A User-friendly Mobile Application to Promote Medication Adherence

J.X. Tan, S. Chan, C.T. Lau

***Abstract* — In this paper, we present a user-friendly mobile application that automatically generates alarm signals to remind a user to take medication. This application is able to automatically process a prescription of multiple medications and provide a visual reminder, as well as an audio reminder in the user's chosen language/dialect. This application can help to promote medication adherence among elderly patients.**

Index Terms – medication adherence, mobile healthcare.

I. Introduction

Mobile computing systems have been increasingly adopted by the healthcare industry. Such systems typically provide greater convenience and better accessibility to both patients and healthcare service providers, often resulting in substantial cost and time savings. With more people owning and using mobile devices, more mobile healthcare applications will be embraced by the general population, especially those perceived to be useful and easy to use [1]. One of the major challenges faced by healthcare providers is poor medication adherence. This problem affected both males and females of all ages. Many of them are not taking their medications as directed [2]. As elderly patients are more likely to be afflicted with chronic diseases, they need to take more prescription medicines and poor medication adherence would lead to serious consequences. Some common factors affecting medication adherence have been identified. The patient may be forgetful, or he may find it difficult to manage multiple medications. A possible strategy to overcome this problem is to make use of pill boxes or reminder packaging.

There are existing devices that can provide reminder services for medication. The Automated Pill Dispenser by MedMinder [3] is a rectangular tray with multiple compartments for storing medications for each day over a week. It is equipped with wireless technology and can be programmed to send visual and audio reminder. This tray is very useful in a home environment but is relatively bulky to be carried around by the user to different places. Since many people own mobile devices nowadays, a mobile application installed on a mobile phone can provide the reminder service without requiring additional cost or equipment.

Several mobile applications for medication reminder have been made available [4-8]. These are examples of Android applications that help users to manage their prescriptions and provide reminders. Typically, this type of applications allows the user to set the time at which the reminder is to be triggered. The reminder is usually in the form of an audible alarm. Some applications also provide a visual reminder. Table I shows the features commonly found on such applications.

TABLE I.
Comparison of Existing Applications.

Reminder Application	Features			
	Snooze function	Interval auto-calculation	Visual reminder	Text reminder
Pillbox Alert [4]	**Yes**	**No**	**No**	**No**
Meds Reminder [5]	**Yes**	**No**	**No**	**No**
Pill Reminder [6]	**No**	**Yes**	**No**	**Yes**
MedBuddy [7]	**No**	**Yes**	**No**	**No**
Medication Reminder Widget [8]	**Yes**	**No**	**Yes**	**No**

However, none of these applications provides voice reminder in a language/dialect chosen by the user. This feature is particularly useful for promoting medication adherence among elderly users who are not familiar with the default language, such as English.

II. System Design

A. Overview

This project aims to develop a healthcare mobile application that provides reminder services to users so that they can adhere to their medication regime. The user creates a new medication schedule by setting a schedule name and specifying the date and time for the first medication notification. Once a schedule is created, additional medication details, such as medicine name, dosage, picture and audio reminder, can be added into the schedule.

Once the schedule and the medication details have been created, the user will be directed to the view schedule list. While viewing the schedule list, the user can choose to activate the schedule reminder immediately or at a later time. Upon activation of a schedule reminder, the application will be sent to the background. When it is time

Manuscript received November 27, 2012.

J.X. Tan, S. Chan and C.T. Lau are with the School of Computer Engineering, Nanyang Technological University, Singapore. (phone: 65-67905047; e-mail: {y090086, asschan, asctlau@ntu.edu.sg).

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

for the medication alert, the application will play an audio reminder (in the dialect chosen by the user). The application will also display the medication details which include medicine name, dosage and medicine photo in both the notification and main screen of the mobile phone. The overview system operation is shown in Figure 1.

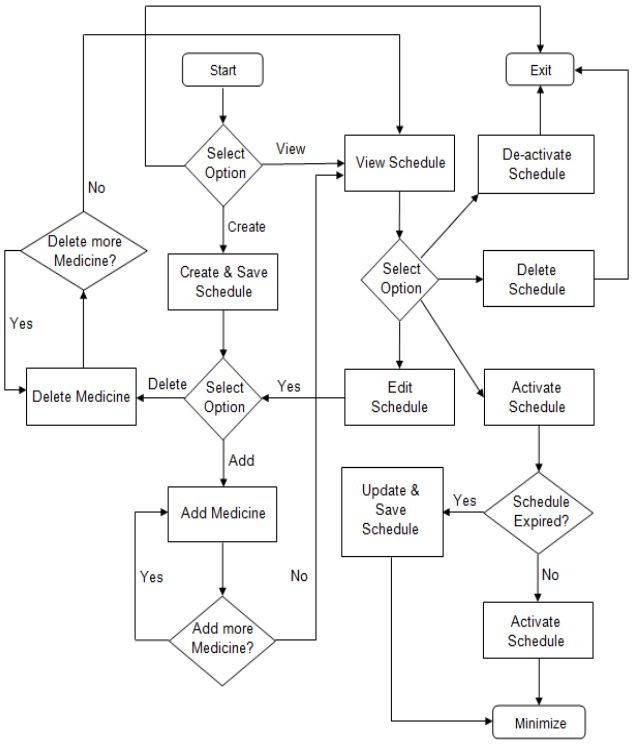

Fig. 1. System Flowchart.

B. Graphical User Interface

The medication reminder application allows the user to create individual or multiple schedules via a graphical user interface (GUI). In order to create a schedule, the user needs to specify a name for the schedule as well as the date and time to start the alert reminder. Once the schedule is created, the user is directed to a menu where the user can either add or delete medicines from the schedule.

To add a medicine, the user simply enters the details of the medicine which include the medicine name, consumption interval, dosage, dosage type (tablet/ml) and dialect-audio (for alert sound). If the medicine added is an antibiotic, a checkbox which indicates the course needs to be completed will be checked automatically. There is also a camera function that allows the user to take a picture of the medicine to be used for visual alert.

The user can select and delete a medicine from the list of medicines that is previously added into the schedule. Deleted medicines are removed from the schedule.

After adding the required medicines, the user is directed to the view schedule page. In this page, the user is given four choices (activate/de-activate/edit/delete). The user can activate the schedule and the alert will start according to the user-defined start time. If the user wishes to delete the schedule while it is activated, the user will need to de-activate it first. A de-activated schedule will be marked as "Expired". If the user wishes to activate an expired schedule, he will be directed back to the create schedule page. The user will need to update the date and time for the schedule reminder. The user can also edit the schedule by adding or deleting medicines from it and is able to view the created schedule at any time.

C. Data Storage

In the application, persistent storage is important as users need to save their input information on the schedule reminder. The information can be stored as plain text files but it may not be practical for the application as it is complex to model a rational database using plain text format. Hence, a database is designed for the application to provide persistent storage. Figure 2 illustrates the design of the database as well as the relationships between the database entities.

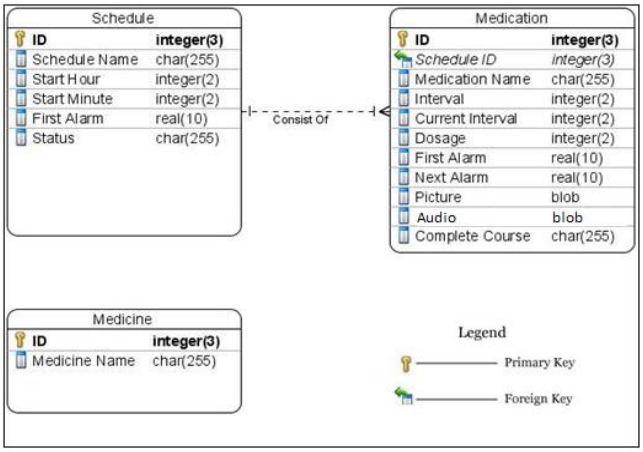

Fig. 2. Database Diagram.

The database consists of 3 entity tables. The schedule table houses all the information relating to a schedule created by the user. The medication table is used to store all the information on a medicine added by the user to a schedule. These two tables are related through a Primary/Foreign key pair. The third table is the medicine table, which is used to store a list of medicines added by the user.

III. System Implementation

Android is a software platform and Linux-based operating system (OS) designed primarily for touch-screen mobile devices. Its OS is upgradeable and new features are usually available in each new version release. In addition, Android is open-source which is freely available for application developers. The increasing demand of Android devices makes it a great platform to leverage on.

This medicine reminder mobile application has been designed to run on the Android platform. It has been

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

deployed and tested successfully on 3 mobile devices: HTC Sensation, Samsung Ace and Galaxy S.

A. GUI Implementation

The user interface (UI) is created in XML format instead of the conventional Java class. It is separated from the actual program logic which ensures that any changes in the program logic will not affect the UI or vice versa. Each UI corresponds to a Java file that will process the information captured on the UI.

The UI can be created by using the palette provided by Eclipse Development Environment. The palette allows developers to create UI by dragging and dropping the components into the created layout and the corresponding XML codes will be generated.

An Android Activity is one of the basic building blocks for creating Android applications. It is an application component that provides a screen or UI with which users can interact with. Each UI file created will have a Java class that is extended from the Activity class.

There is a resource class called R.java which contains all the reference to the UI XML file and UI component created for the application. The code below is used to obtain reference and display the UI on the phone's display.

```
public class MedicationReminder extends Activity {
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      //display UI on phone's screen
            setContentView(R.layout.main);
}
```

However, the reference alone will not display the UI on the phone's screen. We also need to declare the activity in the manifest file in order for it to be accessible to the system.

To obtain the user's input, the Activity class needs to get the reference of a specific UI component. The code below shows a reference to a textbox and the method to obtain the value in the textbox.

```
//obtain reference to textbox
EditText intervalNo = (EditText) findViewById(R.id.Interval);

//get user input from textbox and convert to int
int interval = Integer.parseInt(intervalNo.getText().toString());
```

B. Database and Imaging Implementation

The database is implemented using SQLite database [9] which is supported by Android platform. It is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. Thus SQLite libraries are used to link to the application, forming an integral part of the application. In order to create a SQLite database and methods, Java codes are used as shown below.

```
//define table column
String DATABASE_CREATE = "CREATE TABLE Schedule("
        +"_id INTEGER PRIMARY KEY AUTOINCREMENT,"
        +"scheduleName TEXT NOT NULL,"
        +"startHour INTEGER NOT NULL,"
        +"startMinute INTEGER NOT NULL,"
        +"firstAlarm REAL NOT NULL,"
        +"status TEXT NOT NULL,"
        +"created TEXT NOT NULL);";
//create table
db.execSQL(DATABASE_CREATE);
//create insert method
public long insertMedication(int scheduleIDIn,
            String medNameIn,
            int intervalIn, String dosageIn, long firstAlarmIn,
            long nextAlarmIn, long duration, Bitmap imgIn,
            boolean comleteMed) {
            …}
```

The codes are defined in a Java class as well as the methods in order to access the query and insert data into the database. The database is created only once when the application is first installed on a mobile device. Once the database is created, methods can be called to interact with the database.

For imaging, the application makes use of the built-in camera on the mobile device to take pictures of the medicines for visual reminder aid. The image captured is temporary saved in the SD card storage before it is transferred to the database. In order to invoke the built-in camera function, an intent is created and a trigger is needed to start the camera function. For the application, the camera will start to function when the user presses the "Take picture" button provided in the GUI. Therefore, the button needs to add a listener to detect the press effect and perform the required action. The codes below illustrate how a listener is added to a button as well as the creation of intent.

```
//add listener to camera btn
cameraBtn.setOnClickListener(new OnClickListener(){
            public void onClick(View arg0) {

//call the camera intent
Intent intent =
        new Intent("android.media.action.IMAGE_CAPTURE");

//start camera activity
startActivityForResult(intent,PICK_FROM_CAMERA);
}
```

Once the camera function has started, the user can either take a picture or cancel the action. A method corresponding to the startActivityForResult() is needed to determine the action taken by the user and what actions need to be done by the application. The onActivityResult() method is used to determine if the user has taken a picture. In the case where the user has taken a picture, it will be displayed on the screen; otherwise a default image will be displayed.

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

```
public void onActivityResult(int requestCode, int resultCode,
                             Intent data) {
//take results from camera
if(resultCode == Activity.RESULT_OK && requestCode == 0){
        //display taken image
        onPhotoTaken();
}
else if(resultCode == Activity.RESULT_CANCELED &&
                        requestCode == 0){
        //display default image
        setDefaultPic();
}
```

C. Schedule Reminder Implementation

This section covers the core of the application which provides timely reminders to the user. The reminder is implemented using the Alarm Manager and Notification Manager available in Android platform.

The alarm manager allows the application to access the system alarm services which can be used to schedule the application to be run at some point in the future. In order for the alarm to function appropriately, a broadcast receiver must be defined. The broadcast receiver is created as a separate Java class as shown below.

```
public class AlarmReceiver extends BroadcastReceiver {
 //execute the necessary action when receive broadcast
 Public void onReceive(Context con, Intent in) {
        ...
 } //end of onReceive
} //end of BroadcastReceiver
```

Once the alarm time has arrived, a broadcast is sent out by the system to which the broadcast receiver will answer and execute the actions as defined in the Broadcast receiver class. However, to ensure that broadcast receiver works, we need to declare the receiver in the manifest file in order for it to be accessible to the system. The code below illustrates how the receiver is declared in a manifest file.

```
<manifest ... >
  <application ... >
    <activity android:name = ".MedicationReminder"/>
    <receiver android:name=".AlarmReceiver"
              android:process=":remote"/>
  </application... >
</manifest>
```

In addition, an intent is needed to point to the broadcast receiver and also a reference to the system alarm service must be obtained to set the alarm. This can be done by using a PendingIntent as it identifies the application that is accessing the system alarm. This is essential as there may be other applications that need to use the system alarm as well. Therefore, as shown in the following codes, the alarm.set() method is used to register the alarm with the alarm manager as well as the time of broadcast. The time is presented as milliseconds.

```
//point to broadcast
Intent intent = new Intent(ViewSchedule.this,
                AlarmReceiver.class);
PendingIntent pendingIntent = PendingIntent.getBroadcast(
                ViewSchedule.this, inID, in,
                PendingIntent.FLAG_CANCEL_CURRENT);
//obtain an instance of the alarm service
AlarmManager alarm = (AlarmManager)
getSystemService(ALARM_SERVICE);
alarm.set(AlarmManager.RTC_WAKEUP, alarmTime,
pendingIntent);
```

The Notification alerts the user when it is time to take a medication. It can be programmed to display an icon on the status bar on the device screen. In order to do so, a reference to the system's notification needs to be created as follows.

```
NotificationManager manager = (NotificationManager) context.
      getSystemService(Context.NOTIFICATION_SERVICE);

//creating a notification
Notification notification = new Notification(R.drawable.alert,
"New Medication Alert!", System.currentTimeMillis());

//display notification
manager.notify(alarmID, notification);
```

The notification can be configured by using the flags provided as illustrated in the codes below. It can be used to define the sound (default/custom) to be played when a notification is displayed. For the project implementation, the notification is set to play a custom dialect-audio defined by the user while creating a schedule.

```
//set notification flags
notification.flags |= Notification.FLAG_AUTO_CANCEL;
notification.flags |= Notification.FLAG_INSISTENT;

//select the correct audio to be played
if (aud==1)notification.sound=
   Uri.parse("file:///sdcard/DOWNLOAD/chinese.mp3");
else if (aud==2)notification.sound=
   Uri.parse("file:///sdcard/DOWNLOAD/cantonese.mp3");
else if (aud==3)notification.sound=
   Uri.parse("file:///sdcard/DOWNLOAD/hokkien.mp3");
else notification.sound=
   Uri.parse("file:///sdcard/DOWNLOAD/english.mp3");
```

Once the alert is activated, it will notify the user via the dialect-audio selected as well as a pop-up message on the device main screen displaying a picture, name and dosage of the medicine to be taken. The dialect-audio will continue to play until the user acknowledges the reminder which clears the notification. The message will then proceed to display the information of the next medicine if the schedule includes multiple medicines.

Whenever the alert is activated, the user receives a reminder for his medication where he needs to

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

acknowledge the reminder. Once acknowledged, the application automatically calculates and determines the next medication time for each medication in the schedule.

IV. Results and Discussion

An evaluation has been carried out to get feedback on various aspects of the application such as the GUI design and the ease of use. Twenty users comprising 5 young adults, 5 middle-aged adults and 10 elderly users participated in the evaluation. Their feedbacks are recorded as shown in Table II.

The users are generally satisfied with the application in all aspects. It is noted that younger users are more interested in the pop-up message (80%) whereas elderly users are in favor of the dialect-audio reminder.

TABLE II.
Survey Results.

Aspects	Age Group	Rating 5-Highest, 1-lowest				
		1	2	3	4	5
GUI design	18-30	0%	0%	20%	60%	20%
	31-59	0%	0%	20%	80%	0%
	>60	0%	0%	20%	60%	20%
Ease of navigation	18-30	0%	0%	0%	20%	80%
	31-59	0%	0%	0%	40%	60%
	>60	0%	0%	20%	60%	20%
Ease of creating and setting up the schedule reminder	18-30	0%	0%	0%	20%	80%
	31-59	0%	0%	0%	40%	60%
	>60	0%	0%	40%	60%	0%
Clear instructions provided	18-30	0%	0%	20%	40%	40%
	31-59	0%	0%	20%	60%	20%
	>60	0%	0%	40%	60%	0%
Which part of the application do you like most?		Pop-up message with picture & prescription		Dialect-Audio		
	18-30	**80%**		20%		
	31-59	40%		60%		
	>60	20%		**80%**		
Comments	18-30	▪ Dialect-audio is interesting ▪ Picture message displayed on the screen when reminder sound off is helpful ▪ Very easy to use				
	31-59	▪ Dialect-audio is very helpful ▪ Pop-up message during alert is very useful ▪ It will be good if there is tutorial to guide first timer user				
	>60	▪ It will be better if the font size of the word is bigger ▪ Dialect-audio is very helpful ▪ Pop-up picture message is very helpful				

V. Conclusion

We have demonstrated a user-friendly mobile application that automatically activates alarm signals to remind a user to take medication. Our prototype application is also able to automatically process a prescription of multiple medications and provide a visual reminder, as well as an audio reminder in the user's chosen language/dialect. Future work will include incorporating healthcare measurements such as blood glucose, blood pressure, weight and other medical data.

References

[1] J.H. Wu, S.C. Wang, and L.M. Lin, "Mobile Computing Acceptance in the Healthcare Industry: A Structural Equation Model," *International Journal of Medical Informatics*, vol. 76, pp. 66-77, 2007.

[2] B. Kocurek, "Promoting Medication Adherence in Older Adults and the Rest of Us," *Diabetes Spectrum*, vol. 22 no. 2, pp. 80-84, 2009.

[3] MedMinder, *Automated Pill Dispenser*, www.medminder.com/products/maya-pill-dispenser

[4] Sartuga Software LLC, 2010. *Pillbox Alert*, www.pillboxalert.com

[5] Google Androidzoom, 2011. *Meds Reminder*, www.androidzoom.com/android_applications/medical/meds-reminder_cbewm.html

[6] Google Shop Android Apps, 2012. *Pill Reminder*, play.google.com/store/apps/details?id=com.garland.medminder&hl=en

[7] Appbrain, 2012. *MedBuddy*, www.appbrain.com/app/medbuddy/com.rmcomputing.med

[8] Appbrain, 2012. *Medication Reminder Widget*, www.appbrain.com/app/medication-reminder-widget/com.nio.medicationreminder

[9] SQLite database, www.sqlite.org

ISBN: 978-988-19252-6-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)