Context-Aware Course Reminder: An Application of Context Management Strategy in Mobile Networks

Edem. Williams, Dept of Computer Science, University of Calabar, Nigeria.
edemwilliam@yahoo.com
Giadom Vigale, Department of Computer Science, University of Port Harcourt, Rivers State, Nigeria, email: vigalegiadom@yahoo.com

***Abstract*—Mobile Technology is increasingly being used today in every circle of human endeavor ranging from personal to corporate environment. Little can be achieved in this modern day without the use of a mobile phone. These mobile phones are very important sources of context information as they generate data that can be used to deplete the situation of an entity. They generate information on locations, identities of nearby people objects, and changes to those objects. In this paper, a context aware course reminder was developed using some advance context management strategies already proposed. This application shall among others seek to demonstrate the effectiveness and efficiency of context management strategies in managing context information.**

Keywords -Mobile Technology, Human Endeavor, Corporate, Mobile Phones, Context Information, entity, Course Reminder, Context Management, Strategies. (Keywords)

INTRODUCTION

Context-aware computing is a mobile computing paradigm in which applications can discover and take advantage of contextual information such as user location, time of day, neighboring users and devices, and user activity (Chen and Kotz, 2004). Many researchers have studied this phenomenon and built diverse context-aware applications. Chen and Kotz examined context-aware systems and applications, types of context used and models of context information, systems that support collecting and disseminating context and applications that adapt to changing context

CONTEXT

Schilit et al, (1997), the pioneers of context-aware computing, regard context to be location, identities of nearby people and object, and changes to those objects. They consider where you are, whom you are with, and what resources are nearby to be the important aspects of context. Dey and Abowd's more recent classification of context (Dey and Abowd , 2001) expands the Schilit et al, (1997) definition. They define context as: "…any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and application themselves."

This means that any information that depicts the situation of a user can be entitled context. The temperature, the presence of another person, the nearby devices, the devices a user has at hand and the orientation of the user are examples.

MOBILE TECHNOLOGY CONTEXT

Users of computing devices today are faced with diverse devices (mobile or fixed) featuring diverse interfaces and used in diverse environments. This is a step towards realization of a ubiquitous computing paradigm, or the 'third wave of computing', where specialized devices out number users (Weiser, 1991). However, many important pieces necessary to achieve this ubiquitous computing vision are not yet in place.

Most notably, interaction paradigms with today's devices fail to account for major differences between the static desktop and mobile interaction models. Computing devices are now often used in changing environments, yet do not adapt to those changes very well. Although moving away from the desktop model brings a variety of new situations in which an application may be used, computing devices are seldom aware of their surrounding environments. It has been suggested that enabling devices and applications to automatically adapt to changes in surrounding physical and operational environments can lead to enhancement of user experience. Thus information in the physical and operational environments of mobile devices creates a context for interaction between users and devices (Musumba and Nyongesa, 2013).

CONTEXT AWARE PROGRAMS

These are set of instructions, commands and applications that can adapt and perform tasks on ambient conditions in the physical or the virtual world. In context-aware applications, adaptations are triggered by changes of certain context information. For example, smart applications designed to support meetings may automatically transfer a presentation to a projector as soon as the presenter enters the meeting room. In this case, both the location of the presenter and his/her role in the meeting room are basic pieces of context information used to trigger the transfer of the presentation. Basically, the development of a context-aware application, as in this example, involves the description of the actions to be triggered according to a set of contextual conditions. A same piece of context information may be used for different purposes. The location of the presenter, for example, may also be used by another application to disseminate his availability status for an instant communicator. Moreover, this context information may be provided by different sensors, such as a proximity sensor to identify if the user is inside the classroom and using a microphone connected to voice recognition software to identify specific users in the classroom (Williams and Giadom, 2014).

CONTEXT MANAGEMENT STRATEGIES IN WIRELESS TECHNOLOGY

In Williams and Giadom, (2014), various context management strategies were detailed and analyzed in terms of performance and efficiency. These strategies include:

- Centralized strategy

- Hierarchical strategy
- Distributed strategy
- Hybrid (Hierarchical and Distributed) strategy

A. CENTRALIZED MANAGEMENT STRATEGY

With centralized wireless context management, the wireless controllers that manage access points in each location all connect to a central management console. With centralized context management strategy, all of these can be managed from a single control station or server. Businesses often tend to add equipment and software to networks that can make them more complex; the number of systems to manage is sometimes so large that there is a lack of connections between disparate parts. Centralized network management usually makes user access, data storage, and troubleshooting more convenient.

Managing a network generally includes monitoring performance, but security, balancing of processor loads, and traffic management are usually important as well. A server can be centralized to monitor various operational parameters. It can react in response to particular actions or if certain levels of traffic or processing activity are reached. Operational and security policies can also be set in the system so that centralized network management can be performed efficiently. Specialized software for centralized network management often helps to monitor, analyze, and manage all of the components. It can allow an administrator to set policies, assign network equipment for specific resources, and view the performance of individual components or the entire system on a graphical screen. These data can be applied to developing a model of the network, planning for additional capacity, and forecasting how certain variables will affect performance. Network administration not only involves managing data and system operations. One must also be able to control who has access to what. Passwords and user permissions, as well as event logging, are often part of centralized network management too. On some networked systems, it can be possible to load software from remote locations, especially if there is a connection between data storage and a remote server.

As shown in fig. 1 below, centralized context management is typically useful when many applications running in the system interfere with performance. Services that work with voice, video, and data often face such problems, so administrators can adjust network functions to make things more efficient, like with a communications network, for example. By centralizing operations, administrators can better understand regular network functions and also predict how any changes or upgrades will impact the way things work (Mohsen and Peter, 2009).

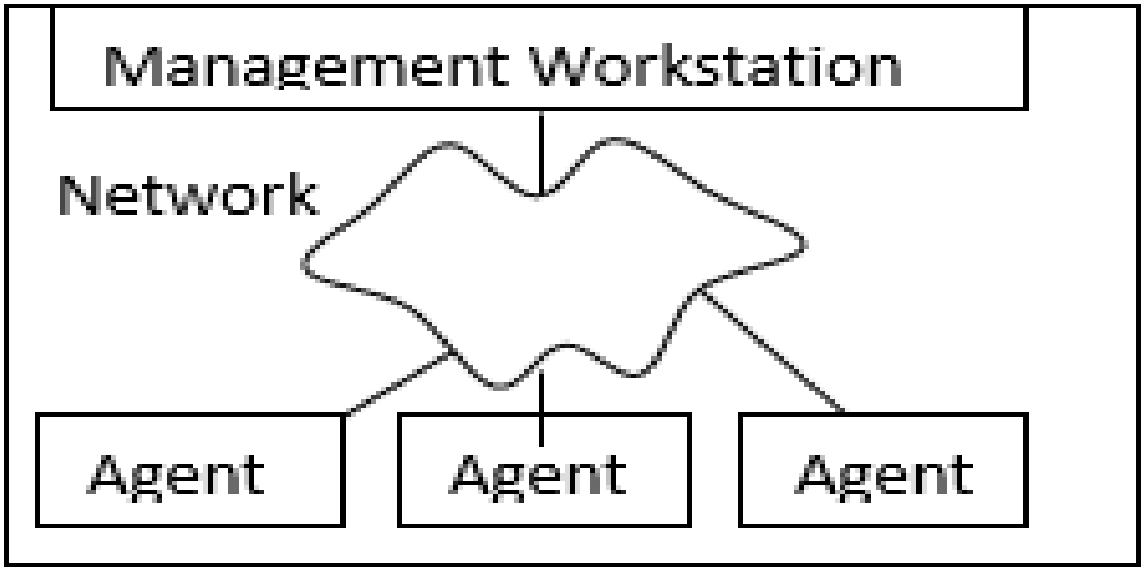

Figure 1: Centralized context management strategy.

B. HIERARCHICAL CONTEXT MANAGEMENT STRATEGY

According to Mohsen and Peter, (2009), large enterprise networks span application, organizational and geographical boundaries. In order to cope sufficiently with the unpredictable growth of the number of network devices, structuring networks in logical hierarchies is being employed as a design and deployment principle. Any of the following or a combination of these partitioning criteria can be used: (a) geographical subdivisions; (b) administrative subdivisions; (c) grouping based on different access privileges and security policies; (d) performance-driven network partitioning between multiple management servers. The design of such networks' management systems must consider that the resultant network segments may be better managed as logical, hierarchically structured domains.

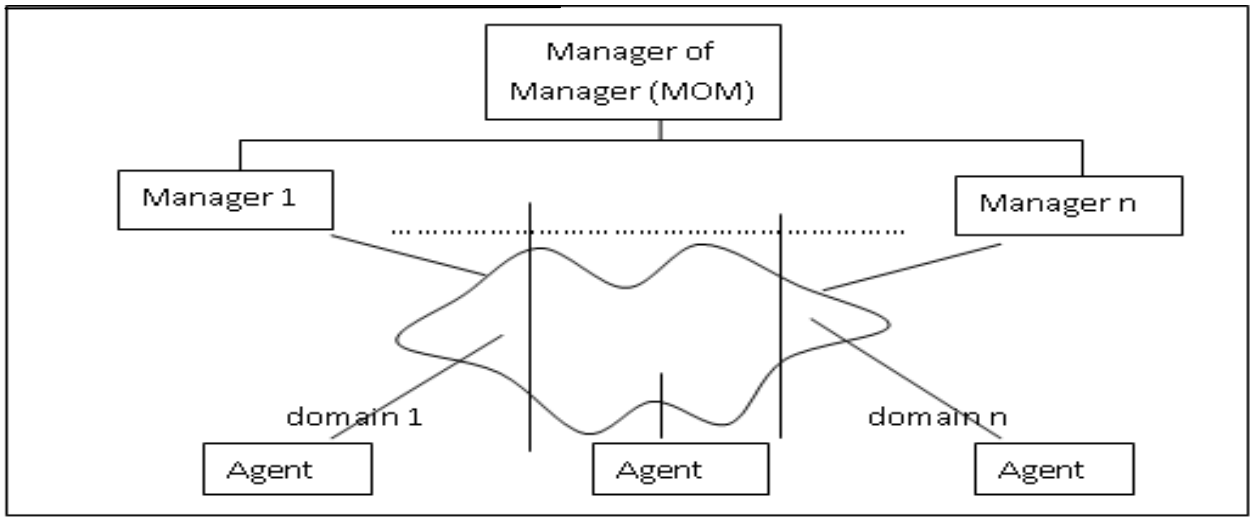

Figure. 2. – Hierarchical context management approach

As shown in figure 2 above, the hierarchical architecture uses the concept of "Manager of Managers" (MOM) and manager per domain paradigm. Each domain manager is only responsible for the management of its domain, and is unaware of other domains. The manager of managers sits at a higher level and request information from domain managers. In this architecture, there is no direct communication between domain managers. This architecture is quite scalable, and by adding another level of MOM a multiple level hierarchy can be achieved.

Hierarchical network management system also uses the concept of SubManager, similar to the manager-to-manager (M2M) protocol described in Simple Network Management Protocol version 2 (SNMPv2). As shown in figure 3, a SubManager is associated with a few agents, and collects the primitive context information from them, performs some calculations, and produces more meaningful values that can be used by a superior manager. This method significantly reduces the amount of management traffics, because only high-level information is sent to the master manager.

Whenever network operators need more, or high-level, information a downloaded Network Management Procedure (NMP) is dynamically assigned to the system. This allows dynamic reconfiguration at run-time, and removes the need to hard-wire everything at compile time. The NMPs are loaded into submanagers, and are stored in two tables: subMgrEntry and subMgrOps. Each procedure is periodically activated and a "Worker" is created which evaluates the procedure and stores the result into subMgrValue table. This table is read by the manager.

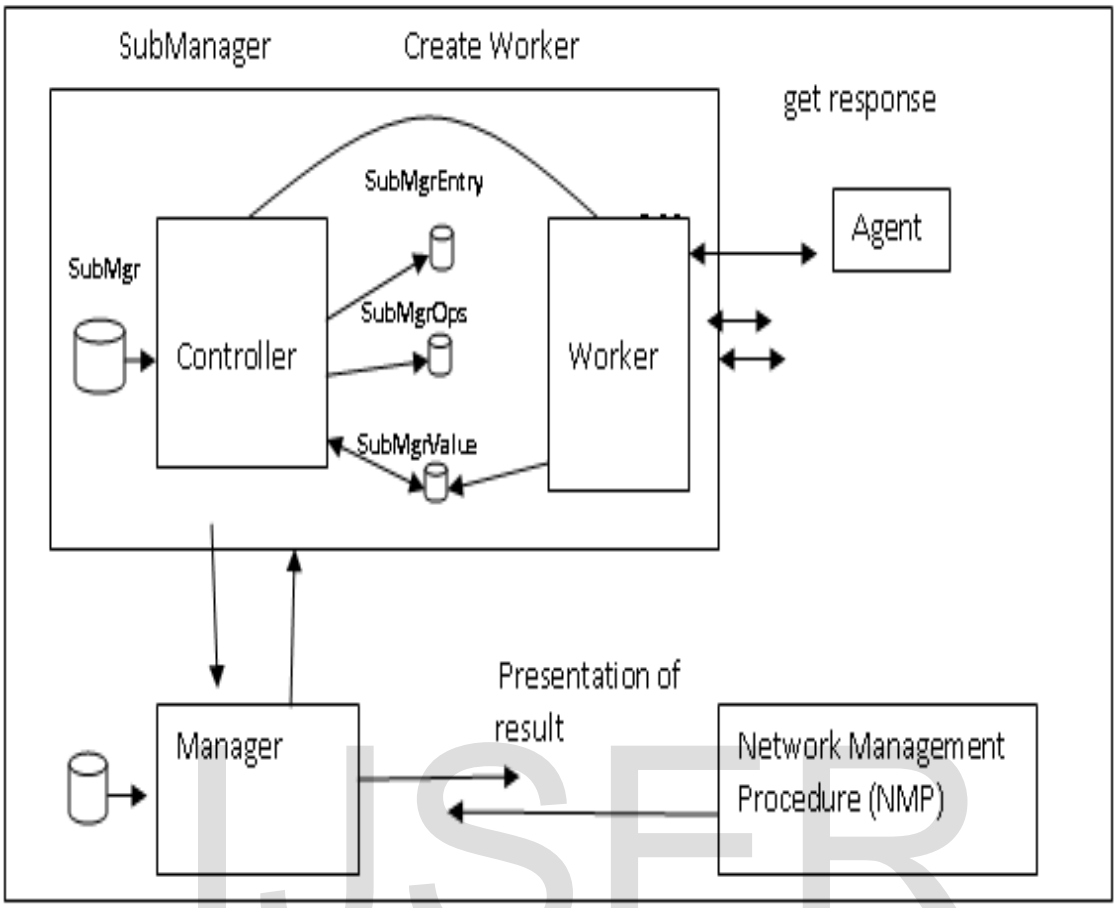

Figure. 3: The Internals of a Context SubManager

C. DISTRIBUTED MANAGMENT STRATEGY

Distributed network is a collection of physically separate, possibly heterogeneous computer systems that are networked to provide users with access to the various resources that the system maintains.

A distributed system should use interconnected and independent processing elements to avoid having a single point of failure. There are also several other reasons why a distributed system should be used. Firstly, higher performance/cost ratio can be achieved with distributed systems. Also, they achieve better modularity, greater expansibility and scalibility, and higher availability and reliability.

Distribution of services should be transparent to users, so that they cannot distinguish between a local or remote service. This requires the system to be consistent, secure, fault tolerant and have a bounded response time. The form of communication in such systems is referred to as client/server communication. The client/server model is a request-reply communication that can be: synchronous and blocking, in which the client waits until it receives the reply; or asynchronous and non-blocking, in which the client can manage to receive the reply later.

Remote Procedure Call (RPC) is well-understood control mechanism used for calling a remote procedure in a client/server environment. The idea is to extend the use of procedure call in local

environment to distributed systems. This results in simple, familiar and general methods that can be implemented efficiently.

The Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) is also an important standard for distributed object-oriented systems. It is aimed at the management of objects in distributed heterogeneous systems. CORBA addresses two challenges in developing distributed systems: making the design of a distributed system not more difficult than a centralized one; and providing an infrastructure to integrate application components into a distributed system. The distributed approach is a peer-to-peer architecture. As shown in figure 4, multiple managers, each responsible for a domain, communicate with each other in a peer-system. Whenever information from another domain is required, the corresponding manager is contacted and the information is retrieved. By distributing management over several workstations throughout the network, the network management reliability, robustness and performance increases, while the network management costs in communication and computation decrease. This approach has also been adapted by ISO standards and the Telecommunication Management Network (TMN) architecture. The Management model for ATM networks, adapted by ATM Forum, is based on this approach, too.

Various variant of the distributed strategy have being proposed among whom are: distributed big brother, from the University of Michigan; Distributed Management Environment, from the Open Software.

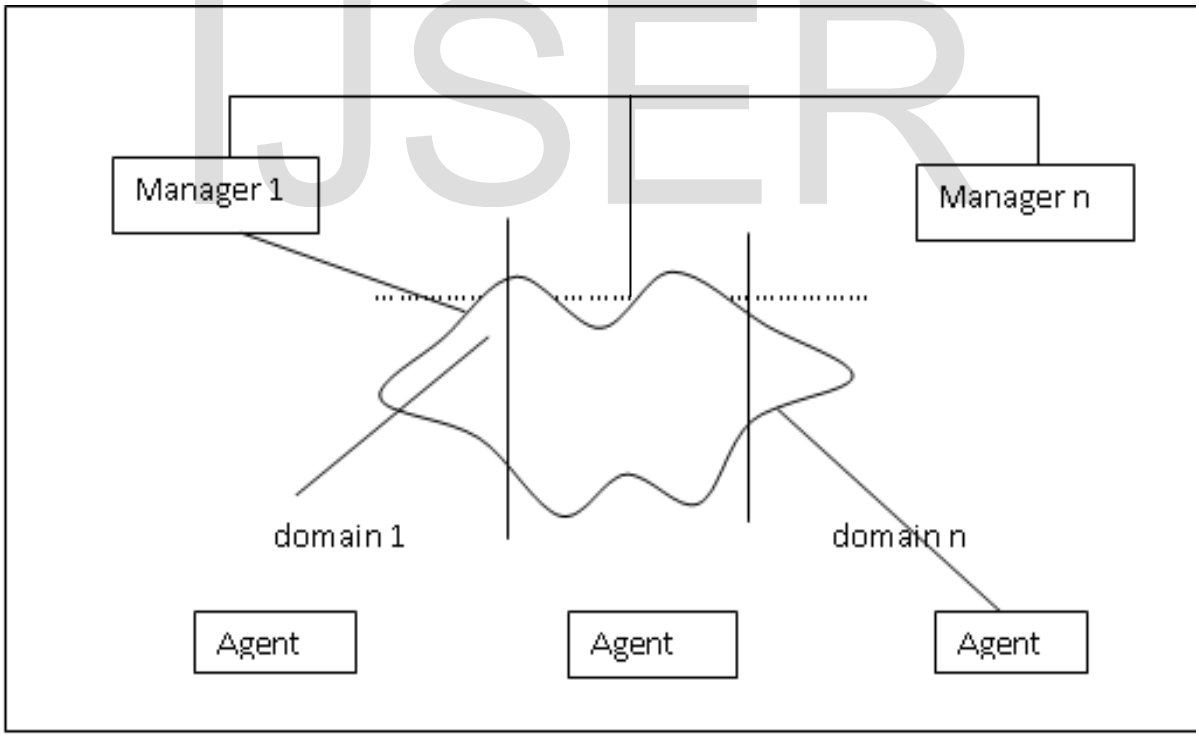

Figure 4 – A conceptual diagram of distributed strategy

D. HYBRID CONTEXT MANAGEMENT STRATEGY

Hybrid context management strategy in wireless technology is a paradigm where by context information is being maintained in order to provide timely information to an adaptation algorithm or manager using a combination of hierarchical and distributed strategy to alter the content and behaviour of context information to suit current condition. Hybrid context strategy tends to combine hierarchical and distributed management strategy to achieve thorough and all encompassing processes of collection, exchange and processing of context information. A combination of hierarchical and distributed context management strategy in wireless network is also known as the Network strategy or architecture.

This architecture as shown in figure 6 below, uses both manager-per-domain and manager of managers concepts, but instead of a purely peer-system or hierarchical structure, the managers are organised in a network scheme. This approach preserves the scalibility of both systems and provides better functionality in diverse environments.

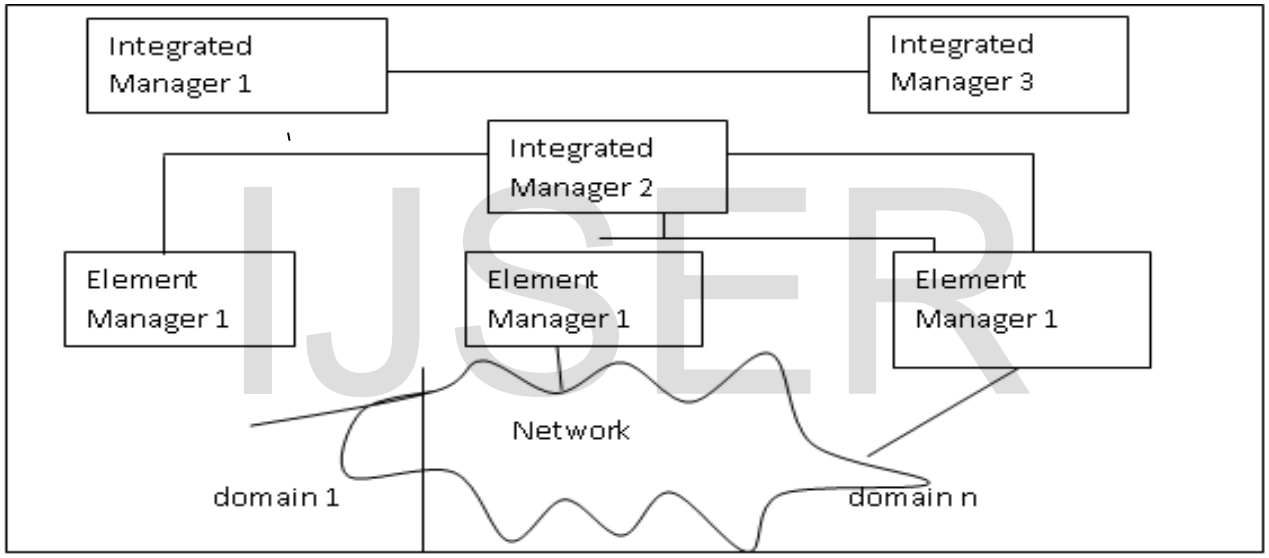

Figure 6: Hybrid context management strategy Architecture

CONTEXT AWARE COURSE REMINDER

Context-Aware Course reminder is an application of context aware strategies as outlined above. The design given here is based on the design of the m3 Context Manager. The purpose of the Context Manager is to gather and collate information from several context sources and make this information available to other components of the platform.

This work employs the waterfall software development process as shown in figure 7 below

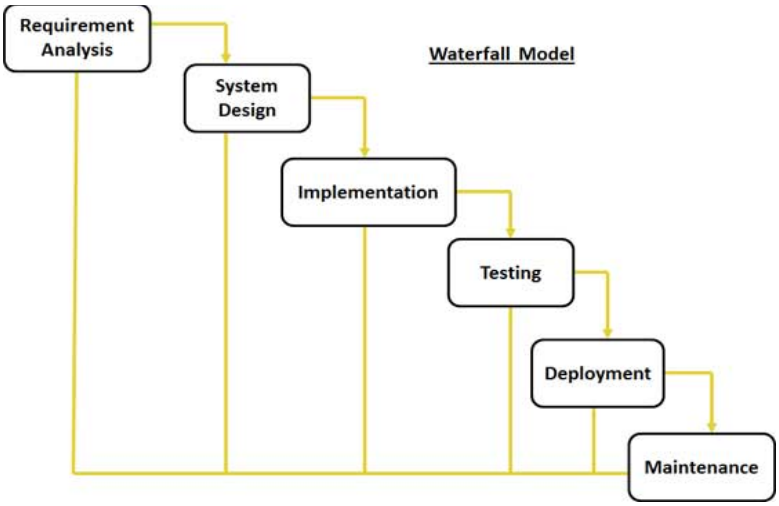

Figure 7: Waterfall Design Cycle

UNIFIED MODELING DIAGRAM FOR CONTEXT-AWARE COURSE REMINDER DESIGN

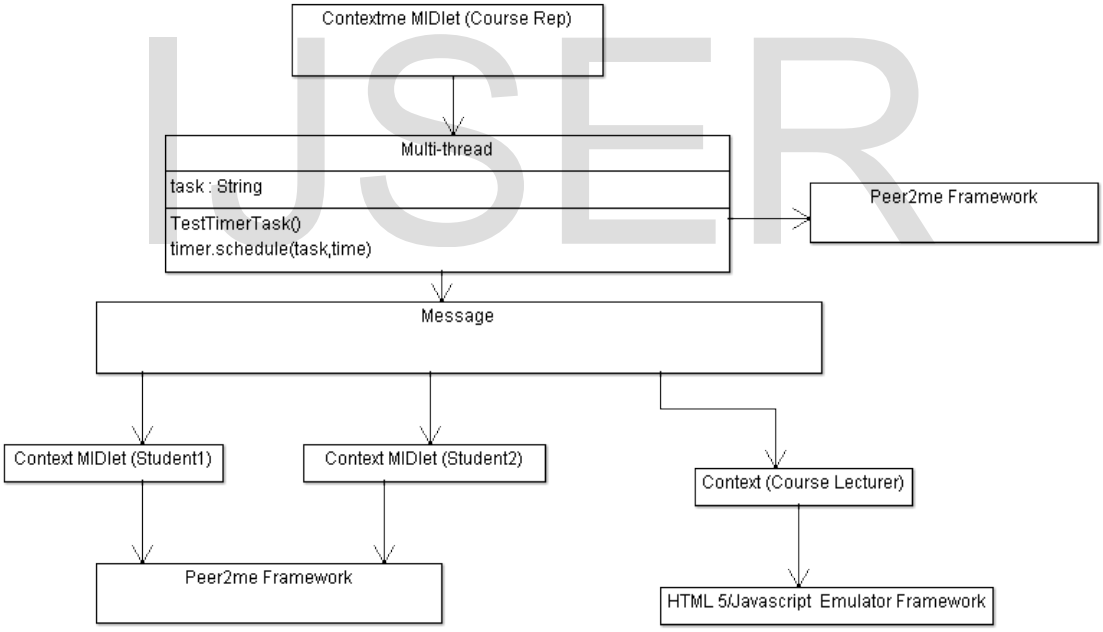

Figure 8: Unified Modeling Diagram for the design of Context Aware Course reminder

Although the context sources, or awareness modules, are part of the Context Manager itself, the Context Manager can best be described by separating the awareness modules from the part that performs the collation and storage of context information.

In Figure 9 below, the awareness modules are segregated from the Context Manager.

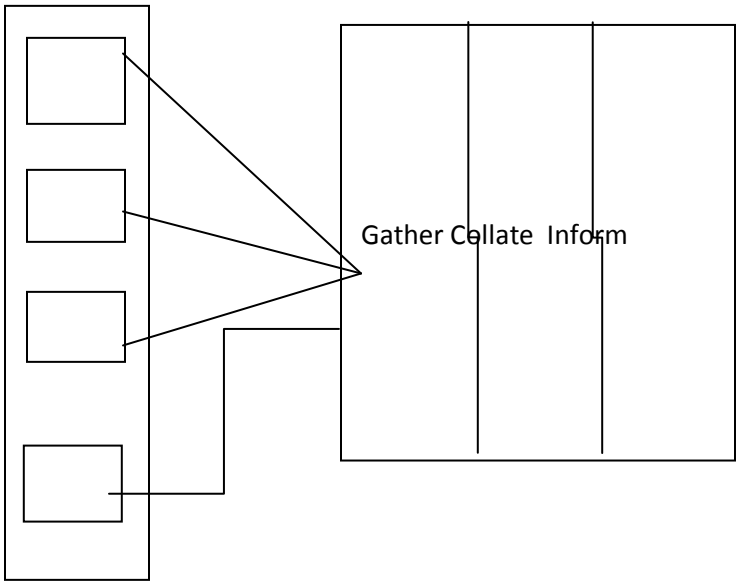

Figure. 9: M3 Context Architecture

Figure 9 shows that there are three main tasks to be performed by a context manager. Firstly, it needs to gather raw context information from the various context sources.Examples of context sources include a location module, an enterprise information systems and mobile device which can transmit device capabilities and user preferences.

COMPONENTS OF THE DESIGN

Accordingly there are several operations that may be invoked on the Context Manager. They are briefly outlined here. The methods can be roughly categorized according to the granularity of context. In practice, these methods reside in a stub on client side, where the client is taken to be one of the components that require context information from the Context Management System.

A. MIDLET

A core feature of the MIDP (Mobile Information Device Profile) technology is its support for developing mobile phone user interfaces. The MIDP provides a set of Java APIs known as the LCDUI (Liquid Crystal Display User Interface), which has functionalities similar to the Java Abstract Windows Toolkit (AWT) and Swing APIs in the desktop world. MIDP (Mobile Information Device Profile) applications are piquantly called MIDlets. It is a standard API in Java that was call up using the import command, this interface enables testing of mobile application in a virtual environment. It will suffice to say that if an application runs successfully on the Midlet, the possibility of running on real mobile phones is guaranteed except for cases of minor hardware configuration.

This is invoked using the import commands below:

```
import javax.microedition.lcdui.*;
```

```
import javax.microedition.midlet.MIDlet;
import javax.microedition.midlet.MIDletStateChangeException;
```

B. MULTI-THREADING

Thisis invoked in the program through the timer and task scheduler. It is instantiated using the declarations

```
private TestTimerTask task;
private TestTimerTask2 task2;
private TestTimerTask3 task3;
```

It is responsible for the scheduling of the time in which notifications are propagated to the context clients. This is controlled by:

```
timer = new Timer();
  task = new TestTimerTask();
  timer.schedule(task,5000);
 //timer = new Timer();
    task2 = new TestTimerTask2();
     timer.schedule(task2,50000);
    // timer = new Timer();
  task3 = new TestTimerTask3();
timer.schedule(task3,80000);
```


C. OTHER COMPONENTS OF THE DESIGN

The design extensively utilizes the concept of Peer2Peer framework in capturing the wireless network aspect of the research, which is the technology on which reminder messages are propagated, in emulating a call to the course lecturer HTML 5 JAVA script emulator was used, this automatically puts up a call to the course lecturer after a specific period of time which is determined by the thread on timer and scheduling.

IMPLEMENTATION

A. SYSTEM REQUIREMENT

The application developed in this work is suitable on mobile phones that are java compliant with Bluetooth Technology for wireless communication.

B. IMPLEMENTATION LANGUAGE

The implementation of this context-Aware reminder is done using Standard JAVA 2 Micro Edition (J2ME) interfaced with Java Script in HTML, the program is run using Netbeans 7.1, which is one of the Integrated Development Environment for Java programs. It consist of three set of codes that controls the context management process - context manager (courserep), the context sources (coursemates) and calling the course lecturer. These set of codes appears on the MIDLET as separate mobile apps that are activated collectively. The development process follows an object oriented programming paradigm using some advance features like Object,

Class, Data Hiding and Encapsulation, Dynamic Binding, Message Passing, Inheritance and Polymorphism

C. DESCRIPTION OF THE APPLICATION

The application implemented in this work is a context aware student reminder that reminds students of their scheduled classes and initials a reminder call to the course lecturer at a specific period of time to the scheduled time.

D. OVERVIEW OF THE APPLICATION

A context management system has been developed that captures the discovered neighbours and time requirements for a given process- class lecture in our case.

The system uses the J2ME programming language and bluetooth technology based on the Peer2me framework. Students including a Course Rep are represented as core MIDlet applications.

Three core actors have been captured: a class Course Rep (Contextme MIDlet) acting as a server, initiates the process of discovery using a bluetooth ad-hoc mechanism to detect potential context sources - students (Two Context MIDlets) in the vicinity of capture.

On detection and after the discovery process, the course rep can then send notification messages indicating the Course, Venue and Time of the lecture and eventually place a call to the course lecturer after a specified period of time. The calling lecturer process is emulated using a HTML5/javascript web app. Thus, the system goes through the process of node discovery, identification and messaging.

The messaging part is the most important and uses a multi-threaded cooperative scheduling event-handling technique coined MCSET. MCSET is enforced by using Timers classes and scheduling to ensure that messages emanating from the Course Rep arrive at destination clients (Class students and Course lecturer) at specified time intervals. Synchronisation of messages is done using a Timer Task thread schedule as given below:

I. THREAD A:

This is Message Part 1. Server (Course Rep) sends a text message of the form:
Math 201, Abuja, 2:00p.m, to client students for duration of approximately 5s.

II. THREAD B:

This is Message Part 2. Thread A is repeated for a duration of about 15s

III. THREAD C:

This is Message Part 3. Duration of about 80s is specified before a call is put through to the course lecturer. A method called “platformRequest” is used here.

E. APPLICATION ANALYSIS

The application is a context aware application that captures two important contexts in the wireless environment.

- Discovered Neighbours: From the literature review, it was stated that Schilit et al., 1997, the pioneers of context-aware computing, regard context to be location, identities of nearby people and object, and changes to those objects. Here the identities of nearby

people captured are the wireless neighbors of students with bluetooth enabled devices which are at a location of close proximity. Due to the limitation of the area of bluetooth coverage, the students need not be out of a radius of about 10meters.

- Time: The notification captures the context of time and propagates the message at a specific time interval; it also puts a call through to the course lecturer at a specific time interval to the lecture time.

F. COMPONENTS OF THE APPLICATION

The application consists of two sections, namely:

i. The Server Section.
ii. The Client Section.

Thereby applying the distributed context management strategy as proposed in Williams and Giadom, (2014).

The server section consists of the following classes:

i. SppServerStart: This class contains the main method and is responsible for initiating the server of our system.
ii. SppServer: This class is responsible for activating the blip node and registering it with the server apart from registering its services, it also initiates the 'SppServerSession' class.
iii. SppServerSession: This class extends a thread and obtains the input and output streams to communicate with the client over the wireless network. It also creates an instance of the 'SppServerDatabase' class which is responsible for writing into and reading out of the database.
iv. SppServerDatabase: This class implements the 'JDBC' to communicate with the database. This class together with the 'SppSserverDatabase' is responsible for communication with the client.
v. SppServerException: This class defines the exception of the type 'SppServerException'. All the classes at the server end call an instance of this class.

The client section consists of the following classes:

i. SppMIDlet: This class extends the 'MIDlet' class and initiates the midlet at the client (in our case a mobile phone). This class adds the 'SppBlipNodeList' class to the display of the device.
ii. SppBlipNodeList: This class is responsible for detecting the Bluetooth devices in the range of this client and register them with the client.
iii. SppDiscovery: This class is instantiated by the 'SppBlipNodeList' class and is responsible for detecting and registering them at the client end.
iv. SppCardForm: This class takes the name, phonenumber and sends them to the server end together with Bluetooth address of the node with which it is associated.
v. SppCardPositionForm: This class obtains the alert message from the server and displays it on the client mobile device.
vi. SppStream: Is responsible for communicating with the server. It contains the input and output streams to facilitate this communication.
vi. SppException: This class contains the 'SppException'. Any class containing methods throwing the exceptions of this type instantiate an instance of this class.

The interaction between the Server and the Client is presented over the Bluetooth Network. Firstly 'BlipManager' is launched which connects to the blipnode and gets a handle to it. To start the system we first start the Server by running the 'SppServerStart' class. This class consists of a main method which instantiates the SppServer class. The SppServer further creates an instance of the SppServerSession class which is responsible for the output and input streams that communicate with the client. This class extends a thread and starts a run method in which the input stream is constantly monitored for any incoming data bytes.

Now we start the application on the mobile phone. The application is started by the execution of the 'SppMIDlet' class. This class creates an instance of the 'SppBlipNodeList' and adds it to the display of the midlet. This 'SppBlipNodeList' provides the context manager (in this case the courserep) to select between different menu options, namely 'search' and 'notify', 'exit'. When the user selects the 'exit' option he can exit the midlet but if selects the 'search' option he can search all the Bluetooth devices that are available in the range. To detect all the Bluetooth devices in the range the 'SppDiscovery' class is instantiated. This class has methods that perform functions such as 'device discovery', 'device management' and 'service discovery'. Once the device discovery and device registration has been completed the user is prompted to enter the necessary parameters. The user can enter these details in the instance of the 'SppCardForm'. The client then stores this data at the server end. The server once it receives this data stores it and sends an acknowledgement to the client. The server then sends an alert message to the client if prompted to do so.

USING THE APPLICATION

To use the application, the courserep (context Manager) and students (context Sources/clients) need to activate the application on their mobile phones. On startup, a form on the application demands for the name of the user which is a way of identification. When the courserep which in this case is the context manager activates the application two interfaces come on for identification and typing of the messages to be propagated. After which the courserep initiates a search of students (Context clients) in close proximity. On discovery of these context sources/clients, the courserep(context manager) can then notify the other students (context clients) of the classes for the day with time and venue. After a specific time interval a call is automatically placed using Java script in HTML to initiate a call to the course lecturer reminding him/her of the class.

DISCUSSION OF RESULTS

This work is geared towards proposing to the academic community the efficacies of centralise, hierarchical, distributed and hybrid strategies in managing context in a wireless network.

The program designed has some distributed attributes which can be explained below.

A. THE CONTEXT MANAGER

The job of a context manager is to collect context information from context sources and make informed decision based on the context gathered. The MIDLETS APPs used as courserep is domiciled on all the mobile phones meaning that if there is a fault on any of the

mobile phones, any of the other phones can act as a context manager, this is an important part of distributed-ness, as there is no single point of failure.

This is one of the major reasons why centralized scheme is not a very robust mechanism of managing context information,where as in central approach, a single management machine collects the information and controls the entire network. This workstation is a single point of failure, and if it fails, the entire system could collapse. In case the management host does not fail, but a fault partitions the system, the other part of the network is left without any management functionality. Also, a centralized system cannot easily be scaled up when the size or complexity of the network increases.

The system uses interconnected and independent processing elements to avoid having a single point of failure. There are also several other reasons why a distributed system was used. Firstly, higher performance/cost ratio can be achieved with distributed systems. Also, it achieves better modularity, greater expansibility and scalibility, and higher availability and reliability.Distribution of services is transparent to users, so that they cannot distinguish between a local or remote service. This requires the system to be consistent, secure, fault tolerant and have a bounded response time. The form of communication in such systems is referred to as client/server communication. The client/server model is a request-reply communication that can be: synchronous and blocking, in which the client waits until it receives the reply; or asynchronous and non-blocking, in which the client can manage to receive the reply later.

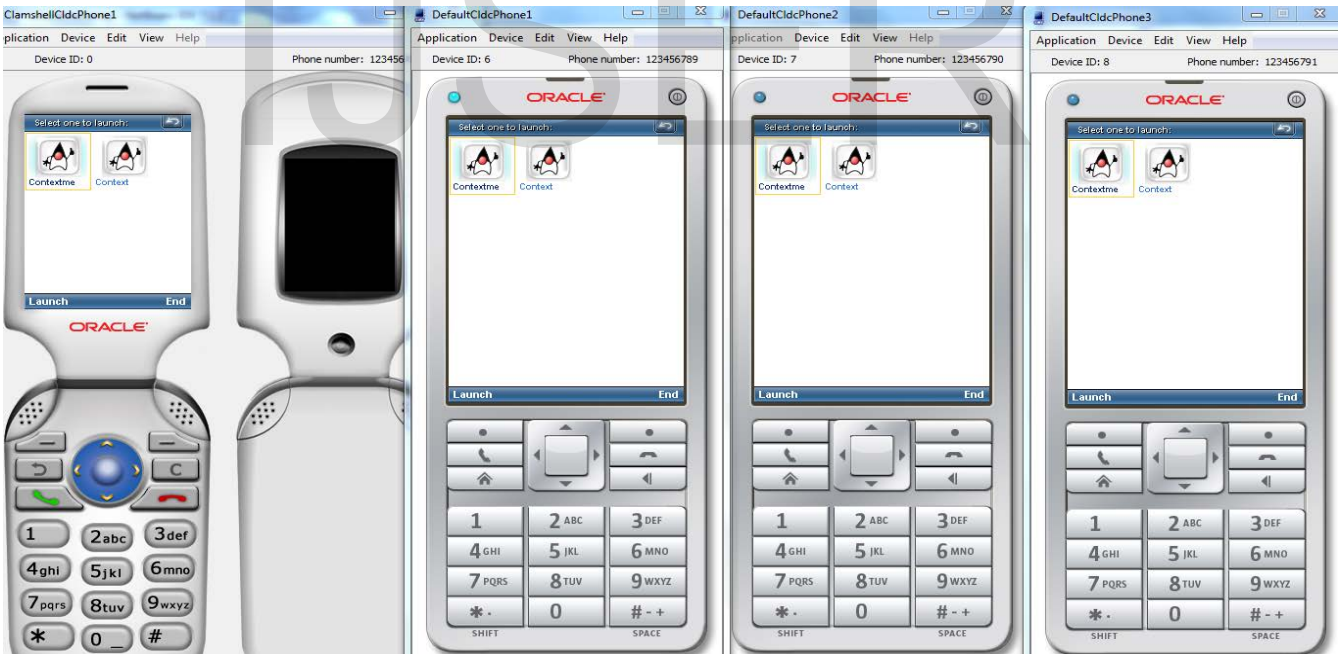

Fig. 11: Screen shot of distributed-ness of Context Manager

SELECTING STRATEGY

As demonstrated by the program it is obvious that distributed management platform is more suitable for most context aware applications and it is advisable to be used for the following reasons:

A. Neeed for distributed intelligence and control

Using the distributed management platform, there is a high level of distributed-ness thereby making application features transparent and robust.

B. Frequency of polling

There is a high frequency of polling in distributed system, the components of the system continuously watch out for one another to know if the other components are good for transmission

C. Ratio of Network throughput/amount of management information

The amount of work the computer needs to do per unit of time is reduce as there is a low resource requirement on the computer with distributed-ness, meanwhile the information gathered from the context sources is good enough and well filtered to take informed decision.

CONTRIBUTION OF WORK

This work has being able to analyze and demonstrate some strategies that can be employed by context application developers to effectively manage context information in wireless network, this has being achieved by making a detailed comparison of these strategies and demonstrating the effectiveness and efficiency of the strategies and by developing a context aware course reminder application, this will enhance informed decisions on the right choice of strategy to be used in a specific environment and in a combination of various environment.

CONCLUSION

This work has been able to bridge the gap in knowledge of context management strategy as regards wireless network, it has uncovered the strategies to be employed in managing context information that comes into a context aware system, it has demonstarted the use of these strategies by implementing a context aware course reminder application and it has also been able to make detail comparison of the various strategies and has given suggestions on the choice of strategy for optimal and efficient context management, of particular interest is the Hybrid strategy that can adapt to various environment owing to the fact that context applications are mostly environment specific.

REFERENCES

Chen G , Kotz D. (2004). A survey of context-aware mobile computing research. Dartmouth: Dartmouth Computer Science Technical Report.

Dey, A. K. (2000). Understanding and Using Context, Personal and Ubiquitous Computing Journals, vol. 5, no. 1.

Dey, A.K. Abowd, G.D. (2001) Towards a Better Understanding of Context and Context-Awareness, Workshop on the What, Who, Where, When, and How of Context-Awareness.

Giadom V.L. and Williams E.E., (2013), Context Management Strategy in Wireless Network, IEEE NIGERCON 2013 Proceedings, pg. 83-88, Nigeria

Mohsen, K., Peter B. H.W., (2009). Decentralised Approaches for Network Management, The Institute for Telecommunication Research (TITR), Elec. and Comp. Eng. Dept., University of Wollongong Northfield Ave, Wollongong, NSW 2522, Australia.

Musumba GW, Nyongesa HO. Context awareness in mobile computing: A review. Int J Machine Learn Appl. 2013;2(1), Art. #5, 10 pages. http://dx.doi.org/10.4102/ ijmla.v2i1.5

Schilit, B., Adams, N. (1997). Want, R. Context-Aware Computing Applications. 1st International Workshop on Mobile Computing Systems and Applications.

Schilit, B., Theimer, M. (1994). Disseminating Active Map Information to Mobile Hosts. IEEE Network.

Weiser M. (1991). The computer for the 21st century. Scientific American. 1991 issue 265(3). Pgs 66– 75. http://dx.doi.org/10.1038/scientificamerican0991-94

Williams, E and Giadom, V, (2014). Context Management Strategy in wireless Network, Internatioanl Journal of Advanced Studies in Computer Science and Engineering (IJASCSE) volume 3 issue 5, pg 7-14, http://www.ijascse.org/ijascse-volume-3-theme-based-issue-5