MobiGrid: A Middleware for Integrating Mobile Phone and Grid Computing

Muthoni Masinde
Department of Computer Science
University of Cape Town
Cape Town, South Africa
emasinde@cs.uct.ac.za

Antoine Bagula
Department of Computer Science
University of Cape Town
Cape Town, South Africa
bagula@cs.uct.ac.za

Victor Ndegwa
School of Computing and Informatics
University of Nairobi
Nairobi, Kenya
victormurage@gmail.com

Abstract—**The popularity and the high processing power of today's smart phones have presented computer scientists with a fertile platform on which to implement grid computing for mobile phones. Such grids will not require much investment since they are designed to make use of 'idle' power on already existing phones. This is because most smart phone users only use their phones for a few minutes or a few hours every day and yet, these phones are powered up 24/7. These kinds of grids are most favorable to developing countries where the penetration of mobile phone exceeds other forms of ICTs. Once in place, the grids can then be utilized to run the much-needed applications such e-health, e-education and drought prediction. In this paper, we present MobiGrid, a middleware for mobile phone grid that is part of a larger research project that aims at integrating mobile phones and sensors to come up with a drought predication tool for use in the developing countries. MobiGrid is an API on which distributed applications can be built. Unlike the rest of grid middleware solutions, the uniqueness of our approach lies in the fact that the middleware is for mobile phones environment**.

Keywords: Middleware, Grid Computing, Mobile Phones

I. INTRODUCTION

Among the many natural hazards that strike the developing countries, drought causes the most suffering on both human and animal kind. Though Information and Communication Technologies (ICTs) have been used to minimize the negative effects of such hazards elsewhere, the same cannot be said of developing countries of Africa which are characterized by poor infrastructure. Low Internet coverage and lack of electricity in these developing countries' rural areas for instance makes use of computer-based solutions in these areas impossible yet this is where natural hazards cause most harm. The good news is that Africa has reported impressive adoption levels of mobile phones [12], a phenomenon that is now creating a paradigm shift where computing is slowly moving from the traditional PC to the phone. However, scientists are not yet fully prepared for this shift and current mobile phone based applications are faced by an array of challenges the main one being interoperability issues. This paper presents results from an initiative towards addressing the interoperability challenge. We present MobiGrid; a middleware for mobile phones in a grid.

As technology advances and mobile phone users continue to demand for more functionality from their phones, more features are being packed into these little devices making them not just communication but also computing devices. This demand has led to introduction of smart phones; phones that offer advanced capabilities often like functions in a personal computer. Growth in number of Smart phones handsets shipped each year has continued in upward trend, for example, 24% average growth in handset shipment for smart phones is expected in 2010[5]. Given the high processing power[1] of smartphones and their support for a wide number of wireless connectivity options[2], these phones can ably compete with desktop computers of less than a decade ago. With all this processing power, mobile phones present a platform for distributed processing.

What does this huge computation power mean to the developing countries of Africa? It means that computer based applications that were a reserve for the developed countries and for resource-endowed cities of Africa could now become a reality for all. Mobile phone-based applications such as e-heath, e-education, e-farming, e-weather forecast and so on can now be found in the remote villages where the use of mobile phones has already preceded. However, given the inherent challenges[3] of using mobile phones as a computation device, there is still a load of work for computer scientists before this can become a reality. In this paper, we present the results of our initial efforts towards developing a middleware for mobile phones. This middleware provides a generic API on which mobile applications can be implemented to use resources of several mobile phones within a grid.

This paper is an extract from a larger research project whose overall objective is to combine the use of mobile phones and wireless sensors to develop an affordable drought prediction tool for the developing countries[7]. There are success stories of grid middleware solutions such as the BOINC (http://boinc.berkeley.edu/) and Folding@home (folding.stanford.edu/) but to the best of our knowledge, middleware for mobile phone grid does not exist. We therefore consider our contribution a major milestone towards 'software applications for all' in the developing countries of Africa.

---

[1] Ranging from tens of MHz to 1000MHz and RAM/ROM capacities of over 512 MiB
[2] Multiple GSM Frequencies (850, 900, 1800 and 1900), Bluetooth1, several data links (CSD, HSCSD, GPRS, EDGE) and Wireless local-area network (WLAN)
[3] Highly mobile; reliance on battery power that gets drained fast; hardware and software heterogeneity and highly personalized

978-1-4244-8909-1/$26.00 ©2010 IEEE

This paper was peer reviewed at the direction of IEEE Communications Society by subject matter experts for publication in the CNSM 2010 proceedings.

The paper is outlined as follows; **section II** gives the theory on which our middleware is built. The MobiGrid is presented in **section III** in terms of the architecture and how it works. We present our discussion, future work and conclusion in **section IV** while **section V** lists our references.

II. LITERATURE REVIEW

A. Grid Computing

Rajkumar[10], defines a computer cluster[4] as a type of parallel or distributed processing system, which consists of a collection of interconnected stand-alone computer nodes working together as a single integrated computing resource. Computer Clusters that are generally deployed for computationally intensive simulations have two variations depending on how tightly coupled the individual nodes are; *Beowulf Cluster* and *Grid Computing* that are tightly coupled and loosely coupled respectively. Grid computing is the category of computer clusters that MobiGrid is based on; its uniqueness being the fact that the nodes are mostly mobile phones.

Grid computing middleware (hardware and/or software) is very critical for the operation of the nodes, its main roles being to support Single System Image (SSI) and to ensure enhanced system availability. The middleware enables the nodes to take advantage of services available in the grid transparently hence freeing the end user from having to know where an application will run. It also makes it possible to view all system resources and activities from any node as well as supports check pointing. MobiGrid is an implementation of grid middleware; its function is to perform middleware roles.

B. Wi-Fi-based Long Distance Networks (WiLDNet)

Wi-Fi presents the developing countries with many advantages such as low installation cost (no wires needed) and wide availability (many devices have Wi-Fi inbuilt in them). Vibrant research and developments in Wi-Fi continue to yield new desirable features such as security (via the Wi-Fi Protected Access Encryption – WPA2), Quality of Service protocols and power saving mechanisms. So far, the development with the greatest potential for Africa is the **Wi**-Fi-based **L**ong **D**istance **Net**works (**WiLDNet**), which provides low-cost[5] alternative to the existing connectivity solutions that are currently costly to implement in the rural areas of Africa. Indeed, WiLDNet finally addressed the headache of limited geographical coverage of conventional Wi-Fi networks. WiLDNet is very promising; Ermanno et al for instance reported coverage of 279 km for the EsLaRed project in Venezuela[9].

Given the infrastructural and budgetary challenges that characterized most developing countries of Africa, a solution built around Wi-Fi would be a welcome idea. MobiGrid is designed with such countries in mind and hence the main reason for choosing Wi-Fi as the communication medium.

[4] Computer clusters are also known as Network of Workstations (NOW), Cluster of Workstations (COW) or Workstation Clusters

[5] Low cost is attributed to the use of cheaper single board computers that consume less power and cheaper off-the-shelf 802.11 wireless cards and towers

C. MobiGrid Design

1) Challenges of using Mobile Phone in a Grid

Given the following inherent characteristics of mobile phones, building a distributed application that runs on mobile phones is indeed a daunting task.

(a) Mobile phones are personal possessions and are almost always in the possession of their owners. This makes getting physical access to the phone when it's idle difficult for a third party;

(b) Mobile phones are, by definition, mobile, even with some way to tap into the computing power of the phone, one must contend with the fact that this power will not always be physically located where it's required;

(c) Mobile phones vary in terms of their design and capabilities. While one device may be well suited for one task, another may be better suited for another. The capabilities of these phones may also not be compatible;

(d) Most mobile phones, especially in African countries, are low end and may be limited in terms of their processing power and the applications they can run;

(e) Mobile phones rely on batteries for power. These batteries drain very fast especially under heavy load. Any attempt to utilize idle time would result in the battery draining even faster.

2) Functional Requirements for MobiGrid

MobiGrid was designed to mitigate some of the challenges described above. It was designed around the following functional requirements:

(a) As a system to establish communication between the various nodes, complete with a messaging system with a finite set of commands understood by the nodes;

(b) A way of determining which nodes are in the grid and what services they offer;

(c) A way to determine a coordinator as well as an election system to manage the transparent election of new coordinators in case of failure;

(d) A facility for service discovery, capable of monitoring all active and failed nodes to determine which services are still available; and

(e) Ability to recover from faulty communication.

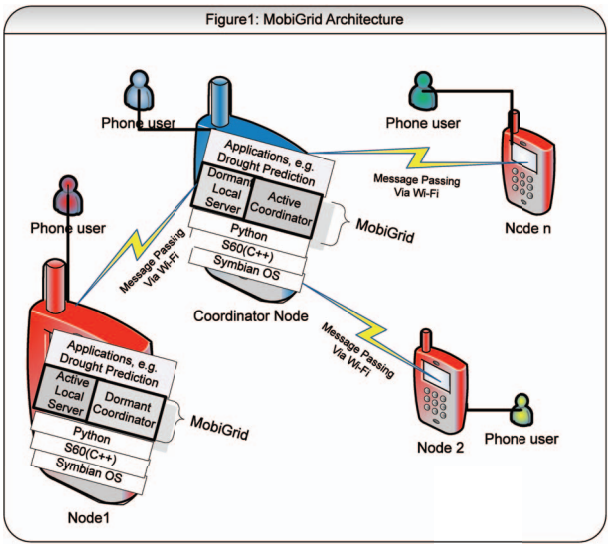
III. MOBIGRID IMPLEMENTATION

A. Overview

In its current form, MobiGrid was designed to run on smart phones and in order to ensure the highest compatibility, phones running Symbian OS (most popular operating system at the time of implementing this project) were selected for implementation. It was implemented using Python programming language for S60 (pyS60) and tested on two phones models: the Nokia E63 and Nokia N95. In order to introduce services similar to those offered by a conventional distributed operating system, a middleware that run on top of the Symbian OS was introduced. This middleware was charged

with the responsibility of determining the capabilities available on a given handset, the load on the handset as well as the battery charge remaining. A handset that is low of battery power for example would not be allowed to act as a coordinator for the grid since it has a fairly high chance of failing. The nodes (mobile phones) in MobiGrid communicate using messaging passing via a Wi-Fi ad hoc network.

B. MobiGrid Architecture



As shown in ***figure 1,*** the implementation of the MobiGrid requires that the application be installed on all phones that are to participate in the grid. On the phone that is currently playing the coordinator, (***Coordinator Node*** in this case) the two main modules; ***Local Server*** and ***Coordinator Server*** are executed. The Coordinator Server acts as the master to all the nodes while the Local Server services requests directed to the custom applications running on this phone. It also maintains state information on this phone, such as free memory available, services running etc. However, due to limitations of S60 (at the time of developing the application), a phone could not run more than one instance of the application. The non-coordinator phones (**Nodes 1**, ***Node 2***, …, ***Node n*** in this case) only run the Local Server instance of the application while the phone acting as the Coordinator run an instance of Coordinator Server. The Coordinator role may be exchanged if/when an election takes place.

C. How MobiGrid Works

As summarized in ***figure 2***, the four main operation of MobiGrid are; initialization, requesting for service, monitoring servers' status and electing a coordinator

1) Initialization Steps
This has the following three steps:

(a) The user starts MobiGrid on the phone;

(b) MobiGrid searches for other phones running MobiGrid. If a phone running MobiGrid's Coordinator Server is found, the phone connects to it as a client. If no Coordinator is found, the Local Server on this phone calls for an election and the winner of the election becomes the Coordinator.

(c) Once a Coordinator has been found, MobiGrid connects to it and registers all services it may have available. These services are provided by custom applications (such as e-health) that run on top of MobiGrid running on the phone.

2) Requesting a service
This step is executed when request(s) to run a particular custom application running on the grid is made.

3) Monitoring Servers' Status
The following commands are used in monitoring the servers' status

(a) ***!Poll*** – Local Server checking if Coordinator is still alive. The Coordinator uses the same command to find out if the Local Servers are running;

(b) ***!Alive*** - a reply from the Coordinator to Local Server(s) indicating that the Coordinator Server is up and running;

(c) !***PollResponse*** - a response from a Local Server to the Coordinator indicating that this particular Local Server is still up and running. The Coordinator then updates status of the Local Server by calling *updateClientState.* This status includes parameters such as battery power, memory capacity and network strength of the phone on which this Local Server is running

4) Elections
The first node to join the MobiGrid automatically becomes the Coordinator and other nodes register with it on joining the grid. The steps involved in finding the coordinator are as follows:

(a) The Local Server broadcasts to a given SERVER_PORT (default is port 2904) asking who the Coordinator is;

(b) The Coordinator responses and registers the Local Server.

(c) If no response is received within a designated TIMEOUT period, the Local Server calls for an election by broadcasting an ***!Election*** message to a designated CONTROL_PORT(default is port 7609). If a node with better ***election attributes***[6] receives a call for an election, it responds by calling its own election. If the caller of the election is not challenged, it automatically becomes the new Coordinator. The bully election algorithm [2] is used.

In its current implementation, MobiGrid picks the Coordinator as the node with the best election attribute. For example, if two nodes ***A*** and ***B*** are being compared against each other in an election, with ***A*** having 75MB free memory (***MA***) and 57 units of charge remaining (***CA***) with ***B***'s 41MB free memory (***MB***) and 79 units of charge (***CB***), then the vote V_A for ***A*** will be:

[6] Determined as a function of the phone's ***free memory*** and ***battery charge*** remaining..

$$V_A = \frac{M_A}{M_A + M_B} + \frac{C_A}{C_A + C_B}$$

$$V_A = \frac{75}{75+41} + \frac{57}{57+79} = 1.0656669371$$

and the vote V_B for ***B*** will be:

$$V_B = \frac{M_B}{M_A + M_B} + \frac{C_B}{C_A + C_B}$$

$$V_B = \frac{41}{75+41} + \frac{79}{57+79} = 0.934330629$$

In this case, ***A*** has better election attributes than ***B***. This process is repeated for all pairs of nodes still taking part in the election until only one overall winner is left. Note that ***VA***+***VB*** always add up to 2.

D. MobiGrid's Communication Framework

Communication between nodes on the MobiGrid is through messages passing. Messages are passed through the Coordinator process as JSON objects encapsulated in UDP packets. A third-party library, json.py, provides support for JSON, which is not available in PyS60. JSON is very close to python's own representation of dictionaries, making it very easy to convert between the two. A dictionary object is converted to JSON for transmission over HPPT then back to a dictionary once received on the other end.

IV. Conclusion and Future Work

This paper presents a proof that grid computing on mobile phones is feasible. A middleware called MobiGrid was developed and implemented using Python programming language. It was tested on Nokia E63 and Nokia N95 phones running Symbian Operating System. Through this middleware, the phones were able to operate in an ad hoc network using Wi-Fi and communicated using message passing. As the name suggests, this is a 'middleware on which distributed applications, which target a computer grid kind of environment can be implemented.

In its current form, the MobiGrid has some limitations that form part of further work; not all features needed by a middleware for a grid are included in the MobiGrid. In particular, the following limitations are noted:

Currently, the middleware can only recognize coordinators using the subnet mask 255.255.255.0. Further, not much attention was paid to enforcing security. Given that MobiGrid was tested on only two types of phones, more useful results could be achieved by use of an emulator. Best ways of executing this are currently being investigated. Further, in order to test MobiGrid further, two mobile-based applications (one for drought prediction and the second one for conserving Africa's tradition knowledge on droughts) are currently under development.

References

[1] Chen-Khong and Rajkumar, B. 2005. SensorGrid: Integrating Sensor Networks and Grid Computing. *Computer Society of India*

[2] Couloris, J., Dollimore, G. and Kinberg, T. 2001. Distributed Systems – Concepts and Design. Addison-Wesley, Pearson Education, UK.

[3] ITU. 2010. Global Telecom Indicators for the World Telecommunication Service Sector. *ITU Global Telecom Indicators for the World Telecommunication Service Sector* 2010, 3-3. http://www.itu.int/dms_pub/itu-t/oth/23/T23010000040001PDFE.pdf.

[4] Jin, H. and Jiang, W. 2010. Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice. Information Science Reference, Information Science Reference.

[5] Juniper Research. 2010. Next Generation Smartphones. *Next Generation Smartphones* 2010, 3-3. http://www.juniperresearch.com/reports.php?id=256.

[6] Kazem, S., Minoli, D. and Taieb, Z. 2007. Introduction and Overview of Wireless Sensor Networks. John Wiley and Sons, Inc

[7] Masinde, Masinde. and Antoine, Bagula. 2010. A Framework for Predicting Droughts in Developing Countries Using Sensor Networks and Mobile Phones. In *Conference of the South African Institute of Computer Scientists and Information Technologists (SAICIST 2010),* ACM, South Africa

[8] Pande Lab Stanford University. 2010. Folding@home - a distributed computing project 2010

[9] Rabin, P., Sergiu, N., Sonesh, S. and Anmol, S. WiLDNet: Design and Implementation of High PerformanceWiFi Based Long Distance Networks

[10] Rajkumar, B. 1999. High Performance Cluster Computing: Architectures and Systems. Prentice Hall, PTR.

[11] Teik-Kheong, T. and Benny, B., Eds. 2003. World-Wide Wi-Fi; Technological Trends and Business Strategies. John Wiley and Sons, Inc, Hoboken, New Jersey.

[12] TeleGeography; Africa Tops Telecoms Growth Thru 2013; May 2009; Accessed on August 30 2010; http://www.telegeography.com/cu/article.php?article_id=28440