application of Mobile Intelligent Agents (MIA) in monitoring the network elements for any potential failure of the core objects of the network to be avoided. However, our main concern is the prediction of possible cellular network faults and cases (or scenarios) extracted from certain parameters' correlation, which in any other case could not be noticed by human operators. These could be solved using an advanced automated solution. This paper proposes and discusses the development of a MIA system for computer-aided analysis, simulation and diagnosis based on mobile intelligent software agents [1]. We proposed a framework that utilizes different Artificial Intelligent (AI) techniques and probabilistic methods [2][3][4].

Neural networks, fuzzy logic, genetic algorithms, among others, are some of the established artificial intelligent techniques used into software agents during the mid 1990's [8]. In this work we combine Bayesian networks model with mobile intelligent agents for proactive fault prediction in cellular network service providers.

In our previous work [5], as the ongoing Modelling of Reliable Service Based Operations Support System (MORSBOSS) project we introduced the use of intelligent agents in automating the cellular network faults prediction. We explored how Bayesian Network model could be incorporated into the agent engine to help in cellular network faults prediction.

The main objectives of using Bayesian Network model is that the cellular network faults can be automatically detected based on a similar fault occurrence that the system has experienced before. The information about the previous fault occurrence can be retrieved from the database where it is stored. This information shows the causal relation between network elements, network faults and services. It also shows the belief or likelihood of a fault at a particular network element. Fault prediction is therefore based on the historical memory of the system about known faults.

This paper is organised as follows. In section II, we give a brief overview of Mobile Intelligent Agent (MIA) and related work. We present the reasons for choosing MIA in section III. In section IV, we present cellular network faults prediction models as engine of the MIA. We present the MIA architecture in section V. In section VI, we provide the preliminary results of our experiments and then we draw conclusion in the subsequent section.

II. MOBILE INTELLIGENT AGENT

Intelligent Agent (IA) is defined, as "software that assists people and acts on their behalf. Intelligent agents work by allowing people to delegate work that they could have done, to the software agent. Agents can, just as assistants can, automate repetitive tasks, remember things you forgot, intelligently summarize complex data, learn from you, and even make recommendations to you" [6]. However, we define IA as an autonomous program with the capability of controlling its own actions and decision-making based on prior knowledge, past experience and on its perception of its environment in pursuit of predefined goals [Figure 1]. However, these kinds of agents cannot migrate from one host to another making them undesirable for this work.

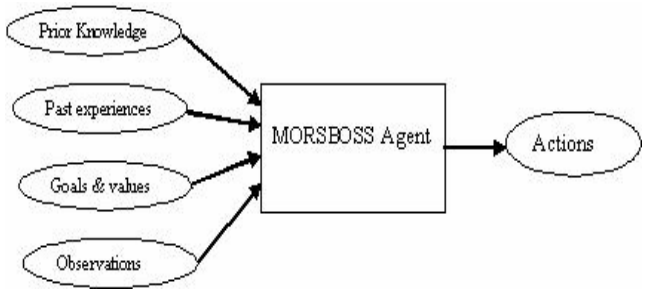

Figure 1: MORSBOSS Intelligent Agent [5]

Mobile Intelligent Agent (MIA) area has ignited interest among researchers in information communication technology (ICT) with applications as diverse as e-commerce, computer games, interface design, personalised information management and management of complex cellular networks. Proactive anomaly detection using distributed intelligent agents is proposed in [7] and faults prediction at the network layer using intelligent agents is proposed in [8]. While their work explored ways of applying intelligent agents and Bayesian Belief Network (BNN) in fault prediction, they fall short of using mobile intelligent agents and relating cellular network faults to services and only explored the network layer. Mobile agents are used by Niki Pissinou et al [11] to automate the fault management in wireless and mobile networks. While they delved on detection of faults and added mobile agents to automate the recovery from faults, they fall short of predicting such faults before they occur. Andrzej Bieszczad et al [12] discussed the potential uses of mobile agents in network management and in [13] MIAs are explored for network management systems. While both papers discuss the application of mobile agents in fault analysis, they fall short of predicting such faults and relating faults to services.

In our previous publications [2][3][4], we explored the use of Bayesian Network model and proposed cellular network fault prediction models. These models will form part of the study and work presented in this paper.

We define Mobile Intelligent Agent (MIA) as a software program that acts on behalf of a user or another program and is able to migrate from network node to network node on a network system under its own control. The MIA chooses when and where it will migrate and may interrupt its own execution and continue elsewhere on the network. The agent returns results and messages in an asynchronous fashion. The intelligence and ability to cooperate features of MIA are exploited in this work.

III. WHY MOBILE INTELLIGENT AGENTS?

Almost every task that can be performed by Mobile Intelligent Agents (MIA) can be done by stationary intelligent agents. However, the use of MIA brings certain benefits over other technologies such as stationary intelligent agents, remote objects, etc, including [12][13][14][15][16]:

- Efficiency savings – CPU consumption is limited, because a mobile agent executes only on one node at a time. Other nodes do not run an agent until needed.
- Space savings – Resource consumption is limited, because a mobile agent resides only on one node at a time. In contrast, static multiple servers require duplication of functionality at every location. Mobile agents carry the functionality with them, so it does not have to be duplicated.
- Reduction in network traffic – Code is very often smaller than data that it processes, so the transfer of mobile agents to the sources of data creates less traffic than transferring the data. Remote objects can help in some cases, but they also involve marshalling of parameters, which may be large.
- Asynchronous autonomous interaction – Mobile agents can be delegated to perform certain tasks even if the delegating entity does not remain active.
- Interaction with real-time systems – Installing a mobile agent close to a real-time system may prevent delays caused by network congestion.
- Robustness and fault tolerance – If a distributed system starts to malfunction, then mobile agents can be used to increase availability of certain services in the concerned areas. For example, the density of fault detecting or repairing agents can be increased. Some kind of meta-level management of agents is required to ensure that the agent-based system fulfils its purpose.
- Supports for heterogeneous environments – Mobile agents are separated from the hosts by the mobility framework. If the framework is in place, agents can target any system. The costs of running a Java Virtual Machine (JVM) on a device are affordable.
- On-line extensibility of services – Mobile agents can be used to extend capabilities of applications, for example, providing services. This allows for building systems that are extremely flexible.
- Convenient development paradigm – Creating distributed systems based on mobile agents is relatively easy.
- Easy software upgrades – A mobile agent can be exchanged virtually at will.
- Dynamic adaptation – mobile agents can sense their environment and react autonomously to changes. When a number of mobile agents are assigned to work with a common goal they can distribute themselves among hosts in the network to maintain the optimal configuration for solving the problem. In the case of a mobile agent moving across a number of host nodes, it can adapt its future behaviour according to information that it has already collected and stored in its state.

However, MIA still suffer from certain cost constraints, which include [13][14]: migration and machine load overhead; high costs in speed for Remote Method Invocation (RMI); resource management; standardization and interoperability; and lastly the directory service used by the agents seemed to slow down communication when the number of agents increases.

IV. THE MOBILE INTELLIGENT AGENT MODEL

The proposed MORSBOSS mobile intelligent model is designed into three tiers conducive to real-time processing. The first tier is specific network elements, which are within the cellular network environment. The mobile agent monitors the network elements/nodes in this environment. The next tier is the mobile agency. The MORSBOSS agent operates at this particular tier. It mediates between the managed network elements and the data tier. The MORSBOSS agent analyses and then logs the fault alarms into the database. Data is the last tier where generated alarms (evidence of fault occurrence) are stored. The alarms variables are aggregated and computed based on Bayesian network model and then these alarms are combined using a priori information about the relationships between the variables to produce network element alarms, which are the indicators of the network health. Figure 2 shows the proposed architecture.

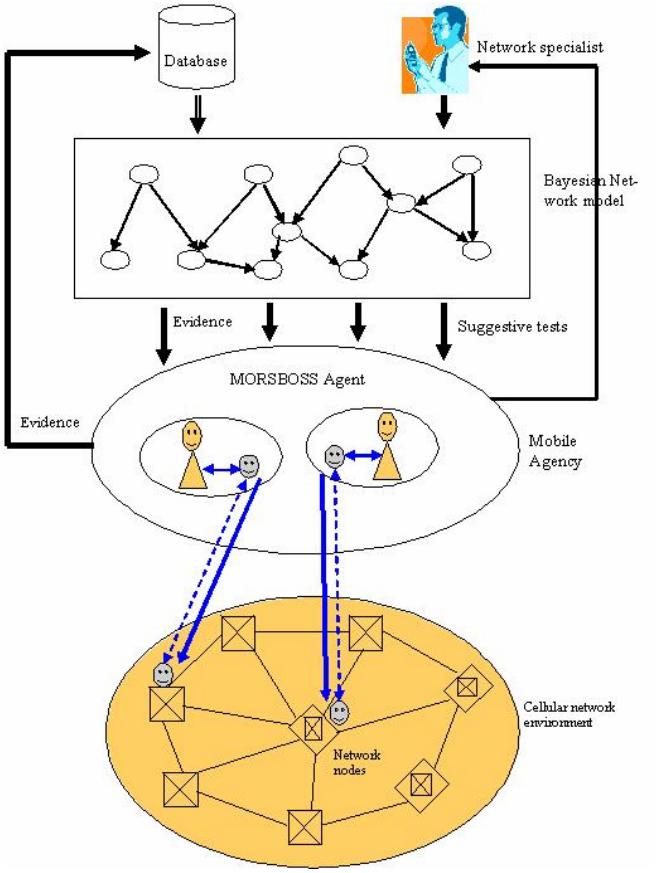

Figure 2: Proposed MORSBOSS Model

The cellular network under study has power (Po), cell (C), transmission (T), and multiplexer (Mux) faults as the network variables to be estimated as shown in details in Figure 3. Each variable has observations, which are stored in the database. The arrows indicate cause and effect of the cellular fault. The structure was chosen from the natural grouping of the network dynamics into four variables mentioned above. The data from different database variables were combined through the probabilistic framework defined by the Bayesian belief network, where the probabilities were estimated from the network variables as in [2][3][4]. These include the conditional probability, for example, assuming multiplexer (Mux) is ok then the only reason it may not perform its functions is when power (Po) fails. This can be calculated by equation (1) and the joint probability distribution using equation (2).

$$p(Mux \mid Po) = \frac{p(Mux) \times p(Po \mid Mux)}{p(Po)} \quad (1)$$

$$\begin{aligned} &\mathrm{p(Po,Mux,C,T)} = \\ &\mathrm{p(Po)} \times \mathrm{p(Mux)} \times \mathrm{p(C \mid Po,Mux)} \times \mathrm{p(T \mid Mux)} \end{aligned} \quad (2)$$

The prediction factor, 'belief' that a variable $X_k \notin \{X_m, ..., X_p\}$ assumes the value x_k is computed using equation (3) when one knows a set of evidences $e = \{X_m = x_m, ..., X_p = x_p\}$, constituted by all the known values of the random variables of Bayesian network, where $\{X_m, ..., X_p\} \subset X = \{X_1, X_2, ..., X_n\}$:

$$p(X_k = x_k \mid e) = \frac{p(X_k = x_k) \times p(e \mid X_k = x_k)}{p(e)} \quad (3)$$

The above equations form part of the MIA engine, which is able to foresee and move to a node likely to be faulty. The proposed model in this paper utilizes different artificial intelligent techniques. The development of the proposed application (software) requires knowledge extraction from the alarms stored in the database (MORSBOSS Database), which must be updated with the new knowledge that might have been discovered from the alarm data and the belief of foreseen fault occurrence.

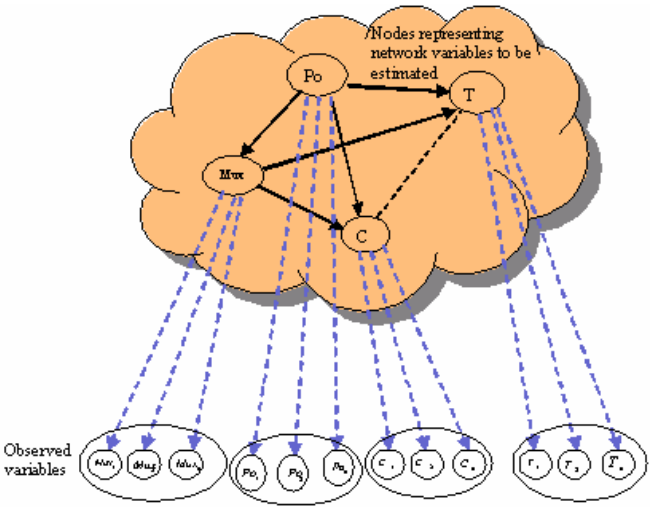

Figure 3: A Bayesian network for fault prediction

The schematic system model is shown in Figure 4. The MORSBOSS database developed in MYSQL stores the variable alarms, which are grouped into four distinct fault variables using probabilistic framework defined by Bayesian belief network. Computation of conditional and joint probabilities based on the information at hand is done within MORSBOSS agent using equations (1) and (2). It then interprets the computed figures, which may lead to evidence of an eminent fault and suggestive tests to be done in order to prevent the eminent failure. It is at this point that MORSBOSS agent sends alarm message to the Engineering assistant agent about the health of the network. This message can be in the form of *excellent*, *good*, *fair* or *critical*. Whenever a critical alarm is sent to the engineering assistant agent who in turn informs selected customers (based on profile) about the foreseen fault, the network is at a high risk of failure. Depending on time to fault, the MORSBOSS agent can decide to inform the selected customers and engineering assistant agent at the same time.

Besides informing the engineering assistant agent, the MORSBOSS Agent moves with speed to the node, which is deemed to become faulty. Since it is a mobile agent, it clones itself in order to cooperate and do the task at hand. In this manner, the agent is able not only to predict but also to be ready to resolve the error; for example, the traffic can be re-routed to other routes avoiding the faulty node. Once a fault occurs then the agent communicates the message (i.e., start time, end time, node id, service id, etc) as evidence, which is sent to the database as shown in Figure 2.

The interpretation of the computed values may show no change, positive change or negative change. The agent will determine this change within a given time window. We took average time to fault as our time window for repeating this

computation in order to determine the node with high frequency failure.

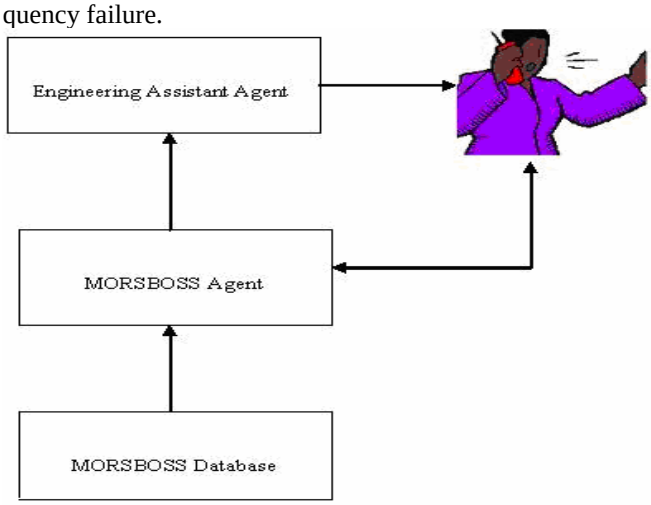

Figure 4: Schematic System Model [5]

MIA operates in an environment, which is cellular network node. The nodes often go into different states of operation. These states can be normal or abnormal. When a state of a node is normal, then such a node is assumed to be operating well without errors and abnormal state implies that the node is faulty. Let us assume that the node (N) may be in finite state set E of discrete states: $N = \{e, e', ...\}$ Where e is normal state and e' is abnormal state. The MIA is assumed to have an array of possible actions available to them, which they can perform to transform the state of the node. The state of the node also dictates which action the MIA will perform. The action of MIA is also dependent on the history of the node. State transformer function [25] is used to represent the effect that MIA's action have on the node:

$$\tau : R^a \rightarrow p(N) \qquad (4)$$

Where *R* is a run of MIA in an environment; *a* actions.
We compute the likelihood value of a fault occurring using Bayesian network model. Using MIA, ***M*** and our modeling techniques, we define the likelihood of an environment being in abnormal state by:

$$M = \begin{cases} p(e_0, e_1, e_2, ...e_n) \\ 0 \qquad\qquad otherwise \end{cases} \qquad (5)$$

Where $e_0 \in N$ is the initial state of the node.
The Utility (U) function of the MIA can be measured on some particular runs (r) using:

$$U(r) \cong \frac{fr}{fd} \qquad (6)$$

Where *fr* is number of faults fixed in *r*; *fd* is number of faults that appeared in *r*.
Let us write $p(r \mid M, E)$ to denote the probability that run *r* occurs when MIA *m* is placed in environment *E*, clearly as:

$$\sum_{r \in R(M,E)} p(r \mid M, E) = 1. \qquad (7)$$

Then the optimal MIA M_{opt} in an environment *E* is defined as the one that maximizes expected utility:

$$M_{opt} = \arg\max_{M \in m} \sum_{r \in R(M,E)} U(r)\, p(r \mid M, E) \qquad (8)$$

V. IMPLEMENTATION ARCHITECTURE

A. Similarities Between MIA Architectures

The mobile agent architectures differ, but almost all of them contain a Mobile Agent (MA) and a Mobile Agent Environment (MAE) [Figure 5]. A naming service and an execution environment comprise mobile agent architecture. A naming service for mobile agents caters for the naming, migrating agents and lookup of the agent execution environments hosting those agents. Reflection and introspection are mechanisms required for the binding and remote communication of mobile agents because of their dynamic nature. A proprietary Agents Communication and Transfer Protocol (ACTP) often support the communication. ACTPs are based on a number of protocols such as Java Remote Method Protocol (JRMP), the Internet Inter-ORB Protocol (IIOP) or sockets relying on TCP or UDP transport.

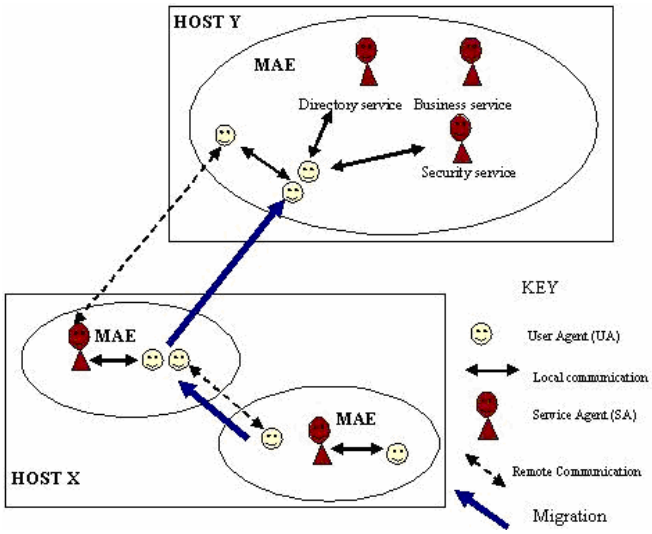

Figure 5: Mobile Agent System

However, the requirement of a project or research, available platform, knowledge of programming language, and support for users are some of the criteria that dictate the choice of a particular architecture.

B. Why JADE Architecture?

A large number of mobile architectures are available today, offering suitable runtime environment for mobile agents as well as high-level programming API supporting mobile agent capabilities. The work presented in this paper was done using JADE-LEAP (Java Agent Development Framework and Lightweight Extensible Agent Platform) [18][23] (mobile agent platform) distributed by Telecom Italia [17][18]. JADE [17] was chosen for a combination of benefits offered such as [26]:

- Simplicity in usage and agent programming,
- Good online community support and documentation,
- Support for the FIPA [19] standards,
- Efficient and tolerant of faulty programming and
- It is an open source (free), etc.

However, it is worth noting that JADE does not provide support for migration between different execution environments as a drawback.

JADE has already been integrated into different major architectures, such as J2EE and .NET allowing JADE to execute multi-platform proactive applications. JADE can take J2EE [24] for servers, J2SE [24] for PCs and desktop computers, Personal Java (Pjava) [22] for wireless mobile devices supporting Pjava (i.e., PDA) and J2ME with CLDC

and MIDP for mobile devices supporting MIDP [21] (i.e., cell phones) as shown in Figure 6.

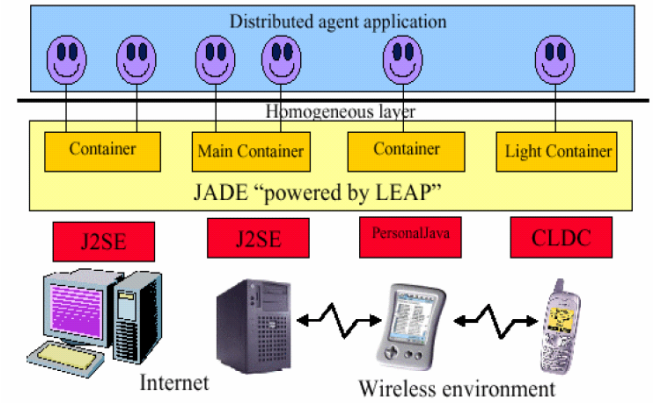

Figure 6: JADE-LEAP agents' architecture, from JADE-LEAP user's guide.

A MIA agent is a JADE-LEAP agent [Figure 6] is a mobile agent with autonomy and high degree of collaboration. The system automatically creates a new Client Agent to act on behalf of the user once a new user logs onto the system. The agent name is unique, and based on the user ID. The Client Agent is able to obtain and maintain the user's personal information and preferences, and to react to various incoming and outgoing messages and requests intended for the user. The Client Agent is automatically removed from the system once the user log outs.

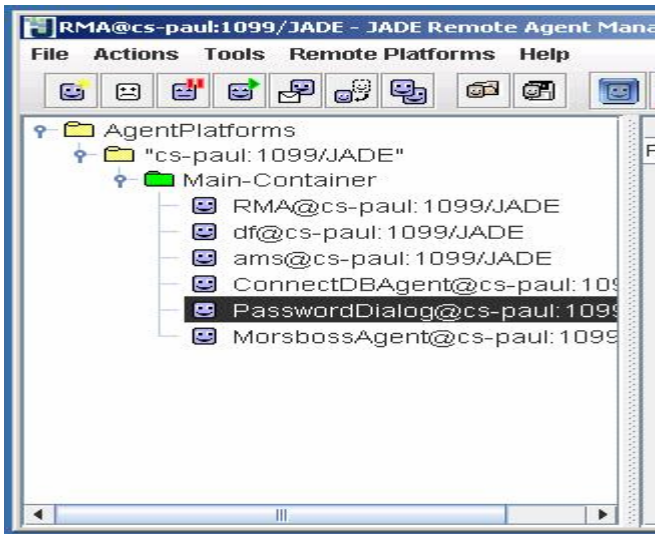

Figure 7: The Agents in Main Container

VI. RESULTS

We set up a simple network of eight devices. 2 iPAQs, a router, 2 desktop PCs, 2 laptops and wireless network access point are the devices that act as nodes in our simple wireless network. We executed our system from one of the PCs acting like server. The MIA resides in the server. The setup and connection of the wireless LAN is shown in Figure 8. We decided to send simple messages, which arrived in time as any normal healthy cellular network service provider.

In our testing of the models, we injected faults to various nodes after a random number of minutes. The injection was done with a view to see if the MIA could respond to not only detecting the faults but also in predicting the faults before they occur. We made 300 runs over a period of two weeks. A run took an average of 30 minutes. During each run the MIA would update the 'belief' after a time window of five minutes, by logging in the faults and computing the new 'belief' of each and every variable. The node status is stored in the database and bears a typical format (for example, faultID: 1 serviceID: 1 faultName: Cell prob: 0.0898438 faultState: VERY UNLIKELY).

The customers consume services, whose performances are affected by the fault. In case of a fault, the MIA may inform the customer of a fault as well as network engineer. The typical message to the customer who consumes the service affected is (Dear Customer, Fault ID: 1 has been created for service ID: 1 MORSBOSS will endeavour to resolve the problem ASAP. Please do not reply this SMS).

By injecting faults at randomly during each run of the MIA system, we measured the success rate in which the system performed. In our experiments of 300 runs, our model could report 235 faults/errors before they occurred, 48 faults/errors after they had occurred, and failed to report the remaining 17 faults/errors, which we injected. This gives us a success rate of about 78% prediction rate.

The results show that the prediction rate increases rapidly at the start with few runs being made but the increase becomes less (stabilizes) as we make many runs. The probability of the network being faulty in each of the runs we made is shown in Figure 9.

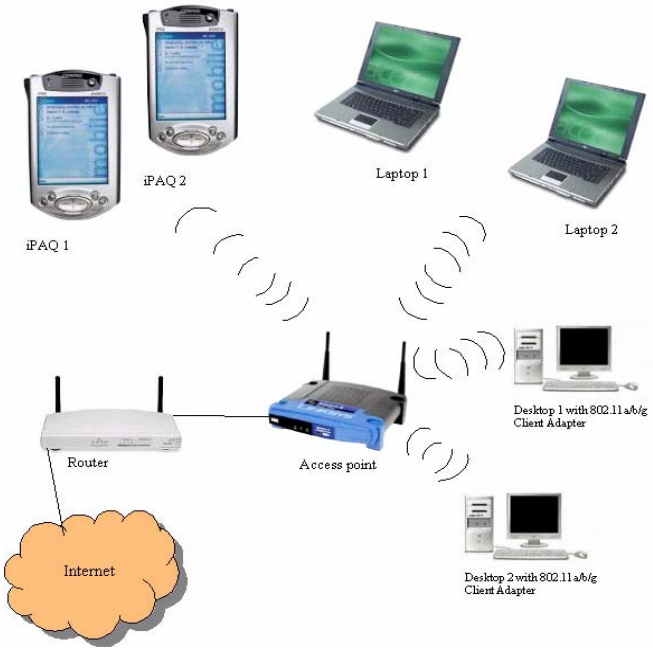

Figure 8: Wireless network for Experiment

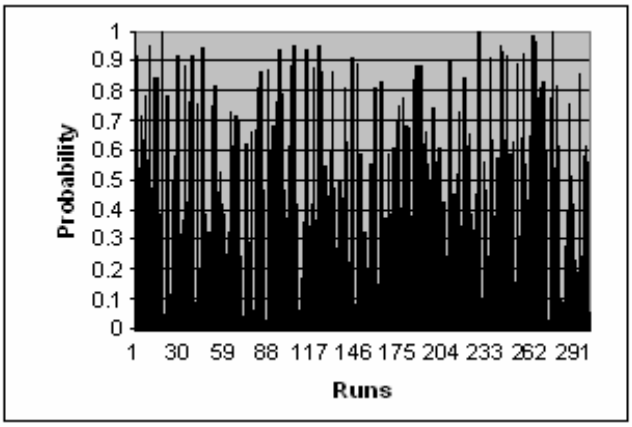

Figure 9: Probability of faults in each run

VII. CONCLUSION AND FUTURE WORK

The model utilizing different artificial intelligent techniques was proposed in this paper. Cellular network fault prediction using mobile intelligent agent technology and Bayesian belief network was presented. The implementation of the system is complete with preliminary results being shown. The MIA shows a success rate of about 78%. Further research will strive to improve on the success rate and incorporate swarm intelligence in furthering the cellular network fault prediction. We also intend to train the MIA covering more nodes and doing more runs if better results can be obtained.

VIII. ACKNOWLEDGEMENT

Okuthe P. Kogeda would like to thank the National Research Foundation, South Africa for the financial support that has made this work possible.

IX. REFERENCES

[1] Wooldridge, M. and N.R. Jennings, "Intelligent Agents: Theory and Practice," Knowledge Engineering Review, 1995. 10: p. 115-152.

[2] Okuthe P. Kogeda, Johnson I. Agbinya and Christian W. Omlin, "Probabilistic Faults Prediction in Cellular Networks", Proceedings of Southern African Telecommunication Networks and Applications Conference (SATNAC), Drakensberg, South Africa, 11th – 14th September 2005.

[3] Okuthe P. Kogeda and Johnson I. Agbinya, "Prediction of Faults in Cellular Networks Using Bayesian Network Model", In the Proceedings of First IEEE International Conference on Wireless Broadband and Ultra Wideband Communications (AUSWIRELESS'06), Sydney, Australia, March 13 – 16, 2006.

[4] Okuthe P. Kogeda, Johnson I Agbinya, and Christian W. Omlin, "A Probabilistic Approach To Faults Prediction in Cellular Networks", In the Proceedings of IEEE 5th International Conference on Networking (ICN'06), Mauritius, April 23 - 28, 2006.

[5] Okuthe P. Kogeda and Johnson I. Agbinya, "Automating Cellular Network Fault Prediction Through Intelligent Agent Technology", In the Proceedings of Southern African Telecommunication Networks and Applications Conference (SATNAC), Cape Town, South Africa, 4th -6th September 2006.

[6] Don Gilbert, Intelligent Agents: The Right Information at the Right Time, 1997.

[7] Marina Thottan and Chuanyi Ji, "Proactive Anomaly Detection Using Distributed Intelligent Agents", IEEE Network, Sept./Oct. 1998.

[8] M. Thottan and C. Ji, "Fault Prediction at the Network Layer using Intelligent Agents",Proc. IEEE/IFIP Integrated Network Management, May 1999, Boston, MA.

[9] B. Chaib-draa, B. Moulin, R. Mandiau, P. Millot, "Trends in Distributed Artificial Intelligence", Artificial Intelligence Review, vol. 6, pp. 35-66, 1995.

[10] David Chess, Colin Harrison, Aaron Kershenbaum, "Mobile Agents: Are They a Good Idea?", IBM Research Report, April 27 1998.

[11] Niki Pissinou, Bhagyavati, Kia Makki, "Mobile Agents to Automate Fault Management in Wireless and Mobile Networks", IPDPS Workshops, Lecture Notes in Computer Science, Vol.1800, pp.1296-1300, Springer, 2000.

[12] Andrzej Bieszczad, Benard Pagurek and Tony White, "Mobile Agents for Network Management", IEEE Communications Surveys and Tutorials, 1(1), 1998.

[13] George Eleftheriou and Alex Galis, "Mobile Intelligent Agents for Network Management Systems", Proceedings London Communication Symposium, 2000.

[14] David Chess, Colin Harrison, Aaron Kershenbaum, "Mobile Agents: Are They a Good Idea?", IBM Research Report, April 27 1998.

[15] Emerson Ferreira de Araujo Lima, Patricia Duarte de Lima Machado, Jorge Cesar Abrantes de Figueiredo and Flavio Ronison Sampaio, "Implementing Mobile Agent Design Patterns in the JADE framework", 2004.

[16] Green, S., Hurst, L., Nangle, B., Cunningham, P., Sommers, F. and Evans, R., *Software Agents: A review*, Technical Report, Department of Computer Science, Trinity College, Deblin, Ireland.

[17] JADE website, JADE "Java Agent DEvelopment Framework –" Home page for the JADE, **http://jade.tilab.com/** , March 2007.

[18] Antonio Moreno, Aida Valls, Alexandre Viejo, "Using JADE-LEAP to implement agents in mobile devices", 2002.

[19] Foundation for Intelligent Physical Agents. Specifications. 2000. Available at **http://www.fipa.org**.

[20] M. Griss and G. Pour, "Accelerating Development with Agent Components", IEEE Computer, pp. 37-43, 2001.

[21] Mobile Information Device Profile, **http://java.sun.com/products/midp**

[22] Pjava, Personal Java. **http://java.sun.com/products/personaljava/pj-emulation.html**

[23] JADE-LEAP, **http://leap.crm-paris.com/**

[24] Sun, **http://www.sun.com**

[25] Fagin, R. et al, "Reasoning about Knowledge", MIT Press, Cambridge, MA, 1995.

[26] Josef Altmann, Franz Gruber, Ludwig Klug, Wolfgang Stockner, Edgar Weippl, "Using Mobile Agents in Real World: A Survey and Evaluation of Agent Platforms", Workshop on Infrastructure for Agents, MAS and Scalable MAS, pp. 33-39, June 2001.

X BIOGRAPHY

Okuthe P. Kogeda received his Master of Computer Application from Dr Babasaheb Ambedkar Marathwada University, India in 1999. He is MCP and MCSD since 2000. He is OCP since 2002. He is currently a PhD student in the Computer Science Department at UWC, South Africa. His research interests include biologically inspired modeling, simulation, Bayesian networks, OSS, Intelligent Agents, Mobile Intelligent Agents and data mining.