Survey On Android OS And C2DM Service

R.R.Dhruva[1] and V.V.N.S.Sudha[2]
[1]CVR College of Engineering, Department of ECE., Ibrahimpatan, R.R.District, A.P., India
Email: rinku_dhruva@yahoo.com
[2]CVR College of Engineering, Department of ECE., Ibrahimpatan, R.R.District, A.P., India
Email: vedantam_sudha@yahoo.com

Abstract—**Android is a mobile operating system for mobile devices such as mobile tablet computers developed by Google Inc and the Open Handset Alliance (OHA).By means of Android Cloud to Device Messaging (C2DM) is a service developers send data from servers to their applications on Android devices. A simple, lightweight mechanism is provided by the service which is used by the servers to tell mobile applications to contact the server directly, to fetch updated application or user data. All aspects of queuing of messages and delivery to the target application running on the target device are handled by the C2DM service. Using C2DM it is easy for mobile applications to synchronize data with servers. This paper mainly surveys on the introduction to Android OS and C2DM service.**

I. INTRODUCTION

There are still several limitations for the current mobile operating systems. Some of them, like iPhone and BlackBerry OS, are designed for and can be used only in specific types of mobile devices [Expert users may need to develop their own applications that require an open platform. Closed source systems such as Windows Mobile are not flexible enough for this purpose. Finally, another important reason is that people want their cell phone functioning like a PC in that whatever they can access on a desktop, they should also be able to access on their cell phones. Therefore, an operating system running on a cell phone should be similar to a common desktop operating system. Symbian OS, while having the largest market share, is not. For all above reasons, on 21 Oct 2008, Google released Android, an open source software platform and operating system, which can run on every mobile device, with the hope of reaching as many mobile users as possible. Android is based on the Linux 2.6 kernel, and it provides an Android is an operating system based on Linux with a Java programming interface. It provides tools, e.g. a compiler, debugger and a device emulator as well as its own Java Virtual machine (Dalvik Virtual Machine - DVM).

Hence, standard Java bytecode cannot be run on Android. Java Class files are converted into "dex" (Dalvik Executable) files using a tool “dx”. The applications of Android are packed into an .apk (Android Package) file by the program "aapt" (Android Asset Packaging Tool).To simplify development Google provides the Android Development Tools (ADT) for Eclipse. Automatic conversion from class to dex files is performed by ADT and is created during deployment. OpenGL libraries in Android support 2-D and 3-D graphics and data storage is supported in a SQLite database. Every Android application runs in its own process and under its own userid which is generated automatically by the Android system during deployment. Therefore, all the running applications are isolated from each other and a misbehaving application cannot easily harm other Android applications.

II. IMPORTANT ANDROID COMPONENTS

An Android application consists out of the following parts [4]:

- Activity - Represents the presentation layer of an Android application, e.g. a screen which is seen by the user. There can be several activities in an Android application and there can be switching between those activities during runtime of the application.
- Views - The User interface of the Activities is build with widgets classes which inherent from "android.view.View"."android.view.ViewGroups" manages layout of rhe views.
- Services - perform background tasks without providing an UI. They can notify the user via the notification framework in Android.
- Intents are asynchronous messages which allow the application to request functionality from other services or activities. An application can call directly a service or activity (explicit intent) or ask the Android system for registered services and applications for an intent (implicit intents). For example the application could ask via an intent for a contact application. Application register themselves to an intent via an IntentFilter. Intents are a powerful concept as they allow to create loosely coupled applications.
- Broadcast Receiver - receives system messages and implicit intents, can be used to react to changed conditions in the system. An application can register as a broadcast receiver for certain events and can be started if such an event occurs.

The real-time responsiveness or latency measurement on Android is broken down in two parts. The first part is the latency introduced in handling of an interrupt within the Linux kernel i.e., the time it takes for the Linux kernel, after

receiving an interrupt (timer interrupt in our experiment), to propagate this event to the event management layer in the kernel. The second part is the latency introduced by Dalvik VM, i.e., the time difference between when it receives the event from the kernel event management layer and passes it up to the Application running on top of the VM.

III. Android Cloud To Device Messaging

Primary characteristics of Android Cloud to Device Messaging are [6]:

- It allows third-party application servers to send lightweight messages to their Android applications. The messaging service is not designed for sending a lot of user content via the messages. Rather, it should be used to tell the application that there is new data on the server, so that the application can fetch it.
- There is no gurantee about delivery or the order of messages. So, for example, while you might use this feature to tell an instant messaging application that the user has new messages, you probably would not use it to pass the actual messages.
- There is no need for an application to run to receive messages. When the message arrives, system wakes up the application via Intent broadcast.
- It does not provide any built-in user interface or other handling for message data. C2DM simply passes raw message data received straight to the application, which has full control of how to handle it. For example, the application might post a notification, display a custom user interface, or silently sync data.
- Android 2.2 or higher version is required that also have the Market application installed.
- It uses an existing connection for Google services. Users are required to set up their Google account on their mobile devices.

C2DM is basically divided into two categories:

a) Components

b) Credentials

Components are regarding the physical entities that play a role in C2DM.Credentials are the IDs and tokens used at different stages which ensure the authentication of parties, and that message is going to correct place. The various components used are Mobile Device, Third-Party Application Server, C2DM Servers. Mobile device is the device that is running an Android application that uses C2DM. This must be a 2.2 Android device that has Market installed, and it must have at least one logged in Google account. Server sends data to an Android application on the device via the C2DM server using third party Application Server .C2DM Servers are servers that take messages from the third-Party application server and sends them to device. Sender ID, Application ID, Registration ID, Google User Account, Sender Auth Token are the various credentials used. Sender ID is an email account associated with the application's developer. The sender ID is used in the registration process to identify a Android application that is permitted to send messages to the device. This ID is typically role-based rather than being a personal account. Application ID is the application that is registering to receive messages. The application is identified by the package name from the manifest. This ensures that the messages are targeted to the correct application. Registration ID is issued by the C2DM servers to the Android application that allows it to receive messages. Once the application has the registration ID, it sends it to the third-party application server, which uses it to identify each device that has registered to receive messages for a given application. In other words, a registration ID is tied to a particular application for C2DM to work, the mobile device must include at least one logged in Google account running on a particular device. ClientLogin Auth token that is saved on the third-party application server that gives the application server authorized access to Google services. The token is included in the header of POST requests that send messages. For more discussion of ClientLogin Auth tokens. The primary processes involved in cloud-to-device messaging is:

- Enabling C2DM: An Android application running on a mobile device registers to receive messages.
- Sending messages: Messages are sent by third-party application server to the device.
- Receiving messages: An Android application receives a message from a C2DM server.

A. Enabling C2DM

This is the sequence of events that occurs when an Android application running on a mobile device registers to receive messages:

1. The first time the application needs to use the messaging service, it fires off a registration Intent to a C2DM server.
2. If the registration is successful, the C2DM server broadcasts a REGISTRATION Intent which gives application a registration ID.
3. To complete the registration, the application sends the registration ID to the application server. The application server typically stores the registration ID in a database.

B. Sending the Message

For an application server to send a message, the following things must be in place:

- The application has a registration ID that allows it to receive messages for a particular device.
- The third-party application server has stored the registration ID.

There is one more thing that needs to be in place for the application server to send messages: ClientLoginauthorizationtoken. This is something that the developer must have already set up on the application server

DOI:10.32377/cvrjst0209

for the application. The ClientLogin token authorizes the application server to send messages to a particular Android application. An application server has one ClientLogin token for a particular 3rd party app, and multiple registration IDs. Each registration ID represents a particular device that has registered to use the messaging service for a particular 3rd party app.

Here is the sequence of events that occurs when the application server sends a message:

1. The application server sends a message to C2DM servers.
2. Google enqueues and stores the message in case the device is inactive.
3. When the device is online, Google sends the message to the device.
4. On the device, the system broadcasts the message to the specified application via Intent broadcast with proper permissions, so that only the targeted application gets the message. This wakes the application up. The application does not need to be running beforehand to receive the message.
5. The application processes the message. If the application is doing non-trivial processing, you may want to grab a wake lock and do any processing in a Service.

An application can unregisterC2DM if it no longer wants to receive messages.

C. Receiving a Message

This is the sequence of events that occurs when an Android application running on a mobile device receives a message:

1. The system receives the incoming message and extracts the raw key/value pairs from the message payload.
2. The system passes the key/value pairs to the targeted Android application in a com.google.android.c2dm.intent.RECEIVE Intent as a set of extras.
3. The Android application extracts the raw data from the RECEIVE Intent by key and processes the data.

D. Writing Android Applications that Use C2DM

The various steps involved in writing the application are:

- Creating the manifest.
- Registering for C2DM.
- Unregistering from C2DM.
- Handling received data.

IV. Creating The Manifest

Every application must have an AndroidManifest.xml file in its root directory. The manifest presents essential information about the application to the Android system, information the system must have before it can run any of the application's. To use the C2DM feature, the manifest must include the following [6]:

- com.google.android.c2dm.permission.RECEIVE states that the application has permission register and receive messages.
- android.permission.INTERNET states that the application has permission to send the receiver key to the 3rd party server.
- applicationPackage + ".permission.C2D_MESSAGE prevents other applications from registering and receiving the application's messages.
- The permision com.google.android.c2dm.SEND is required by the receiver, so that the message can be sent only by the C2DM framework. Both registration and the receiving of messages are implemented as Intents.
- If the C2DM feature is critical to the application's function, android:minSdkVersion="8" should be set in the manifest. This ensures that the application cannot be installed in an environment in which it could not run properly.

V. Registering For C2DM

An Android application needs to register with C2DM servers before receiving any message. To register it needs to send an Intent (com.google.android.c2dm.intent.REGISTER), with 2 extra parameters:

- sender is the ID of the account authorized to send messages to the application, typically the email address of an account set up by the application's developer.
- app is the application's ID, set with a PendingIntent to allow the registration service to extract application information.

Registration is not complete until the application sends the registration ID to the third-party application server. The application server uses the registration ID to send messages that are targeted to the application running on that particular device.

VI. Handling Received Data

When the C2DM server receives a message from the third-party application server, C2DM extracts the raw key/value pairs from the message payload and passes them to the Android application in the com.google.android.c2dm.intent.RECEIVE Intent as a set of extras. The application extracts the data by key and processes it, whatever that means for that application.

VII. Developing And Testing Your Applications

Here are some guidelines for developing and testing an Android application that uses the C2DM feature:

- To develop and test your C2DM applications, you need to run and debug the applications on an Android 2.2 system image that includes the

necessary underlying Google services.

- To develop and debug on an actual device, you need a device running an Android 2.2 system image that includes the Market application.
- To develop and test on the Android Emulator, you need to download the Android 2.2 version of the Google APIs Add-On into your SDK using the Android SDK and AVD Manager. If the C2DM feature is critical to the application's function, be sure to set android:minSdkVersion="8" in the manifest. This ensures that the application cannot be installed in an environment in which it could not run properly.

VIII. Limitations Of C2DM:

- The message size limit is 1024 bytes.
- Google limits the number of messages a sender sends in aggregate, and the number of messages a sender sends to a specific device.

Conclusion

Android is truly open, free development platform based on linux and open source. Handset makers can use and customize the platform without paying the royalty. A component-based architecture inspired by internet mash-ups. Parts of one application can be used in another in ways not originally envisioned by the developer and can even replace built-in components with own improved versions. This will unleash a new round of creativity in the mobile space.C2DM service is more efficient than other techniques like polling as it results in fresher data and more efficient use of network and battery.

References

[1] Wikipedia *en.wikipedia.org/wiki/Android.*
[2] *Google Android SDK, http://developer.android.com/sdk/index.html*.
[3] http://android-developers.blogspot.com/2010/05/android-cloud-to-device-messaging.html.
[4] http://www.vogella.de/articles/AndroidCloudToDeviceMessaging/article.html.
[5] http://blog.mediarain.com/2011/03/simple-google-android-c2dm-tutorial-push-notifications-for-android/
[6] http://code.google.com/android/c2dm/

DOI:10.32377/cvrjst0209