International Journal of Information & Computation Technology.
ISSN 0974-2239 Volume 4, Number 9 (2014), pp. 891-896

http://www. irphouse.com

Quality Assurance in Localization

Aparna Sharma[1]and Manisha Bhatia[2]

[1]*Computer Science, BanasthaliVidyapith, C-Scheme, Jaipur, INDIA*
[2]*Computer Science, BanasthaliVidyapith, C-Scheme, Jaipur, INDIA*

Abstract

This paper presents a Quality Assurance model for localized application which has been implemented as an android based application. The model performs the terminology testing (which is to test the terms used in the android application) and Testing results in a number of suggestions of the words/terms that can also be used in the current context. The purpose of this paper is to present the control functionality of the quality tester module named 'Localizer" i.e. the flow of control, how the localized application sends the terms as an attachment to the broadcast message which in turn responded and completed by the Localizer. The Localizer in turn accesses the WebAPI for the corresponding *terms* and *fetches* the different suggestions. Also the quality tester module runs as a background service which means testing the quality for terms will not affect the current functionality.
Keywords: WebAPI;Sandbox;Localizer;Intent;localize_intent;
Broadcast Receiver; OnReceive;AndroidManifest;SQLite;Threads

1. Introduction

In android as every application runs in its own "sandbox", a running application (cannot communicate directly) must communicate with other installed applications on device either through intents or through broadcast message.*Intents* are to activate a component in the application and *Broadcast messages* are implemented using the BroadcastReceiver class of Android that sends broadcast messages across applications.

The tester module named "Localizer" uses the concept of Broadcastmessages. Any Localized application which wants to profile the terms used in the localized android app specifies its action as "LOCALIZE_INTENT" which is a custom action

that can only be recognized and received by the Localizer quality tester. The tester module analyses the terms attached with the broadcast message, compare them with Web based API to identify its correctness and as a result it shows a number of suggestions regarding the terms that can be used in the same context.

2. Functionality and Control Flow

Testing the quality of a localized android application corresponding to the *terms* being used is accomplished by functioning of various components at different stages. Figure 1 shows the complete functionality of Localizer as a combined outcome of functions of different components.

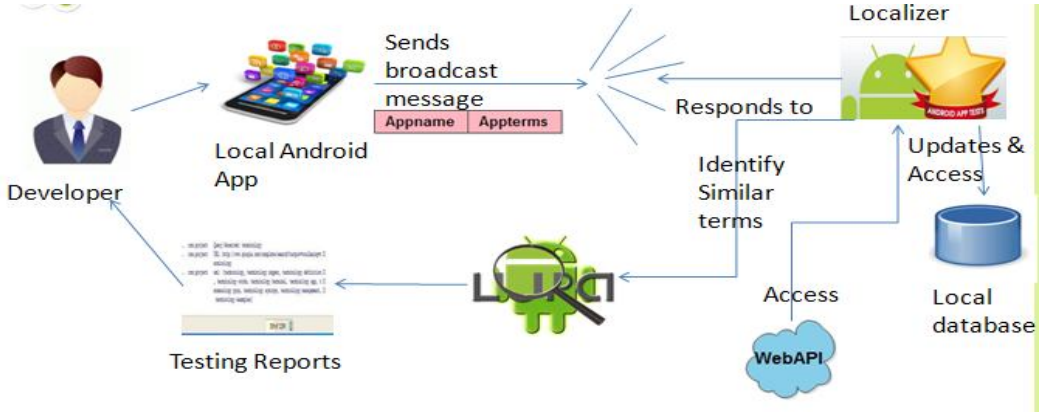

Figure 1. The overall modular functionality of the Localizer

2.1 Developer

The Developer can access Localizer in one of two ways:-

2.2.1 Through sending broadcast message: Whenever the developer wants to tests its own local android application he/she will send a broadcast message together attached with its own *Application name* and the String array of the *Application terms* that need to be tested.

```
String[] terms={"thanks","thanku","thanks giving"};
Intent intent = new Intent();
PackageManager packageManager = this.getPackageManager();
ApplicationInfo applicationInfo = null;
try {
      applicationInfo = packageManager.getApplicationInfo
(getPackageName(), 0);
} catch (final NameNotFoundException e) {}
final String title = (String)((applicationInfo != null) ?
packageManager.getApplicationLabel(applicationInfo) : "???");
intent.setAction("com.gunvatta.LOCALIZE_INTENT");
intent.putExtra("Appname", title);
intent.putExtra("Appterms", terms);
        getApplicationContext().sendBroadcast(intent);
```

The tester module Localizer implements the class BroadcastReceiver class of android so that it can receive any broadcast message when its specified action in the AndroidManifest file matches with the intent action of the broadcasted message.

```
public class LocalizeReceiver extends BroadcastReceiver {

@Override
public void onReceive(Context context, Intent intent) {
      Intent service = new Intent(context,SaveService.class);
      service.putExtra("Appname", intent.getExtras
().getCharSequence("Appname"));
      service.putExtra("Appterms",intent.getStringArrayExtra
("Appterms"));
      context.startService(service);

}
```

```
<intent-filter>
      <action
android:name="com.gunvatta.LOCALIZE_INTENT"></action>
</intent-filter>
```

A service that works in background is being started as soon as the application receives the broadcasted message.Localizer updates its local SqLite database with the new *terms* and *application*received, by implementing the *onReceive* method of *BroadcastReceiver* class.

In the service activity of the android application the local database is updated with the new data received.

```
protected void onHandleIntent(Intent intent) {
   System.out.println("service started ");
   String [] terms = intent.getStringArrayExtra("Appterms");
   SQLiteDatabase db = openOrCreateDatabase("Localdb", MODE_PRIVATE, null);
   db.execSQL("CREATE TABLE IF NOT EXISTS Application (Appname VARCHAR,
Appterms   VARCHAR);");
   for (int r=0;r<terms.length;r++){
   db.execSQL("INSERT INTO Application VALUES('" +      intent.getExtras
().getCharSequence("Appname") +"','" + terms[r] + "');");
   }
   db.close();
```

2.2.2 Directly executing Localizer as an application:

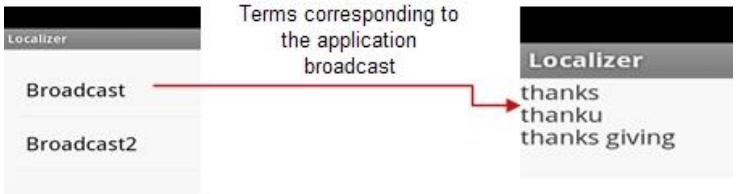

Figure 2. Application terms corresponding to their name retrieved from local DB

As shown in Figure 2 whenever the developer access the Localizer it retrieves the information of Application local database (i.e. the terms and name) and matches with the google WebAPI tofind out the corresponding similar terms with minor variations that can also be used in the same context. Google WebAPI responds to the suggestions in the form of XML as shown in Figure 3.

```
a.l...  com.project   Query Received: thanks
a.l...  com.project   URL: http://www.google.com/complete/search?output=toolbar&q=t
                      hanks
a.l...  com.project   xml: [thanksgiving 2013, thanksgiving, thanksgiving recipes,
                      thanksgiving menu, thanksgiving side dishes, thanks for shari
                      ng, thanksgiving desserts, thanksgiving coloring pages, thank
                      sgiving appetizers, thanksgiving crafts]
```

Figure 3. Suggestions fetched from WebAPI corresponding to the term “thanks”

3. Conclusion

Terminology testing of a local android application is of major importance as it evaluates the effectiveness and acceptance of a local product.The implemented quality tester module checks for the “terms” of the local application sent through broadcast message with WebAPI and results in a number of suggestions that can fit in the current context, Also it works as a *generic quality model* that can test any android based application because the developer itself need to send the profiling terms with the broadcast message.As it checks for each and every *term* in the application it can also work as a spell checker or to identify the grammatical violations in the application. As shown in Figure 4 the model also works when two or more applications broadcasts messages parallely, because it retrieves and completes the requests through threads.

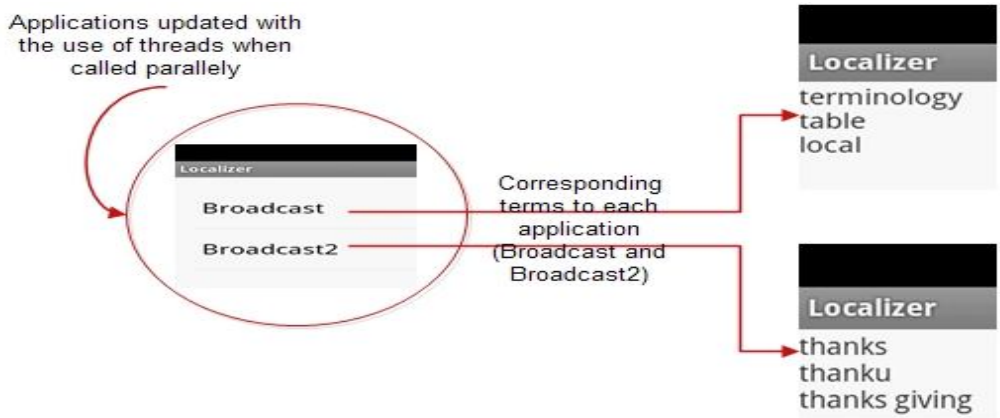

Figure 4. Using threads when two application sends broadcast message in parallel

So, when adopted for a local product(like when a developer need to profile his/her app) it may refer directly to the “Localizer” which in turn retrieves the

suggestions as a result of comparing them with WebAPI and returns back the suggestion to the caller of the application.

In addition to all these factors as per the *software localization practices* one more important aspect is that whether the end localized product follows the *cultural conventions* of the locale or not, and if it does then till what level it is being satisfied. So, such factors must be separately considered to achieve the higherquality and maximum satisfaction as an end result.

References:

[1] Manisha Bhatia, Aparna Sharma (28thOct 2013), Article in a regular Journal, Research Inventy International Journal of Engineering and Science, Control, 3, 10, pp, 37, 46

[2] Manisha Bhatia, Aparna Sharma, Varsha Tomar (30th Sep 2013), Article in regular journal, Inter. J. SR in Computer Science Applications and Management Studies, Control,2,5

[3] Android App Development with Java Essential Training (2013), Online Video tutorial, http://Lynda.com

[4] K.K. Aggarwal & Yogesh Singh (2007), 'Software engineering (3rd Ed), New Age International Publishers

[5] Roger S. Pressman (2000), 'Software Engineering (5th ed.), McGraw-Hill Higher Education Publishers