International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 13 (2015) pp 33165-33169

mobileDBMS: A Client- Server RDBMS for Mobile Devices using Bluetooth Technology

Nitesh Kumar[1]
Department of Information Technology,
Maharaja Agrasen Institute of Technology
Delhi 110086, India.
nit.wadhera@gmail.com
Rohit Beniwal[2]
Division of Computer Engineering,
Netaji Subhas Institute Of Technology
Delhi 110078, India
rohitbeniwal@yahoo.co.in
Anup Singh Kushwaha[3]
Department of Computer Science & Technology,
Manav Rachna University
Faridabad 121004, India
Anup.usit@gmail.com

Abstract- In today's era, developing Relational Database Management System (RDBMS) supporting client server architecture is a very conventional phenomena to store & maintain data for aiding business needs. However, these days, people spend more time with mobile phones as compared to computer systems. Therefore, if the same development is to be done by using mobile phones supporting different operating systems using Bluetooth technology, it accumulates various problems such as low processing power, platform dependency, security issues, etc. Here in this paper, we provide an attempt to develop platform independent RDBMS for mobile phones supporting client server architecture using Wireless LAN or Bluetooth technology. Our approach is step forward to assist future business needs such as creating distributed RDBMS using mobile phones.

Keywords- mobileDBMS, Bluetooth, J2ME, RMS, jMeSQL.

I. Introduction

In the present era, wireless communication has a great role to play. Wireless communication had its inception at the turn of the 20th century with the invention of radio. Since then, Society got transformed and made the world a smaller place due to the power of instant communication over long distance. Moreover, mobile phones have turned conventional radio broadcasting's one-way model into two-way conversations. With the advent of 4G LTE, Wi-Fi, WIGIg, and other technologies, computer data networks have brought wireless communications into the 21st century [1]. Mobile phones provide connectivity to real world on the move i.e. without a wire. Smart mobile phones have added new dimensions to the business world. Whether we are in traffic or at home, we can handle our business using smart mobile phones. However, mobile phones at present have numerous disadvantages as compared to computer systems such as limited processing power, less security, ephemeral battery life, etc. Nowadays RDBMS is used frequently to provide an environment that is both convenient and efficient to store & maintain data aiding various business needs. Therefore, if the same functionality can also be provided by mobile phones too, it will turn out to be a major advantage for assisting future business needs. However, applications running on mobile phones are platform dependent. Further mobile phones have their own limitations as compared to computer systems, which are described above. Hence, in this paper, we provide an attempt to develop platform independent RDBMS for mobile phones supporting client server architecture using Wireless LAN or Bluetooth technology. We call it 'mobileDBMS'. In mobileDBMS, server is a mobile device whereas client can be either a mobile device or a computer system. A client connects to the server using Bluetooth technology and can then execute its queries. To achieve platform independency, we used SUN based Java2 Platform, Micro Edition (J2ME) kit [2], which supports applications to run on almost any platform. Further we used jMeSQL [3] for connecting client to mobile server using Bluetooth technology.

The rest of the paper is organized as follows: section II discusses the architecture of J2ME followed by illustration of Record Management Store and jMeSQL in section III; section IV examines the Bluetooth technology; section V expounds the architecture of mobileDBMS followed by its implementation in section VI and section VII concludes the paper.

II. Architecture of J2ME

The goal of this section is to discuss the architecture of J2ME, which is one of technology used for implementation purpose. According to TIOBE Index for May 2015 [4], Java is the most popular programming language. As it is a platform independent language, we can run java programs on

any computer system such as Windows, Macintosh, Linux, Solaris, Mainframe, and even on cell phones. However, developing applications for small computing devices such as mobile phones, set-top boxes, Blu-ray Disc players, M2M modules, digital media devices, printers, etc. cannot be simply done using Java 2 Platform, Standard Edition (J2SE) and Java 2 Platform, Enterprise Edition (J2EE) because of small computing power of these devices. To solve this problem, Sun Microsystem provides J2ME. J2ME consists of Java Virtual Machine (JVM), Configuration, J2ME APIs, Profiles, and strict subset of J2SE [5]. Fig. 1 shows the architecture of J2ME [2]:

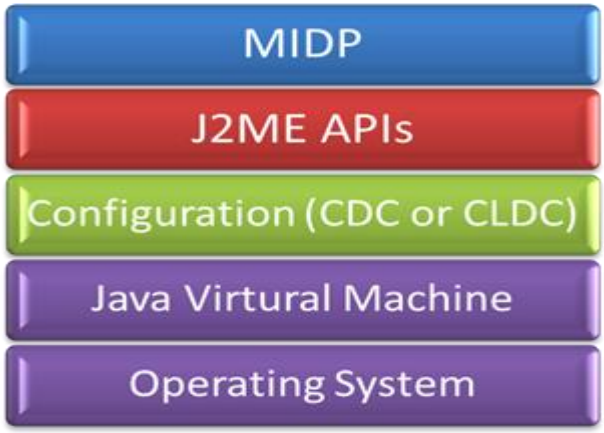

Fig.1. J2ME Architecture

The architecture of J2ME can be broadly divided into four layers, which are as follows:

1. Operating System layer: Each small computing device runs on some operating system like Symbian OS, Window, Android, etc. The J2ME works above this layer.
2. JAVA Virtual Machine: Like J2SE, J2ME also comes with JVM, which executes the byte code. J2ME provides two types of JVM: one for Connected Device Configuration (CDC) devices, which supports full version of JVM [2,5] and another for Connected Limited Device Configuration (CLDC) devices, which is a stripped-down version of JVM [1,5] aka Kilo JVM (KVM).
3. Configuration Layer and J2ME APIs: This layer consists of minimum set of APIs for small computing devices. J2ME supports two types of configuration [2,5]. One is CDC and another is CLDC.
4. Mobile Information Device Profile (MIDP): This layer consists of device specific set of APIs [3]. Each java enable mobile phone supports MIDP. The MIDP has several versions such as MIDP 1.0 [6], MIDP 2.0 [7], and MIDP 3.0 [8].

This paper specifically focuses on CLDC devices, which supports MIDP 2.0 profile or higher versions.

III. Record Management System (RMS) and jMeSQL

The aim of this section is to illustrate Record Management System (RMS) and jMeSQL, which are one of technology used for implementation purpose. The Record Management System (RMS) provides the persistence storage mechanism for J2ME devices. Fig. 2 shows the RMS for J2ME. In J2ME, r*ecord store* stores the data in the form of records. The *MIDlets* can store and retrieve the data using RMS [9]. There are two types of *record store*: one is private and another is shared *record store* as shown in the Fig. 2 [5]. We cannot access the private *record stores* of other MIDlet suite. However, access to shared record store is permissible [5]. Each *MIDlet* can create *record store* with a unique name in its own *MIDlet* suite. Although two different *MIDlet* suites can have same name private *record store*. In Fig. 2, *record store AB* [5] can be accessed by both *MIDlet* suite A and B, but *record store* 1 of *MIDlet* suite A cannot be accessed by *MIDlet* Suite B and vice-versa.

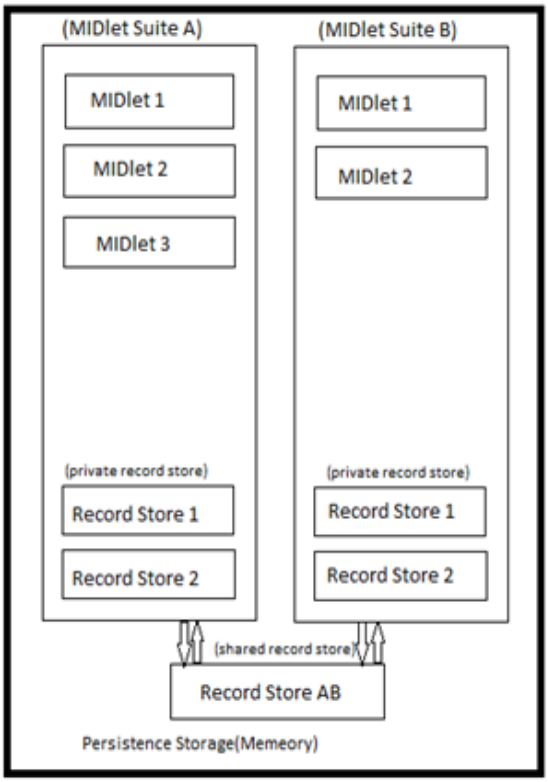

Fig. 2 RMS for J2ME

Record store stores the data in the form of byte array, thus it require conversion from string to byte array and from byte array to string every time we access the *record store* [2,5]. Each *record store* has two columns namely, recorded and byte array data [2,5].

jMeSQL is an open source Relational Database Management System (RDMS) for J2ME. The jMeSQL is a light-weight RDBMS. Source code for jMeSQL is available at [3]. Users can write their SQL command and execute it. In J2ME, data is not stored in form of table, thus jMeSQL is used to provide the mechanism to store the data in form of table. The user can create, alter, and delete tables in jMeSQL RDBMS. It works above the RMS layer of J2ME and uses the *record stores* to provide persistence to data.

IV. Bluetooth Technology

The objective of this section is to examine the Bluetooth Technology, which is one of mechanism used for implementation purpose. Bluetooth wireless technology is a short range radio technology. It provides a common protocol stack to transfer data between any types of mobile devices. The Bluetooth protocol is defined in [10]. In Bluetooth, one device acts as a master and another one serves as a slave [5] and they both form a Personal Area Network (PAN). Using Bluetooth, we can transfer text, audio or video data within a PAN. Bluetooth uses unlicensed 2.4GHz ISM band [5, 11].

J2ME provides a set of API for Bluetooth enable mobile devices [5]. Using them, we can develop applications for Bluetooth enable mobile devices. Java provides following package for Bluetooth technology [5]:

javax.bluetooth: Bluetooth classes and interfaces required by JSR 82.

javax.obex: OBEX (Object Exchange Protocol) classes and interfaces required by JSR82.

V. Architecture of mobileDBMS

Developing mobileDBMS has several goals, which are as follows:

1. To provide a relational database in J2ME Environment that can operate over Bluetooth network.
2. To enable the client to connect to central database server to process queries.
3. To provide a user friendly interface.
4. To provide platform independency.

Fig. 3 shows the mobileDBMS client-server system.

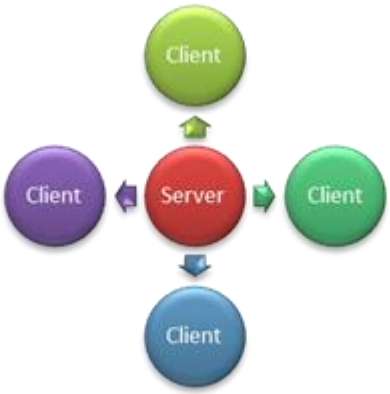

Fig.3. mobileDBMS client-server system

In this mobileDBMS client-server system, the database clients and server are communicating using Bluetooth technology. Both mobileDBMS client and server are running on a small computing mobile device with CLDC configuration and MIDP 2.0 profile.

In our work, we developed two *MIDlet* applications: one is for mobileDBMS server and another is for mobileDBMS client. The layered architecture of both applications is described below.

Fig. 4 and 5 shows the architecture of mobileDBMS server and client, which we developed in this project.

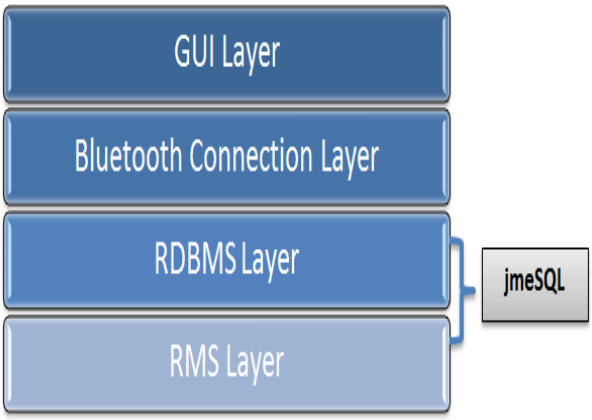

Fig.4. Architecture of mobileDBMS Server

The architecture of mobileDBMS server consists of four layers. The first layer is RMS layer, which is provided by J2ME for persistence storage. The second layer is RDBMS layer, which manages the data in form of tables and converts the tables into records for RMS layer and vice- versa for user. First and second layer are already implemented in jMeSQL, which is an open source RDBMS for J2ME. In our work, we used jMeSQL to provide RDBMS. The third layer of mobileDBMS server architecture provides the mechanism for open Bluetooth connection to mobileDBMS client. The fourth layer of mobileDBMS server architecture is GUI layer, which provides the mechanism for receiving the queries sent by the client and send the processed result back to the client.

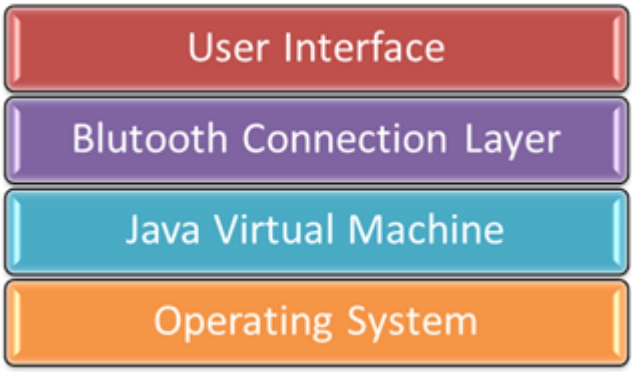

Fig.5. Architecture of mobileDBMS Client

The mobileDBMS client architecture consists of four layers. The first and second layers are in-built layers i.e. operating system and JVM. Third layer is Bluetooth Connection layer, which provides the mechanism for accepting Bluetooth connection from the server. It receives queries from the user, which are then sent to the server. The processed result from the server is also received.

The fourth layer is User Interface layer, which takes the input from user and also displays the received result from server to the user. The user enters the jMeSQL queries as an input.

VI. Implementation

We used Oracle SUN J2ME SDK3.0 and Sun Java wireless toolkit-2.5.2 for implementing our work. The screen shots of the mobileDBMS client and server using Sun Wireless

Toolkit are shown below. Fig. 6 shows the welcome screen when the server program gets executed.

Fig.6 mobileDBMS Server

When we press the start button, the server starts its service and switches on the Bluetooth of the device running this application.
Fig. 7 shows the running server.

Fig.7 mobileDBMS Server start running

When we press the start button, the server starts the Bluetooth service named SQL_SERVER and when we runs the client application, the client first searches for Bluetooth service named SQL_SERVER. If it founds the service, it shows the connection status in status box of client application.

Fig. 8 shows the client window. The client first searches for the server's services and if it founds the service, it gets connected to the server for processing the queries.

Fig 8. mobileDBMS Client connected to mobile DBMS server

Fig. 9 shows the client window executing the create table command. The command specifies the table as employee along with attribute as name. It also includes the data type for attribute as varchar.

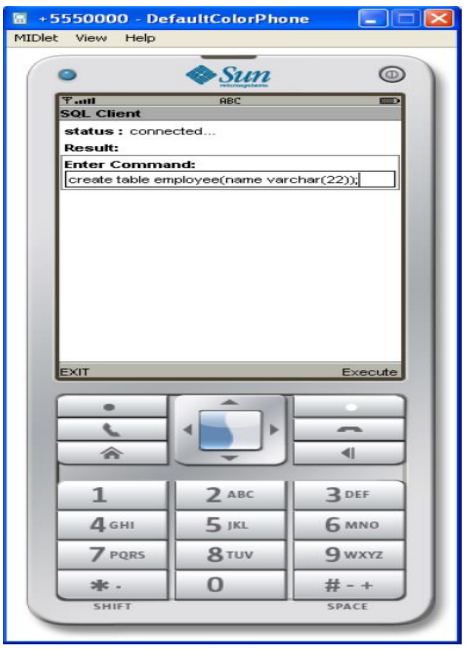

Fig.9 Creating table employee using mobileDBMS client

Fig. 10 shows that the query is executed successfully.

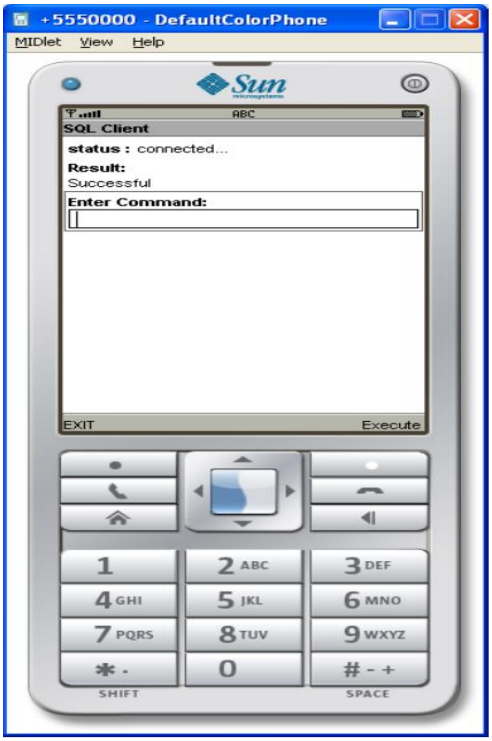

Fig.10. Create table command get execute and successful result displayed

Fig. 11 shows that the execution of select command. The command displays that employee table is created.

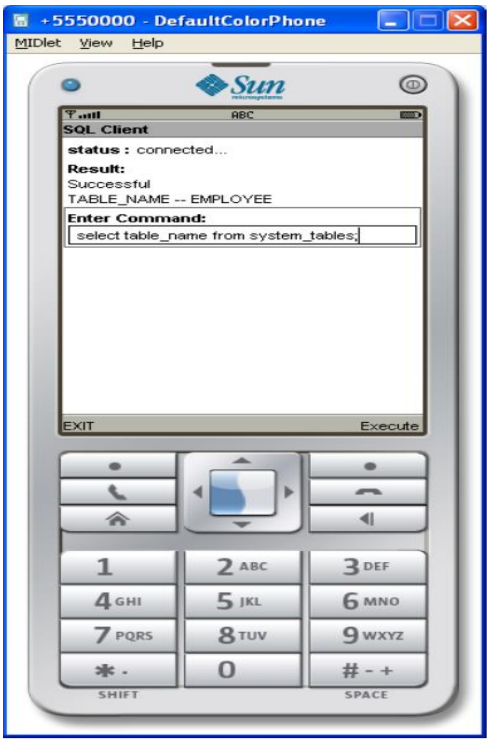

Fig.11. Selecting Table name exists in Database from system tables

VII. Conclusion and Future Scope

Using mobileDBMS client- server architecture, we can connect a client such as mobile or computer system to a mobile database server by employing Bluetooth technology. So far the mobileDBMS tested for one client which gets connected to server using Bluetooth connection and has worked successfully.

We would like to extend our work to assist future business needs of creating small business network containing distributed RDBMS supporting client- server architecture for mobile phones using Bluetooth technology.

References

[1] What Are The Advantages of Wireless Communication? Retrieved from http://science.opposingviews.com/advantages-wireless-communication-1121.html
[2] The Complete Reference J2ME by James Keogh, TMH, pp1-70, pp 296-299, 2003 Edition
[3] jMeSQLa open source RDBMS for J2ME. Retrieved from http://www.jMeSQL.net/
[4] TIOBE Index for June 2015. Retrieved from http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
[5] Beginning J2ME: From Novice to Professional, Third Edition by SING LI AND, JONATHAN,KNUDSEN, pp 1-10, pp 103-116, pp 197-207, 2005 Edition.
[6] JSR-000037 Mobile Information Device Profile (Final Release). Retrieved from http://jcp.org/aboutJava/communityprocess/final/jsr037/index.html
[7] JSR-000118 Mobile Information Device Profile2.0 (Final Release). Retrieved from http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html
[8] JSR-000271 Mobile Information Device Profile 3. Retrieved from https://jcp.org/aboutJava/communityprocess/final/jsr271/index.html
[9] J2ME Record Management by Soma Ghosh (sghosh@entigo.com), Application Developer, Entigoretrived on 28-11-2009 http://www.ibm.com/developerworks/library/wi-rms/#2
[10] Core System Architecture. Retrieved from http://www.bluetooth.com/Bluetooth/Technology/Works/Core_System_Architecture.htm
[11] Overview of Operation of Bluetooth Technology. Retrieved from http://www.bluetooth.com/Bluetooth/Technology/Works/Overview_of_Operation.htm