Session Management and State Handling using HTTP connection on GPRS enabled mobile phone devices: A New Communication Model

Niaz Morshed Chowdhury[1], Syed Murtuza Baker[2], Kazi Md. Thoufiqur Rahman[3], Khandakar Abul Hasnat[4], Taskeed Jabid[5], Kazi Khaled Al Zahid[6]

Department of Computer Science and Engineering, East West University, Dhaka.

E-mail: [1]ex_notredamian@yahoo.com, [2]galib@ewubd.edu, [3]toufiq_the_best@yahoo.com, [4]reobd@yahoo.com, [5]taskeed@ewubd.edu, [6]polash@ewubd.edu

Abstract

Handheld devices such as cellular phone, PDA etc offer a whole variety of interesting and exciting possibilities for different applications coupled within an extremely resource-constrained environment. The ever growing popularity of internet, merging with the constantly increasing use of handheld, prepared the grounds for introducing this new type of services with impressively large application domain and use range. Multi-client applications, such as Multiplayer Gaming, Instant Messaging (Chatting) etc, are not popular on GPRS enable mobile phone devices as they need Socket connection to manage sessions and handle different states of the application. Socket connection is available only in expensive mobile phones that support MIDP-2. Low price mobile phone usually supports MIDP-1 that only provides HTTP connection. Largest group of mobile phone users, especially in developing countries usually use low price mobile phone and can not enjoy those multi-client applications. We have proposed a Communication Model based on HTTP connection that will allow the user to run multi-client applications on their mobile phone while using even a lower price set that only supports MIDP-1. Moreover, in this model no dedicated connection is used. As a result it will be cost effective and also creates less congestion on the network.

Keyword: Session Management, State Handling, MIDP, MIDlet, GPRS, J2ME.

I. INTRODUCTION

The use of handheld devices such as cellular phones and Personal Digital Assistance (PDA) has exploded in recent times. More than 350 million handheld devices are now used around the globe and it is expected to rise to 1 billion in the next few years [1]. The proliferation of mobile/wireless-internet has brought a new era in technology world. A General Packet Radio Service (GPRS) enable mobile phone is just like a computer having Internet connection on the palm. The devices vary rapidly in their operating system, storage capacity and communication resources. So, ample emphasis has been given on developing a powerful mechanism on supporting the application portability. Sun Microsystem's J2ME provides the best resource to develop a cross platform application with a cut-down java for the handheld devices. J2ME widely allows HTTP connection through Mobile Information Device Profile-1 (MIDP-1) which is good enough for web browsing and other applications that does not require session management and state handling. But in case of multi-client applications where session management is a vital issue, it fails. In this paper a communication model has been proposed to enable HTTP connection to work with the multi-client applications by creating sessions and handling different states.

II. ORGANIZATION

The organization of the paper is as follows: Section III indicates related work of developing different applications using J2ME, Section IV defines the necessary constraints that has to be maintained while designing the model, Section V and VI focuses on the architecture of our proposed model, Section VII and VIII describes detailed features and working procedures of the model and finally Section IX gives a cost analysis of the proposed model when it will be implemented as an application.

III. RELATED STUDY

Lots of works have been done to develop cross platform applications that are portable in most of the handheld devices. Bennett, Armstrong and Gupta [2] made a case study on message board client for handheld devices where the user would do chatting in the message board. The message board client could connect to the message board server, send a short note to the message board, and retrieve messages from the board. Retrieved messages contain author's name as well as the date. Coulton, Rashid, Edwards and Thomson created an entertainment application [3] for cellular phones, where they developed a system to update the user about the recent premiership football results as well as provided another very exciting fantasy game. Johnny Li-Chang Lo and Judith Bishop worked with the security of cross platform wireless applications. Different methods for secure communication and authentication between different mobile devices and servers by using different cryptographic means such as symmetric key encryption and public-key encryption have been proposed in their paper [4].

IV. CONSTRAINTS

This model basically concentrates on lowering the data communication expense and enabling the services of multi-client applications on the inexpensive mobile phone devices. To ensure these two aims, following constraints have to be considered.

A. Constraint 1: GPRS Based Model

Lots of SMS based multi-client applications can be found in the market [9], [10]. Those applications charge a full SMS sending cost for a single data communication step. As a result those SMS based multi-player gaming or chatting applications cost extremely high to use them. On the other hand, General Packet Radio Service (GPRS) [5], [11] cost is very low and Java 2 Micro Edition (J2ME) [6] supports all the necessary features that is needed to access internet through GPRS. That is why GPRS will be the most suitable platform for this model to work on.

B. Constraint 2: MIDP-1 and HTTP Connection

J2ME has two different versions of MIDP; MIDP-1 and MIDP-2 [6]. MIDP-2 has both Socket and HTTP connection [8] facilities to access the internet through GPRS. But this version is only supported by expensive mobile phone devices. Low price mobile phone devices support only MIDP-1 where HTTP is the only way to connect with the internet. So, HTTP will be the only connection method for this model to access the internet.

C. Constraint 3: Non-dedicated Connection

One of the straightforward ways to lower the data communication cost is to send data when it is needed. It is only possible when the connection will be a non-dedicated one. This model will have to use such a non-dedicated connection rather than using a dedicated one.

D. Constraint 4: Minimum Control Information

Multi-client application not only sends user provided data but also transmits some extra data which is used for state and session control. Those control information plays an important role to manage the application sessions and handle different states. This model should minimize this control information transmission as less as possible.

V. BASIC COMMUNICATION MODEL

Proposed model follows Client-Server architecture for its basic data communication by satisfying all the constraintts defined in the previous section.

A. Client

In this model mobile users are considered as clients. Client mobile phone device have a J2ME MIDP-1 application which is called MIDlet [6]. In other word we can say MIDlet is nothing but the gaming or chatting application that we have discussed in the previous sections. This MIDlet accesses the server through internet by using GPRS and provided the necessary services to the user.

B. Server

In this model Server runs on the world-wide-web and accessible through internet. Server contains a set of Java Servlets [7] to manage the server side task. Number of Servlet depends on the nature of the applications.

C. Bridge

In this Client-Server model Servlet plays the most important role [7]. It runs on the server and acts as a bridge between the clients. If a client wants to send any data to some other client, it needs to send the data to the Servlet first. Servlet processes the data and sends it to the desired client. In this way, two clients communicate with each other. Fig. 1 shows the basic communication model.

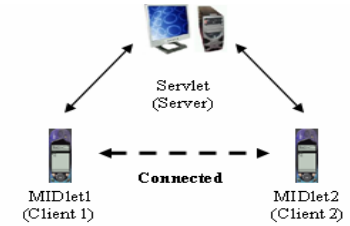

Fig. 1 Basic Communication Model

VI. EXTENTED COMMUNICATION MODEL

Though the concept discussed in the Basic Communication Model is the key principle behind this model, but life is not so easy to work with. This model can not work just with this minimum concept. It requires some extensions and changes to make this model work perfectly. Those extensions and necessary changes have been illustrated in the following:

A. Extension 1: Tokenizing

In the basic Communication Model, client needs to connect and communicate with the server and vise versa. As this model uses stateless HTTP connection [8] for data communication purpose, server can only do response to the requests that have been submitted to its queue. But server itself cannot initiate any request to the clients. As a result, a critical problem arises. Due to this ambiguous situation all the clients seem to be the same to the server, as it cannot identify them individually. So when a message arrives at the server, it will be mystified as to whom this message is to be forwarded. It will hamper the Basic Communication Model to work properly. To overcome this problem this paper

introduces “tokenizing” in the model that allows the server to identify the client discretely.

Tokenizing: Each client will be allocated a unique identification mark named *token*. Whenever a client sends any data to the server that token will be added in front of the payload data. At first server will read the token to know whose data it is and then take necessary steps according to the application requirements. Token allocation will be in bits. For an example, a maximum 256 clients supported application token will be in 8 bits.

B. Extension 2: Spooling

In the basic communication model server needs to send data to the other client using HTTP connection. But by using HTTP connection server can only do response to any request [8]. It cannot initiate communication with the client from its own. To overcome this problem *spooling* is introduced as an extension in the proposed model.

Spooling: Each client will spool the server after a particular period of time; such as 2 seconds or so on. While spooling, client will send a blank message containing just its token. During this spooling server will send the necessary information to the respective clients. As this spooling message contains extremely small data (just few bits), this model minimizes the control information to a grate extend.

C. Extension 3: Multi-Threading

Keeping User Interface (UI) active is an important task for the multi-client applications especially for simultaneously interacted applications. As this paper has used *spooling* for Server-Client communication, it will freeze the UI and create problems with taking inputs from the users. To overcome this problem multi-threading technique is used.

Multi-Threading: This paper embeds the main theme of Multi-threading to solve this freezing problem, that is using different threads for different types of tasks. If the spooling is done separately from a different thread for the communication process, its effect will not freeze the UI, which will be running under the main thread.

The final communication model is thus developed by combining the extended propositions with the basic model. This extended Communication Model keeps the principle of the Basic Model and integrates the necessary extensions to keep the process working properly. In this model each client will spool the server after a particular period of time from a different thread. While spooling the server, client will always hold the token allocated by the server at the front of its payload data to identify it exclusively.

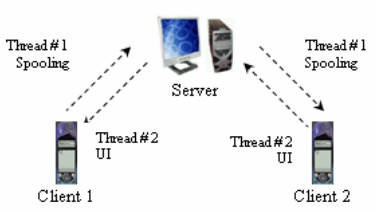

Fig. 2 Extended Communication Model

VII. SESSION MANAGEMENT AND STATE HANDLING

An entertainment pack has been developed using J2ME for the MIDP-1 enabled mobile phone devices in order to implement the proposed model. This pack contains two types of applications. One is a chat client, which is an example of simultaneous application and another one is a multi-player game, example of a turn based application. Both Simultaneous and Turn-Based Applications require some important states that control the flow and activity of the application. The total state control flows are discussed below.

A. Log in State

In this state a client will send the *Login_Request* to the server in order to login in the game or chat room. During initiating the login, client will add a special type of token *LogInReqToken* with his identification information. When the server receives data containing that special token, it identifies that user as a new one and allocates a *SessionToken* for that client. Session token will be in bits depends on the maximum number of client support provided by the applications. A file will be maintained in the server, which will store each client’s information against his allocated token to specifically identify that client. At the same time, server will start a new session for that user. This session will last for 30 minutes after the last data received or until the client *LogOff* from the room.

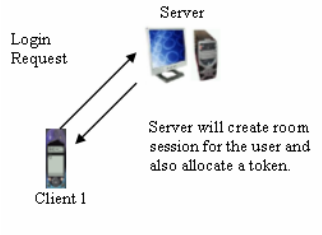

Fig. 3 Login State

Figure 3 depicts this working flow of this login mechanism where client 1 is requesting sending the *Login_Request* packet to initialize the login. Then the server allocates a new token to client 1 and creates a room session for that client.

B. Initiating State

In case of gaming a client can create a new game or can challenge some other client to play a game. Whatever

the situation, when any client accepts the proposal of playing any game, a new session in between those two clients will be started. In case of chatting when a client invites some other client to do a chat, a new chat session will be started. This session will last for 30 minutes after the last data received or until a user leave the game or chat.

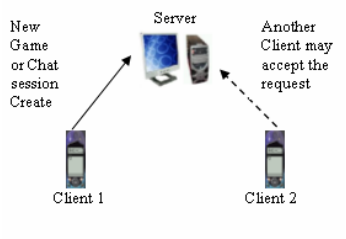

Fig. 4 Initiating Sate

In figure 4, client-1 is requesting the server to create a new game. When the game is created, client-2 accepts client-1's challenges and joins into his gaming session.

C. Moving State

Moving state works differently for Simultaneous applications and for Turn-based Applications. Their working principles are as follows:

***Simultaneous application*:** Simultaneous application clients interact simultaneously from the both sides according to the application purpose during this state. Client will spool the server after a particular period of time to send and get the necessary data. Here the server is not concerned about the time in between each interaction from a particular client. Any client can communicate anytime and can remain idle up to the maximum session time.

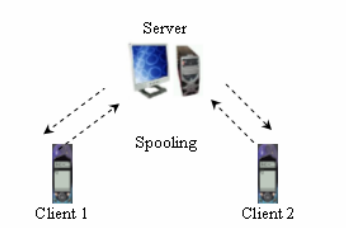

Fig. 5 Moving Sate (Simultaneous Apps)

Figure 5 describes the moving state in a simultaneous application.

***Turn-based Application*:** In turn-based applications, this state is a connectionless state. During this period a user may think or take his time to decide the proper move for a card game or a chess game. If he takes more than 15 minutes then his session will be expire and another player will be declared as the winner. Session time may change based on the applications behavior.

D. Waiting State

There is no waiting state in simultaneous application as any client can communicate at any time according to his requirement. It is found only in turned-based applications. While a client decides his proper move for any game or any other applications, another client remains in the waiting state. This is a connected state. After a particular period of time, client will spool the server to know that whether the opponent has given his move or not.

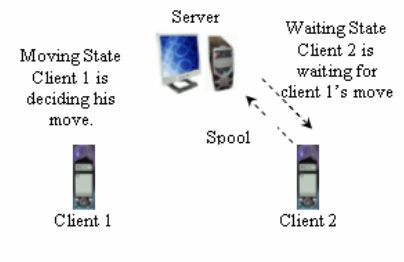

Fig. 6 Moving and Waiting State
(Turn-based Applications)

E. Winning State Identification

At the end of each move from a client, the server checks whether this is a winning state for any client or not. Whenever the server gets any winning state for a client, it declares that client winner and another one loser and the application moves to the leaving state. For the chatting applications there is no such state because there is no winning or loosing situation in chatting.

F. Leaving State

A client may leave any game or chat anytime. This state will close the game or chat session for that particular event (not the room session).

G. Logging off State

When a client quite the game or chat room the server makes his (leaving client's) token free for further allocation and clear his session. A new user may get this 'token' later.

Fig. 7 Log Off State

In figure 7, client-1 sends *LogOff* request to the server. In response the server sends a *LogOff* confirmation message to client-1. Thus the client-1 is logged off, his token is freed from allocation and finally all his information is removed from the session file.

VIII. STATE HANDALING FLOW CHART

Each and every application requires proper state handling to make it work in a perfect manner. In this model two types of applications have been considered, Simultaneous and Turn-based applications. Both the type needs proper state handling but their level of importance is different. From the nature of these two types of applications, it is clearly understandable that Turn-based applications need more states than Simultaneous applications. As a result, state handling becomes an important and vital part for Turn-based applications. On the other hand most of the time Simultaneous applications maintain very few states with one or two principal states. Applications usually stay 90% of its running time at those principal states. As a result its state handling becomes very straight forward and condition-free. But Turn-based applications jump from one state to another state again and again based on different conditions. Those jumps and frequent state change makes the state handling difficult and complex.
The whole procedure of state handling for the Turn-based application is crystallized in a flowchart as an example of a game in figure 8.

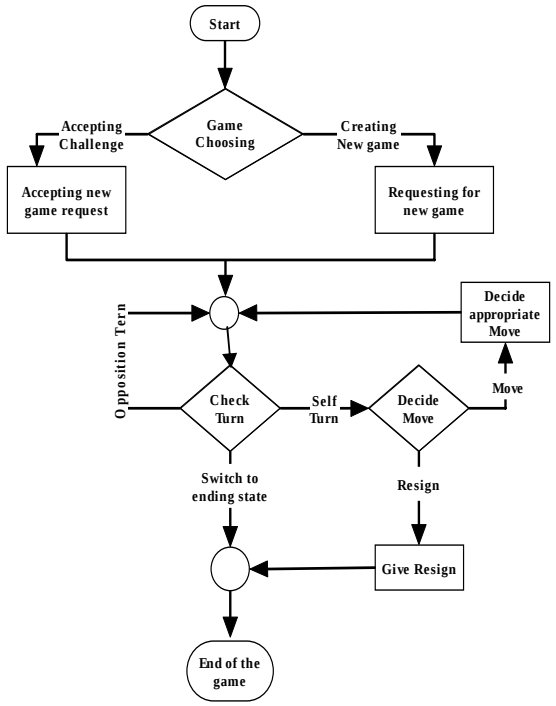

Fig. 8 Flowchart of the turn-based application

IX. COST ANALYSIS

This communication model is designed in such a way that the user can do chat or play game or run any multi-client applications at a very lower cost. This cost analysis is based on an operator operating in Bangladesh named "AKTEL" where its GPRS tariff is 1.5 Paisa for per KB data transfers [12].

A. Data Transfer Cost

A 512 character long array including token, packet header and other necessary elements require 1.5 Paisa to be transferred. It will allow a user to send 1 MB data at a cost of only 3 Taka. By sending 1 MB data a user may do chat for 1 hour or play complicated simultaneous game for 30 minutes.

B. Spooling Cost

Spooling is a necessary part of this model. Spooling cost varies with the maximum number of clients to be allocated in a room. To allocate 256 client in a room, token needs only 8 bits or 1 byte. In case of spooling, client only sends this token to the server. A 1-byte spooling at an interval of 2 second for 5 hour costs less than 3 Taka. Besides, in turn-based application, when a client stays in a Moving state model allows him not to spool. It will lower spooling cost for turn-based applications further more.

X. CONCLUSION

This paper presents a new communication model for handheld devices that allows the user to use multi-client applications on their less expensive mobile phone devices at a very lower cost. Moreover, this model does not follow any dedicated connection rather it works in discrete manner. It will create less congestion on the network while running any multi-client applications. It was created initially for mobile phones but can be configured to work in any sort of handheld devices. Future work focuses on implementing security features in this model to provide secure connectivity and authentication between mobile devices and the remote servers.

XI. ACKNOWLEDGEMENT

This work is a part of our (Author 1, 3 and 4) final year project "Mobile Entertainment Systems (MES)". We are extremely grateful to Syed Akhter Hossain, Chairman and Associate Professor, Department of Computer Science and Engineering, East West University, Dhaka. We are also thankful to Mr. Zakir Hossain Sorkar, Senior Lecturer and Ms. Sazia Mahfuz, Lecturer, Department of Computer Science and Engineering, East West University, Dhaka for their constant help and support.

REFERENCES

[1] Cellular Telecommunication Industry Association, 2002. *CTIA's Semi-Annual Wireless Industry Survey.*

[2] Jennifer M. Bennett, Mickey L. Armstrong, Swapna Gupta, "A Message Board Client for Handheld Devices" *ACMSE'04*, April2-3, 2004, Huntsville, Alabama, USA

[3] Paul Coulton, Omer Rashid, Reuben Edwards and Robert Thompson, “Creating Entertainment Applications for Cellular Phones”, ACM Computers in Entertainment, Vol. 3, No. 3, July 2005, Article 3B.

[4] Johnny Li-Chang Lo And Judith Bishop, Component-based Interchangeable Cryptographic Architecture for Securing Wireless Connectivity in JavaTM Applications, SAICSIT 2003, Pages 301 – 307

[5] Johan Mellberg, “GPRS Overview: Applications over GPRS”, Ericsson Publication.

[6] Jonathan Knudsen and Dana Nourie, “Wireless Development Tutorial Part I”, January 2006.

[7] Jonathan Knudsen, “Wireless Development Tutorial Part II”, January 2006.

[8] R. Fielding, UC Irvine, J. Gettys, “Hypertext Transfer Protocol -- HTTP/1.1”, Request for Comments: 2616

[9] Barb Dybwad, “Nokia files patent for Morse Code-generating cell phone”, Engadget, 12, March. 2005.

[10] Mark Henderson, “A race to the wire as old hand at Morse code beats txt msgrs”, Times Online, 16, April. 2005.

[11] Alan Sicher, Randall Heaton, “GPRS Technology Overview”, DELL Publication.

[12] http://www.aktel.com/post_tariff.php