A Model for Collaboration Services between Mobile Devices

(Full Paper)

M. Lee Son and A.P. Calitz
Department of Computer Science and Information Systems
Nelson Mandela Metropolitan University, P.O. Box 77000, Port Elizabeth, 6031
Tel: +27 41 5042326, Fax: +27 41 5042831
Email: Matthew.LeeSon@nmmu.ac.za / Andre.Calitz@nmmu.ac.za
18 April 2006

Abstract – Collaboration services enable users utilising mobile devices to share information and communicate with each other. Research has been conducted on various models that support collaboration. However most of these models are not scalable and depend on specific technologies. Additionally these models are not extensible, meaning that they only support designated services. The models depend on an existing wireless infrastructure and operating within the range of a wireless access points. Ad-hoc or infrastructure wireless environments, allow users to make use of these services and not rely on a wireless access point. This paper discusses the design and development of an extensible model to support collaboration services between mobile devices in a wireless environment. A prototype framework is created to validate the proposed model.

Index Terms – collaboration services, mobile devices, extensible, collaboration model.

I. INTRODUCTION

The development of technology over the past years has changed the way in which users communicate. People are the increasingly making use of mobile devices. Typical applications running on these mobile devices can be broadly classed into two categories, namely individual applications or collaborative applications [1].

Individual applications are local to the mobile device and its user. It is initiated and executed completely on a single device. Examples of simple applications are phone books and calendars. Collaborative applications, by definition require collaboration between or distribution of the computational load across two or more devices [1]. This implies that several devices work together to achieve a common goal or task between them. Typical examples of collaboration services are email, video conferencing, file sharing, instant messaging, etc.

Collaborative applications usually make use of an infrastructure wireless network environment, where communication between devices is relayed through a base station (wireless access point). However these wireless sessions are dependent on the access point and being in range of the access point. In the event of the access point failure or out of range, the session is no longer valid or accessible. An alternative to the above network is and ad-hoc wireless network. Devices in the network communicate with one another directly rather than through a central controller or access point.

Extensive research has been conducted in this area and various collaborative models have been developed. The Collaborative Problem-Solving Framework and the Decentralised Ad-Hoc Groupware API and Framework are two models that have distinguishable characteristics. These models were utilised in the design and development of the proposed extensible model.

Section II of this paper reviews the two existing collaboration models. Section III outlines and discusses the proposed extensible model. Section IV focuses on the implementation of a prototype framework based on the models outlined in Section II and Section V highlights future work.

II. RELATED WORK

Collaboration models allow concurrent users to share resources and information. Users can further communicate with each other or work on a common task. The models rely on an existing wireless infrastructure and wireless access points. The models are not extensible and provide specific services. This section reviews two models that uniquely contribute to the development of the proposed extensible model.

A. A Collaborative Problem-Solving Framework for Mobile Devices

A Collaborative Problem-Solving Framework for Mobile Devices [1] allows wireless mobile devices to collaboratively form a wireless session using the grid computing paradigm. The grid computing paradigm allows the pooling of unused mobile resources to form a computational grid in order to solve any computational problems that would not have been possible to complete individually. The model decomposes the problem into smaller tasks and distributes them across other mobile devices. Figure 1 illustrates the decomposition and distribution of a problem.

Mobile devices can take advantage of the computational benefits offered by this approach by using a subset of a communication protocol as a means of communication within the framework. The framework comprises of four main components (Figure 2):

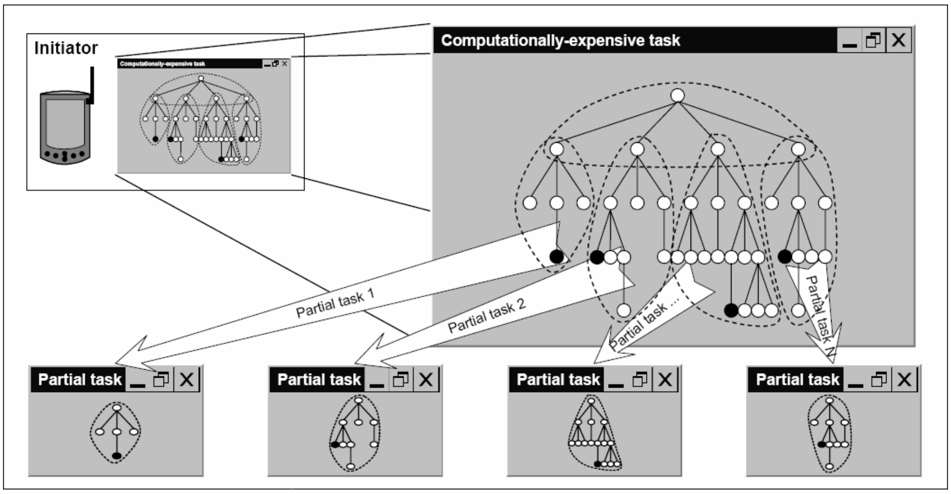

Figure 1 - Decomposition and distribution of a problem [1]

- Subordinates;
- Initiator;
- Brokering service; and
- Keep-Alive Server.

Subordinates are clients running the application on their mobile device. They can request to collaborate with other devices by helping to find a solution, request to leave the grid, initiate a task, run assigned sub-tasks or return completed sub-tasks.

An initiator is any subordinate requesting a computational task to be solved by the computing grid. It is the responsibility of the initiator to submit the request to the Brokering Service which will then utilise the pooled computational resources. The initiator at any given time can retrieve any completed sub tasks or abort the initiated task.

The brokering service handles all communication between the devices in the computation grid. In addition it contains two major information stores, Active Agent Repository (AAR) and Task Allocation Table (TAT). The AAR contains information about all devices participating in the grid computation, while the TAT is responsible for maintaining and updating information on distributed tasks in the grid. A device completing the task assigned sends the information block back to the brokering service to be stored in the TAT.

The Keep-Alive server is a TCP/IP service that is responsible for handling all communication aspects in the grid infrastructure. The Brokering Service maintains an accurate record of the current state of the grid. Messages are sent to and received from all participating devices use a keep-alive protocol. Mobile devices participating in the grid periodically send messages to the server to signify that they are still active. The time-sensitive nature of the keep-alive protocol requires that network devices communicate directly with the server to avoid time-outs. Any devices entering the wireless network and are willing to participate in the grid infrastructure are advertised to the Keep-Alive server as being available. A special mechanism is implemented to ensure that computing devices do not over use the resources in the grid without making their own resources available.

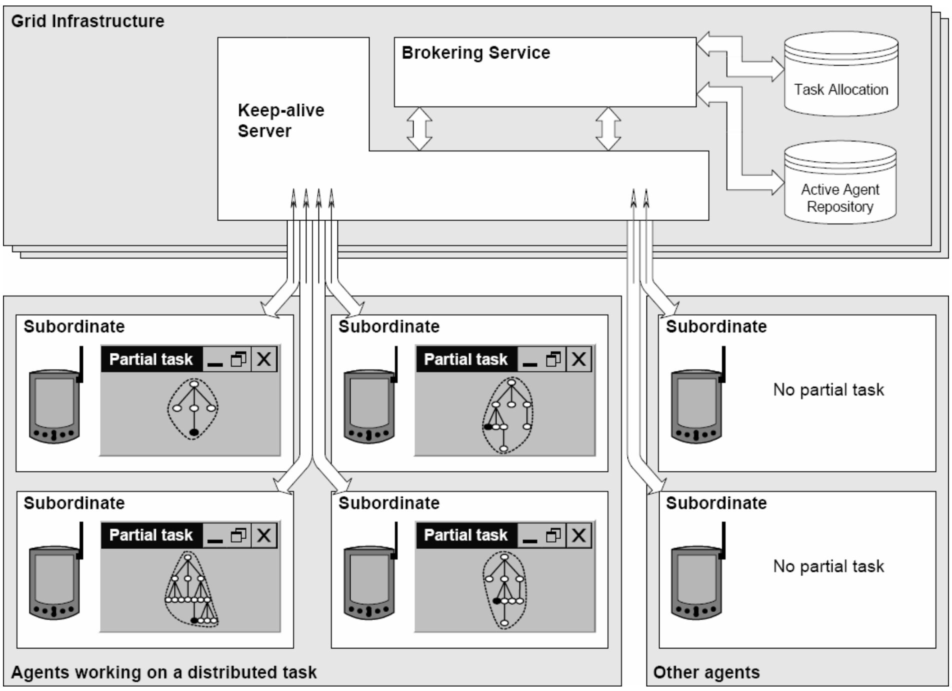

Figure 2 - Grid Based Collaborative Framework Architecture [1]

Devices that are very mobile, moving into and out of the grid will cause problems for the Brokering service and Keep-Alive server. These services can quickly overwhelmed by a large number of devices entering and exiting the grid infrastructure. Load balancing techniques are used to preserve and restore grid stability in the event of it becoming unstable.

The above discussions show that when designing services for mobile devices, developers must take into account the entering and exiting of mobile devices in the wireless session. The model highlights the importance of the communication services.

B. Decentralised Ad-Hoc Groupware API and Framework for Mobile Collaboration (YCab model)

The Decentralised Ad-Hoc Groupware API and Framework for Mobile Collaboration [2] allows developers to rapidly develop custom collaboration applications for the mobile environment. In addition the fault-tolerance is incorporated into the design of the framework so that collaborators floating in and out of the ad-hoc network won't cause network instability.

The YCab model is divided into two parts, the framework API and services. The framework is responsible for establishing an environment to share information from services with other collaborators over an ad-hoc network. Services can be easily created with this framework as developers do not have to worry about the underlying network facilities.

The framework consists of the following components (Figure 3):

- Communication and service managers;
- Session and election services;
- State recovery manager; and
- GUI components.

A high degree of flexibility is built into the framework, allowing developers several levels of uses when designing the desired mobile application. Figure 3 illustrates the several uses of the YCab API. The API is further divided into 3 broad level of usage as shown in Figure 4.

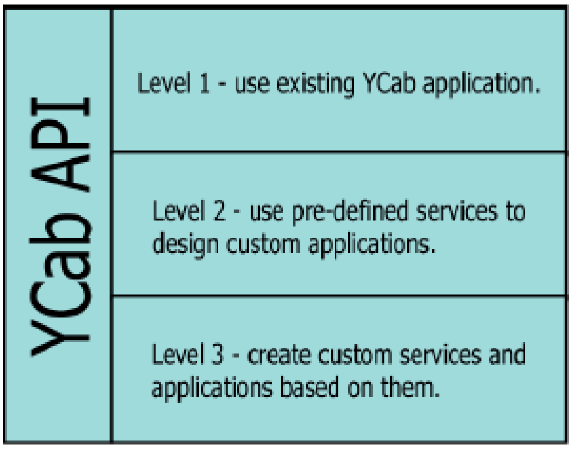

Figure 4 - Three levels of use of the YCab API [2]

Level 1 allows developers to use the application provided by the API or at Level 2 developers can develop their own application using the provided services. However, developing your own applications requires the developers to have a good understanding of the framework. At Level 3 the most complex level, a skeleton service class is provided to the developer. Powerful modules created can easily be incorporated into any custom application.

The main aims of this model were to:

- Provide a framework for use by collaborative services in a wireless ad-hoc network; and
- Provide a lightweight and flexible API that allowed developers to rapidly create highly optimised collaborative applications to run on mobile devices.

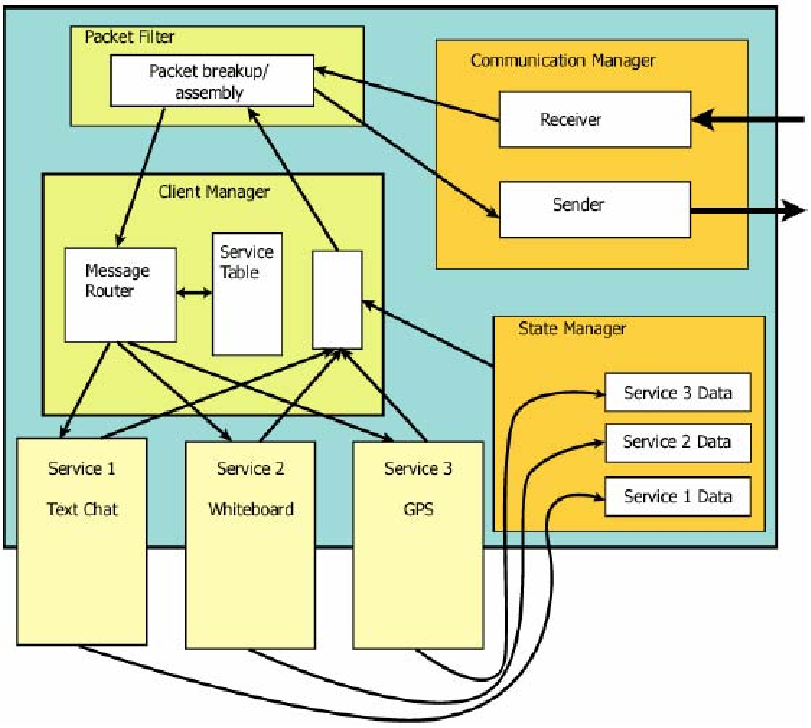

Figure 3 - The YCab API Framework [2]

After reviewing the YCab model, there are two unique features that distinguish it from other collaborative systems. Firstly, the framework supports an entirely decentralised system, whereby all clients run the same software. Secondly, multicast packet communication is used between clients. Although each client receives the message, only relevant messages to that client are processed.

III. PROPOSED MODEL

The proposed model utilises various characteristics from the models discussed above. Extensibility is incorporated into the model, allowing developers to add new features to the model without disturbing any existing modules. This section outlines and elaborates on the various modules in the proposed model (Figure 5).

A decentralised approach is used in the design of model. In a centralised approach, the host controls all communication and data exchange within the wireless network. Although this provides better control, a single point of failure is evident. For example, if the host fails, the system fails thereby terminating all running and associated processes. In addition this approach requires the host to have sufficient processing power to manage all communications within the network. A decentralise approach on the other hand eliminates a single point of failure by distributing data over multiple devices. This is important as mobile devices can easily be disconnected from the network, reasons ranging from atmospheric disturbances to signal interference. Furthermore this approach assumes that all participating devices are equal, allowing control to be equally distributed, consequently creating a robust and stable wireless network. However this approach requires data to be replicated across all devices whereby careful monitoring and synchronisation mechanisms need to be in place to ensure system integrity.

Fault-tolerance architecture is also implemented in the design of the model because ad hoc networks are often plagued with frequent disconnections. Although this is not actually a failure, it is perceived as one.

A. Input / Output Manager

The IO (input /output) manager, illustrated in Figure 5, handles all communication within the wireless network. It is extensible as developers can write plug-in network technology modules that can be easily incorporated into the system. The bridge ensures that relevant data between the various network modules are not mixed up with each other.

Contained with in the IO manager is threaded sender object. It is responsible for sending outgoing data though a multicast socket on a specific port number. All outgoing data are pre-packaged by the data manager as an array of bytes, datagram packet. The packets are then placed in an outgoing queue in the sender object. The queue not only acts as a buffer but also help to ensure thread synchronisation. Messages are accessed by enqueue and dequeue methods provided by the C# language specification.

Given that the IO manager is responsible for receiving as well as sending, a threaded receiver object is used to 'listen' on a specific port for multicast packets sent across the network. As with the sender object, the receiver also has the notion of a queue, incoming queue. Since the goal is to have the receiver object do minimal processing, all incoming datagrams are quickly scanned and placed on the incoming queue. One the received datagram is enqueued, the IO manger notifies the Communication Manager of its arrival.

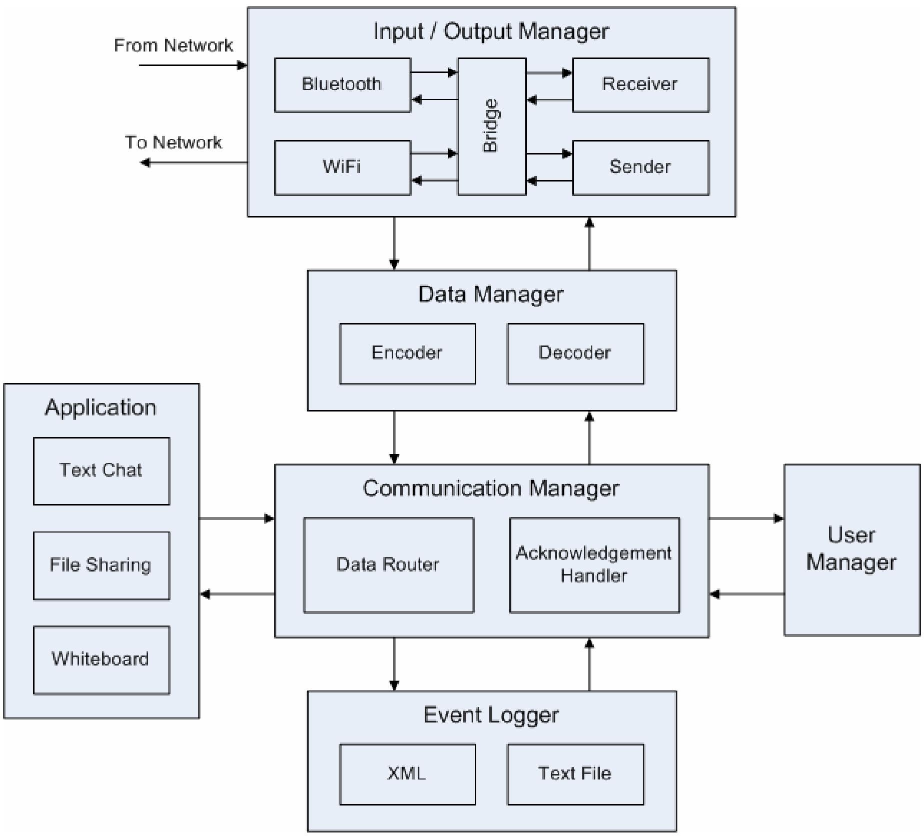

Figure 5 - Proposed Model

B. Data Manager

The data manger is responsible for encoding and decoding data in the wireless network. Any data sent in the network is encoded into an xml format. The xml is then serialised into an array of bytes for sending. Data received in the network, arrives in a datagram form. The decoder de-serialises the data to be used by the Communication Manager.

C. Communication Manager

The Communication Manger is the most important component in the model. It is responsible for processing de-serialised decoded datagrams, as well as keeping track of clients and ensures system integrity thought an acknowledgement handler. Additionally it is also responsible for initiating the other components in the model. Contained within the Communication Manger is the Data Router, which is responsible for handling all incoming data.

D. User Manager

The User Manager is responsible for keeping track of mobile devices in the wireless session. It is updated when users join or leave the session. A new user joining the session initially broadcasts its user and computer name to all devices in the session. Existing members receive the incoming datagram and updates their user list. In addition a datagram containing their user and computer name is specifically sent to the new user. It is important to note that if existing members broadcasted their identification, the existing algorithm would have to be modified to include a section to check for duplicate client credentials. This extra modification increases processor usage of an already limited processor power. Since many mobile devices have low processing power, the latter approach was not used.

E. Event Logger

The event logger is responsible for recording all communication in the wireless network session. This information can be reviewed for later use. Log files can be written in various formats to suit the needs of the user. Currently it supports XML and text file (.txt) format. Additional formats can easily be written and incorporated into the framework. A skeleton class will be provided to guarantee flawless compatibility.

IV. IMPLEMENTATION

A prototype framework has been partially implemented to validate the proposed model. The framework was implemented using the programming language C#. Careful consideration was taken to ensure that methods used were fully supported by the .Net Compact framework for mobile devices.

A modular approach was used which allowed modules to be tested and implemented individually. Currently the Data Manager and IO Manager have been completed. The Communication Manager is partially completed, with the Acknowledgment Handler still being implemented, while the Event Logger has yet to be completed. A text chat application has been used to test the partially completed framework.

V. FUTURE WORK

In order to fully validate the proposed model the remaining functionality has to be implemented. The Acknowledgment Handler, User Manager and Event Logger still have to be implemented and tested. When the framework is fully completed, a usability study will be conducted in a wireless mobile environment to test its effectiveness.

Additionally the data collected by the event logger can help isolate application difficulties or it can be mined to find interesting usage patterns. For example a text chat application logs can be data mined to see the most commonly discussed topic.

VI. CONCLUSION

Mobile teamwork is an emerging requirement for many organisations. The ability to share expertise, documents and information is the next step in information dissemination. Software do exist to fulfil collaboration needs, however most of them are designed to run on a stable and fixed network environment.

This paper presents a model whereby developers can harness the power of mobile computing without being concerned about the underlying communication of a wireless network. Additionally the framework can easily be extended. The implementation of a partial prototype framework shows the proof of concept of the extensible model.

VI. REFERRENCES

[1] Kurkovsky, S., Bhagyavati and Ray, A., *A Collaborative Problem-Solving Framework for Mobile Devices,* 2004.

[2] Buszko,D., Lee, W. and Helal, A., *Decentralized Ad-Hoc Groupware API and Framework for Mobile Collaboration,* 2001.

[3] Nunamaker, J.F, *Collaborative Computing: The Next Millennium*, Computer, September, 1999, pp 66-71.

[4] Kirda, E., Fenkam, P., Reif, G. and Gall, H., *A Service Architecture for Mobile Teamwork,* Distributed Systems Group, Technical University of Vienna, July 2002.

[5] *IBM Workplace Collaboration Services: unifying people, tools and resources to simplify day-to-day business,* http://www.lotus.com/products/product5.nsf/wdocs/workplacehome, 24 June 2005.

[6] Nunamaker, J.F, *Collaborative Computing: The Next Millennium,* Computer, September, 1999.

Matthew Lee Son is currently studying a MSc in Computer Science and Information Systems at the Nelson Mandela Metropolitan University.