Water Community: A Web Based Android Application for Sharing and Updating Received Water Issue Complaints

Dinithi N. Sumanaweera
Department of Computer Science & Engineering, University of Moratuwa
Sri Lanka
dinithi.10@cse.mrt.ac.lk

***Abstract* - In this paper, the project on developing a web based android application named 'WaterCommunity' is being discussed in terms of the need for such application, the nature of the application, which technologies and tools were being used, along with how the overall development process was carried out. WaterCommunity application lets the members of a typical water community to share, notify and update details about water issue complaints they receive from the general public. A member can assign another member as responsible for a certain complaint and send him/her a notification. This software application gives the ability for such community to have efficiency and supervision on their routine work processes as well. The project was done according to the RUP software development process with two iterations for three months and finally an initial version was obtained as a successful outcome.**

Index Terms – Water Community, Android, SOAP Web services, Google Maps, GCM

I. Introduction

A typical water community can be considered as a group of people who contribute to the well-being of human beings by providing safe and clean water on a daily basis. In reality, this can be a nationwide organization which involves in water supply and drainage of a particular country. Such water service supplier engages in the activities related to a national water supply network by carrying out operation and maintenance of different water supply schemes. In general, the members of such a community are the engineers, lab technicians etc. and they communicate very frequently with each other regarding the matters related to the water industry. Therefore, it is much important for them to have some kind of a proper mobile communication platform for sharing information. 'WaterCommunity' is an android application for fulfilling that purpose.

Today, the majority of the people carry smartphones around with them, with its operating system being Android and for so many purposes. They make their lives much easier in terms of communicating with other people through the provided social networking platforms via the existing Android applications. The same concept can be used for the work related communication which in turn can result in efficiency as well as smoothness in the routine work processes. There are such applications that are already being used to maintain the work flow of a particularly targeted type of organizations. Here, it has been applied to a water community.

'WaterCommunity' is a web based Android application for the members of a water community to share and update information among each other. This was initially developed as a basic application which can enable them to address the public water issue complaints in an efficient manner. This research paper discusses the need for such an application, how it was designed, implemented and how the developed application solution makes a water community members' life so much easier than before.

II. Background Study

In a country with wide water supply and drainage coverage, it is highly probable for different types of issues to occur with time. Things can go wrong due to water leaks, pipe bursts, water quality issues, low water pressure circumstances and other connection related issues including illegal connections [6][7]. These issues have a direct impact on the proper existence of a water supply network, creating so many problems for the general public.

People expect for their essential requirement of water resources to get fulfilled by a flawless water supply and drainage system. A nation-wide water supply and drainage organization should ensure the water safety, cleanliness and non-disturbance of their water supply and drainage schemes. Therefore it is very important for them to be alert all the time by having a continuous inspection and observation over the entire system. In that way, they have to make sure that any inconvenience or an interruption has not been occurred to the general public.

If an issue gets created in the system, it is usually the general public who encounter the problem first. The normal way to make a complaint about it to the organization is via telephone calls, emails, web portals or in person [1][2][3][4]. Especially for a developing country, the most viable way is to complain over the phone. But through all of those methods, the complaint might not be directed to the relevant responsible person of the water community. In that case, another member will have to inform him/her with all the necessary information. Thus, it would take a considerable amount of time for a complaint to get processed and be addressed. Sometimes due to miscommunication, it might be difficult for the complaints to be tracked and addressed in a proper manner. Since the whole procedure is somewhat long, it will not help the members to immediately address those issues and provide solutions. Even though there are online issue tracking and status updating services provided by some existing water organizations for the same purpose, it might still slow down the whole process as the members are required to login to a web system and then work on the related matter [5]. Such scenarios may lead to inefficiency in the normal work flow of community members as well as a deterioration of the organizational reputation. General public will no longer have trust and confidence in them. The nature of such consequences emphasizes the need of a well-mannered coordination required for gathering and addressing water issue complaints.

If there is a proper mobile communication platform to immediately redirect a received water complaint to the relevant responsible community member with related details and then have supervision on it until the situation gets handled, a water community will be able to establish a better relationship with the general public as well as with their own members. Today, since almost any water community member has an android smart phone with him/her, why not take that as an advantage for fulfilling the earlier said purpose.

III. PROPOSED SOLUTION

'WaterCommunity' android application perfectly fits in as a method to address issue complaints received by any of the members in a nation-wide water supply and drainage organization. The major feature of this software application is a map of the country which can be used to share and display information.

Firstly, a community member can join the WaterCommunity service by getting registered. Then he can login to the application. Each member of the community gets to see a complete water issue related complaint map of the country and also the details of each complaint including its status. When a member receives a complaint on a water issue from the public, not only can he use this application to immediately mark the complaint with relevant details on a map in terms of the correct location for the issue, but also to send an immediate notification to the responsible person so that he can take actions on time. That particular person will be able to keep the others informed of the issue status; whether it is being addressed or an action has already been taken, by updating that information. This will lead to an indirect supervision over the issues, each being resolved by a relevant responsible person. Such a visible supervision can be a direct influence on the member to address the issue at the earliest possible. Therefore the overall efficiency in handling, managing and resolving complaints starts to get increased.

Another functionality which the application can provide is to get a list of all members in the organization with their contact details. This is a good way when one member needs to make the responsible person aware of an issue situation in case of an emergency and even to contact for other work related purposes.

IV. DESIGN

WaterCommunity is a web based Android application that has two major components; an Android client application and a web service. Therefore the high level architecture is a client server architecture which can be shown as in Figure 1.

A. *High level architecture*

As previously mentioned, the major functionality of this application is to share a received complaint with details on a country's map by its location and assign a responsible member through sending a notification. Thus the application uses Google Maps service for getting the map viewing facility and Google Cloud Messaging (GCM) service for sending out a notification.

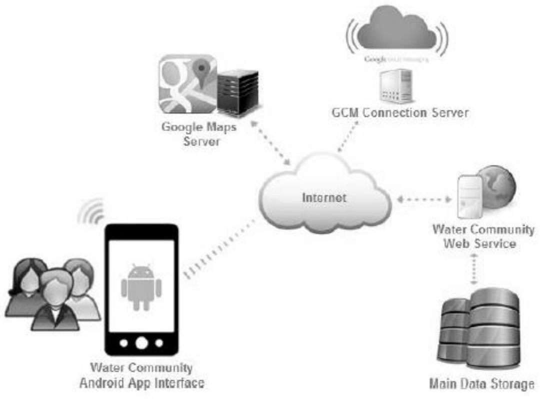
Google Maps Server
GCM Connection Server
Internet
Water Community Web Service
Water Community Android App Interface
Main Data Storage

Figure.1 High Level Architecture

The client side includes the Android application which presents the front end interface to the end users. The server side includes the web service containing the business logic and the database where all the application data is stored. When the user makes a request through the android application in order to login, share & notify about complaints or view details, the client side requests from the web service and then the web service accesses the backend database in order to do relevant data retrievals or modifications. The client has to connect with the Google Maps server in order to display the map. Also the web service and client android device both have to get connected to the GCM server in order to obtain the notification functionality. Moreover, the android device has to be registered for the GCM service at first, in order to receive notifications from the web service.

All these can be considered as external interactions of the system with external server entities.

B.4 Tier architecture

The overall system architecture can be presented as a four tier layered architecture. Having a layered architecture gives scalability, modularity, extendibility, reusability and a better secure flow control within the system. It may help carrying out an efficient system development as well.

1) Presentation Logic Tier

This tier includes the logic to implement the GUI for the end user and also the distributed logic required for connecting with the back end web service in order to send requests and receive data. The sub tier called proxy tier deals with Simple Object Access Protocol (SOAP) to communicate with the web service. In this system architecture, Web Service Caller layer belongs to the Proxy Tier.

2) Business Logic Tier

This tier includes the logic to implement main business functionalities.

3) Data Access Tier

This tier includes the logic for interfacing with the database to access, store and retrieve application data in the database.

4) Data Tier

This tier includes the database; the main application storage of the system along with the database management system.

In designing the packages, client and web service server were recognized as two different entities inside the same application. The web service component has the three package components; WaterCommunity Web Service component, Login/Register service and Issue Monitoring service. Client side component has five components; Login/Register, Map View, Member List View, Water Issue Handler and the Web service caller. Water Issue Handler component gets connected with GCM remote server component. Map View component makes remote procedure (RPC) calls to Google Maps API at a remote Google Maps server.

V. Implementation

The development was done according to the generic Rational Unified Process with two iterations. During that period, an online blog was maintained as a project log to keep track of daily work. At the initial two phases; Inception and Elaboration, the required documentation including feasibility study, vision, development case, requirements specification, architecture document and the quality assurance plan were done according to RUP documentation standards. The project schedule was prepared using OpenProj project management tool. Also proper research was done in order to gather information as mentioned earlier. The plan was to complete the development task within three months' time.

A. *Technologies and Development tool*

The IDE for the WaterCommunity client component was Android Developer Tools; the Eclipses' special platform version known for Android development purpose. Java compiler compliance level of 1.6 was used throughout the implementation. The targeted platform was specified as Android 4.2.2. For executing the Android client side, both Android emulator and an Android device with 4.0.4 Android version were being used throughout the development process.

For the web service development, Netbeans was used as the IDE. The main reason for using two different IDEs for the client and server sides was to clearly make the separation and the independence of each other so that it can represent the real world function. After going through some search about an ideal open source web service stack, the Metro web service was selected and it was deployed inside Apache Tomcat web container. The remote Android client was made to consume the web service by accessing the service methods using SOAP.—Ksoap2-android-assembly-2.6.0-jar-with-dependencies.jar library was used for that purpose.

Before starting the implementation of essential methods, the communication between the Android client and the remote web service was tested and got confirmed. Since both of the components reside in the same development machine, the web service was at localhost and the Android client was got connected to localhost either through the emulator or through networking the Android device with the machine. The backend database was a MySQL database and it was deployed at the Wamp Server. Metro web service application connects to the database via the JDBC driver.

After ensuring that the communication between client side, server side and the backend database happens properly, the next task was to implement the methods directly required for main functionalities in the application. The GUI components were developed while giving the importance to the look and feel standards. All other activities and services were implemented according to the earlier prepared software architecture document, but with some changes in the design that were found to be required later through research. At the Android client side, all presentation logic was separated from the main GUI thread by implementing separate threads using Asynchronous Tasks (AsyncTask) for each logic implementation.

B. User interface design and implementation

During the implementation, there was a considerable amount of concern over GUI designing. Since the user friendly nature is an essential aspect of any application; especially for a smartphone application, the UI components were carefully designed. The home screen was implemented using three tabs; Home, Map View and Members which simply represents the application functionality. For the purpose of providing information, dialog box messages were used. When deciding on the colour schemes, a special concern was on the theme colour blue; the appropriate colour for any Water Community. Popping up of necessary status messages at each user action was an important consideration too.

C. Map view implementation

The main feature of this application is the map view where every member gets to see all issue information around the country. The responsible members can keep the others updated of the issues by modifying statuses and details. Therefore it is essential to update the map in a frequent manner. For this purpose, another asynchronous task was implemented to retrieve map information from time to time to get updated. It was implemented as a background thread.

D. Notification functionality implementation

The most challenging task was to implement the GCM notification functionality. In here, the Android device needs to get registered to the GCM service by communicating with a remote GCM server. After that, when a member gets assigned as responsible for addressing a complaint, he should be notified by the web service through sending a request to a remote GCM service with his devices' GCM registration number and the message so that, the remote server can send the GCM notification to the specified device. The GCM for Android library was used for this purpose. Currently Google has presented a new GCM API and had stated that the earlier library has been deprecated. However, since it seemed much difficult for the server side at Netbeans to be implemented using the new API, it was decided to go with the old library instead. At the server side, Json_simple-1.1.jar library had to be used in order to make the web service communicate with the remote GCM server using RPC calls. After that, the required code implementation was started.

E. User session management

Another aspect was to implement session management in order to maintain a logged in session between the android client and web service. This was done using Shared preferences in Android and a separate Session Manager class was implemented for the purpose. However, checking the session ID at web service before fulfilling client requests per single login session could not be implemented. Therefore, this aspect was not fully completed.

All these implementations were carried out while going through various online developer tutorials and applying/reusing relevant concepts/code segments as well.

F. Testing

During the software development process, client side unit testing was done using a special modified version of JUnit test automation framework known as Robotium (robotium-solo-4.3.jar). This framework gives the ability to automate user actions. Therefore it was used for testing the interactions between activities when the user does actions through the user interfaces. A separate testing project was created in order to do so. For the server side, JUnit was used inside the same project.

VI. RESULTS

The final outcome after two iterations of the development process was an initial version of the WaterCommunity Android Application. After installing and launching the application on the Android device, the user gets to see the login screen as shown in Figure 2.

If the user is already a member, he can just enter the username, password and login to the service. Otherwise, he can click on 'register here' link and go to the registration screen where he can enter his first name, last name, designation, NIC, email, password as details and hit the register button. Then the user

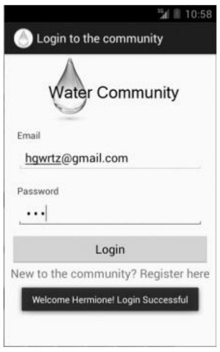

Figure.2 Login Screen

will be directed to the home screen with three tabs. At registration, the user password is encrypted using MySQL AES_ENCRYPT () and stored at the server side. At login, the entered password is encrypted and compared to the stored encrypted value. This ensures the confidentiality.

The Home tab gives a welcome message as well as buttons for recording an issue and getting registered to the GCM service. Any newly registered user has to click on ‘Get registered for notifications’ button. At this version, the View all notifications function was not implemented thus the button does not work.

When the user clicks on the ‘record a water issue complaint’ button, he will be directed to a screen including a Google map as well as a disabled form which needs to be completed for recording the issue. First, the user is given an instruction to click on ‘Start recording an issue’ button to enable the form. Right after that, an instruction text appears below the button saying to mark the location of the issue on the map by doing a long click. After a long click, a map marker with the colour red will appear on the correct location. Red refers to an issue with the status as “not addressed yet”. Then, he can continue on entering issue title, relevant details and selecting a radio button

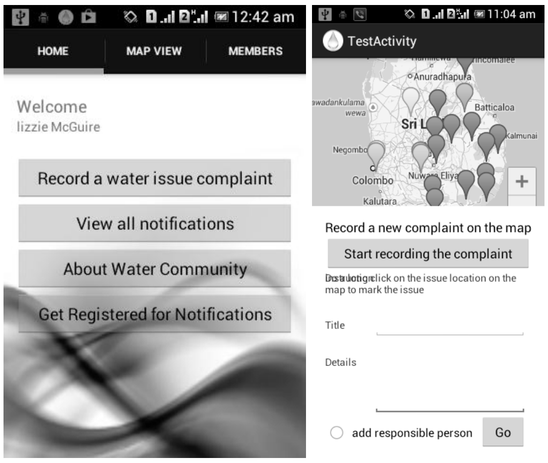

Figure.3 Home Screen (in left) and record new issue screen (in right)

if he wants to assign a responsible member to it. Then after clicking the ‘Go’ button, if his option was to assign a member, another screen with a member list will appear by giving the user the ability to select a member. After the selection, the user will be directed to the home tab while indicating the success of recording the complaint.

At the server side, the recorded issue details will be received and the server will store them in the back end database. In the meantime, it will send a notification to the assigned member. In this initial version, for demo purposes and for tackling the complexities, the notification was send back to the device which made the record. Otherwise, several Android client devices are needed for checking this functionality. For this project, it was not done. Also storing the GCM registration ID of each member device is an important requirement which had been missed in the implementation. They can be addressed and done at further development.

When the user selects the Map View tab, he gets to view the overall complaint map of the country. The markers are the locations of each complaint and they are in four different colours; Red, Orange, Green and Yellow. Red indicates ‘not addressed’, Orange indicates ‘being addressed’ and Green indicates ‘addressed’. Yellow is an indication that the particular issue is a responsibility of that particular device user. The user can tap on a marker so that he will get a dialog screen including the details about the complaint, status, notes and also another section to add notes and change status which will get enabled only if that particular user is the responsible person. After that, when the user clicks on OK button, the dialog box gets closed. If it was the responsible person and he did add notes or change status, the relevant request will be sent to the web service for modifying the database. After several seconds, the map view will get updated and display the new issue information.

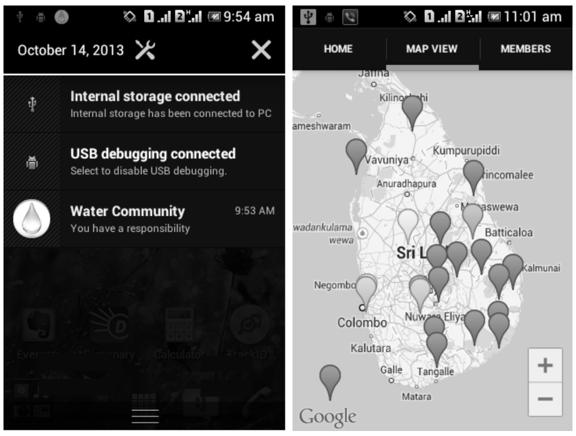

Figure.4 Responsibility notification (in left) and Map view tab (in right)

The user can select the Members tab and see a whole list of all the community members with their name, designation and contact email. Also by navigating to the options menu at any instance of the application, Map colour scheme, information and your details sections can be selected. Map colour scheme gives map marker colour indications. Information dialog box gives details and instructions about the usage of this application. Your details section displays the users' member details.

VII. Conclusion

The 'WaterCommunity' web based Android Application addresses the essential requirement of a mobile communication platform for the members in a typical water community. This initial version of the application gives the main functionality of letting members share, notify and update information about water issue complaints all around the country. Also they can view a list of all community members as well. However, there are some implementation aspects which have to be addressed or can be extended more and further developed in a better way. Following are some of such identified aspects.

1) Completion of the notification feature implementation

The main feature was implemented almost in a complete manner but the notification sending feature was not completed as it was done to the same device.

2) Enabling WS security features in a well manner

Throughout the development, the security in between the client and web service communication was not focused enough. Therefore, an intrusion risk is there. By using the Metro web service security features to define and ensure WS security, it can be resolved.

3) Use of Serialization between client and web service

The communication between the Android client and the web service was done using SOAP just in terms of parameter passing. Actual bean objects are not being serialized and passed. In the future enhancements, the application can be modified in order to pass serialized objects required in request processing. In that way, there is no need to transmit a large number of plain parameters inside a SOAP request.

With the above mentioned changes and new features, further development of the application can be carried out while adding new functionality such as the ability to search members by name/designation/region, sharing and updating water quality information and more community work related features. However, it can be concluded that this initial 'WaterCommunity' Android Application development was a success.

References

[1] www.waterboard.lk/scripts/ASP/Customer_Grievances_System.asp [last accessed: 13.07.2014]

[2] http://www.thameswater.co.uk/help-and-advice/9778.htm,[last accessed: 13.07.2014]

[3] http://www.durban.gov.za/City_Services/water_sanitation/Pages/default.aspx , [last accessed: 13.07.2014]

[4] http://www.southernwater.co.uk/about-us/about-southern-water/our-customer-promise/complaints.asp , [last accessed: 13.07.2014]

[5] http://www.waterboard.lk/Scripts/htm/Grievances_Status.htm, [last accessed: 13.07.2014]

[6] https://www.affinitywater.co.uk/customer-leakage-advice.aspx , [last accessed: 13.07.2014]

[7] http://en.wikipedia.org/wiki/Non-revenue_water , [last accessed: 13.07.2014]

[8] Android and GCM - Broadcast yourself, http://avilyne.com/?p=267 , [last accessed: 21.11.2013]

[9] Android App Development: Menus part 1: Options menu |Mobile Orchard , http://mobileorchard.com/android-app-development-menus-part-1-options-menu/,[last accessed: 21.11.2013]

[10] Android App Development: Menus part 1: Options menu|Mobile Orchard , http://mobileorchard.com/android-app-development-menus-part-1-options-menu/, [last accessed: 21.11.2013]

[11] Android Custom ListView with Image and Text, http://www.androidhive.info/2012/02/android-custom-listview-with-image-and-text/, [last accessed: 21.11.2013]

[12] Android Developers, http://developer.android.com, [last accessed: 21.11.2013]

[13] Android Fragment Example,http://manishkpr.webheavens.com/android-fragment-example/, , [last accessed: 21.11.2013]

[14] Android-er:Google Maps Android API v2 example: detect MarkerClick and add Polyline, http://android-er.blogspot.in/2013/01/google-maps-android-api-v2-example_5213.html, , [last accessed: 21.11.2013]

[15] Android Login activity with MySQL database connection | Code On Cloud, http://codeoncloud.blogspot.com/2012/07/android-login-activity-with-mysql.html, [last accessed: 21.11.2013]

[16] Android Login and Registration Screen Design, http://www.androidhive.info/2011/10/android-login-and-registration-screen-design/, [last accessed: 21.11.2013]

[17] Anroid MySQL Client (Part 1) | Code On Cloud, http://codeoncloud.blogspot.com/2012/03/android-mysql-client.html, [last accessed: 21.11.2013]

[18] Android Notifications - Tutorial, http://www.vogella.com/articles/AndroidNotifications/article.html, [last accessed: 21.11.2013]

[19] Android Push Notifications | Parse, https://parse.com/tutorials/android-push-notifications, [last accessed: 21.11.2013]

[20] Android SDK: Unit Testing with the JUnit Testing Framework, http://mobile.tutsplus.com/tutorials/android/android-sdk-junit-testing/, [last accessed: 21.11.2013]

[21] Android SDK: Working with Google Maps - Application Setup, http://mobile.tutsplus.com/tutorials/android/android-sdk-working-with-google-maps-application-setup/, [last accessed: 21.11.2013]

[22] Android User Session Management using Shared Preferences, http://www.androidhive.info/2012/08/android-session-management-using-shared-preferences/, [last accessed: 21.11.2013]

[23] Android web service access using Async Task | Code on Cloud, http://codeoncloud.blogspot.com/2013/07/android-web-service-access-using-async.html, [last accessed:

21.11.2013]
[24] Basics | Android Tutorial, http://android.programmerguru.com/category/basics/, [last accessed: 21.11.2013]
[25] Getting started with JAX-WS Services, https://netbeans.org/kb/docs/websvc/jax-ws.html, [last accessed: 21.11.2013]
[26] How to call java web service in android|Android Tutorial, http://android.programmerguru.com/how-to-call-java-web-service-in-android/, [last accessed: 21.11.2013]
[27] How to create Java webservice in netbeans|Webservice Tutorial, http://programmerguru.com/webservice-tutorial/how-to-create-java-webservice-in-netbeans/, [last accessed: 21.11.2013]
[28] Implementing Fragment Tabs in Android, http://www.androidbegin.com/tutorial/implementing-fragment-tabs-in-android/, [last accessed: 21.11.2013]
[29] Implementing Push Notifications with GCM|Fryer Blog, http://fryerblog.com/post/30057483199/implementing-push-notifications-with-gcm, [last accessed: 21.11.2013]
[30] Java SOAP Web Service with Apache Axis2 in Netbeans|code zone4, http://codezone4.wordpress.com/2012/11/03/java-web-service-with-apache-axis2-in-netbeans/, [last accessed: 21.11.2013]
[31] ksoap2-android - A lightweight and efficient SOAP library for the Android platform, https://code.google.com/p/ksoap2-android/, [last accessed: 21.11.2013]
[32] MySQL AES_ENCRYPT() - w3resource, http://www.w3resource.com/mysql/encryption-and-compression-functions/aes_encrypt%28%29.php, [last accessed: 21.11.2013]
[33] Speaking SOAP with Android, http://www.shanekirk.com/2011/11/speaking-soap-with-android/, [last accessed: 21.11.2013]
[34] Tomcat MySQL Connection - Using JDBC to connect Tomcat to MySQL, http://www.mulesoft.com/tomcat-mysql, [last accessed: 21.11.2013]
[35] Using Google Cloud Messaging - Samsung Developers, http://developer.samsung.com/android/technical-docs/Using-Google-Cloud-Messaging, [last accessed: 21.11.2013]
[36] Using SOAP with Java|Binary Data Types, http://www.jguru.com/article/web-services/soap-with-Java-part2-binary.html, [last accessed: 21.11.2013]
[37] Web Services Tutorial, http://www.w3schools.com/webservices/default.asp, [last accessed: 21.11.2013]
[38] Yiyu Jia's technical Blog: A simple tutorial about creating SOAP Web Service in netbean 6.9.1 with Metro and user user defined class as parameters for web service call, http://yiyujia.blogspot.com/2011/02/simple-tutorial-about-creating-metro.html, [last accessed: 21.11.2013]