The Design and Implementation of Individual Push System Based on Android Platform

Heng-hui LIANG and Ze-hua GAO
Beijing University of Posts and Telecommunications, Beijing, China

Keywords: Push system, Individual, XMPP, Androidpn, Subscribe model.

Abstract. With the number of users of android phone increasing, the research on push technology facing android client is becoming increasingly important. There are mainly three traditional push method: Polling, SMS Push and IP Push. However, these push methods have disadvantages respectively. This article analyzed and compared above three methods firstly, and then focused on the push system based on XMPP. In addition, we designed and implemented a push system of XMPP protocol based on the open-source project Androidpn, and imported subscribe model into this push system. Finally, we verified the function of this push system.

Introduction

According to the 37th Statistics Report For China Internet Developing Situation: the amount of users of mobile phone has reached 620 million, of which 90.1 percent of users surf the Internet by mobile phone, mobile network and internet created a new pattern of life. Push technology, also known as webcasting technology, is a kind of technology of broadcasting custom information to users by internet. This kind of technology is proposed firstly by Point Cast Network Company. The presence of this technology greatly enhanced pertinence and validity of getting information to users. Therefore, the research of push technology facing mobile phone should have good potential applications and have important significance to help obtaining valid information.

According to implementation rationale, push technology based on Android can be divided to three kind of technology: Pull, SMS Push and IP Push. Compare to traditionally initiating by client and pulling data from server, proactively push technology is faster, more accurate and easier to customize. In this paper, firstly we analyzed the principle and Implementation of three kind of push technology, looking at the principle of push technology based on XMPP[1] (Extensible Messaging and Presence Protocol) and the design and implementation of push system based on open source project Androidpn (Android push notification). Finally, verified function of this push system by functional testing and analyzing this system.

Analysis of Push Technology Based on Android

The Overview of Push Technology

Push technology is a kind of technological means used to publish information. We can obtain information from Internet or Manufactures by certain technology standard or protocol and then transfer message to users by certain methods (such as e-mail, short message, webpage, notification, etc). So far, push technology facing Android platform primarily has Pulling, SMS Push, IP Push so on three methods.

Polling

When using this technology, we design relative trigger condition according to application scene and rules of use that defined in advance. When satisfy trigger condition, android client will use pulling method to obtain updated message from server by sending request to server. Using this method, one scenario is that android client directly obtains updated data from server when it satisfies conditions designed in advance. Another scenario is that android client checks whether there is updated message
or not in server by sending requests to server constantly. If there is updated message, android client will pull data to local device. Otherwise, android client will wait to send the next request. Actually, this is a polling method. This method is easy to realize, using which users can freely select update strategy and frequency as needed, but periodically connecting to Internet is not a good solution to android client whose electric quantity and data traffic is limited. What is more, if we select a long period as request cycle, using this method we can’t obtain updated message from server timely and effectively. This method is just suitable to some scenarios that don’t need highly real-time capability and have periodic trends, such as weather, news and so on.

**SMS Push**

We can push message to users by sending short message with base station set up by mobile telecom carrier. As long as the SIM card in your phone is active, we can send push messages to users through base station in any time. However, SMS (short messaging service) need high charge, the kind and size of message is limited using this method.

**IP Push**

IP Push is the most popular push technology in current most applications. Its principle is client that has fixed IP address maintains a persistent connection with server by heartbeat mechanism. We can transfer data between client and server by this persistent connection.

**C2DM**

C2DM(Cloud to Device Messaging) is a solution of push message that is released by google company in Android 2.2 version, which can help developer pushing message to client. This service provides a simple, light weight push solution, which can realize the function of server pushing message to client initiative. The message which is pushed can be specific content or just a notification which is used to notify client when there are updated information or data on server. C2DM is in charge of maintaining the persistent connections between clients and server, maintaining request queue, pushing message from server to android client and so on. However, this service has significant disadvantages. Firstly, C2DM requires that the version of android client must be higher than 2.2 and devices whose version is between 1.5 and 2.1 do not support this service. Secondly, devices whose version is lower than 4.0 need to register google account and then google account must be online always because the server of C2DM could not keep in touch with this client if it is offline. Finally, this service is not stable when we use it in our country because of the server of C2DM being in foreign.

**Initiative Push System**

In consideration of the security of using third-party push platform, the most popular solution is setting up push platform by ourselves. Currently, MQTT (Message Queuing Telemetry Transport) and XMPP [1] (Extensible Messaging and Presence Protocol) are two of popular protocols to realize push service. MQTT is a concise, tiny, power-saving, data-traffic-saving protocol [2] which has strong expansibility. And there is sever-side component rsmb based on C++. However, MQTT is premature and not easy to realize. Moreover, the server-side component rsmb is closed and expensive to deploy. XMPP is a mature and powerful protocol which has very strong expansibility because of transporting by XML stream and extending the flexible character from XML [3]. Moreover, XMPP has been a standardization protocol which is standardized by IETF of ISO [4-5]. XMPP has been applied to many chat systems and there has been open-source project Androidpn based on XMPP protocol which is realized by Java language.

Through the above comparison and analysis, we can conclude that to satisfy the request of power-saving and efficient, push mechanism based on MQTT is better compare to above other several free push mechanism. However, considering the immature and uniform of MQTT, we chose XMPP which has good performance as well, as the main part. We will focus on the push technology based on XMPP.
XMPP Protocol

Brief Introduction of XMPP

XMPP is the most flexible protocol, which is transported by XML stream and inherited the flexible character of XML. Therefore, applications based on XMPP have strong expansibility. We can send message and handle request from users by expanded XMPP, develop applications based on XMPP (such as content distribution system, Location-based service and so on). In addition, XMPP includes a software protocol for server-side which allows a client communicating with another client. This character allows developer to develop application for client or add function for a system.

Concept of XMPP

Jabber, as the predecessor of XMPP, is an instant messaging communication protocol which is created by open source group. The standardization work of XMPP has been finished by IETF of ISO [4-5]. The core product of standardization is the definition of transport protocol based on XML stream, which allow XMPP can be used in more specification platform of communication protocol. With the feature of easy to read and parse, XMPP becomes a concise and beautiful protocol.

Network Structure of XMPP

XMPP defines three roles: client, server and gateway [6]. Any two of the three roles support bidirectional communication. As is shown in Figure 1 below, Server is responsible for managing connection, recording routing information and client information. Gateway is in charge of communication with isomeric instant messaging system. Clients connect to server by TCP/IP protocol and then transport XML stream by this connection.

Transport Content of XMPP

The instructions to transport are relative to instant messaging. Previously, these instructions are transported either by plain text adding blank character, parameter and line break (e.g. MSN) or by binary system (e.g. QQ). However, XMPP transport instant messaging instructions by text formatted by XML, which made it easy to read, easy to parse and easy to debug and develop. The hardcore of XMPP is the stream protocol of transporting XML stream on Internet segmentally. The base of instant messaging instructions of XMPP is just this stream protocol, which is a very important network based protocol that can be further used. That is to say, what is transported in XMPP by TCP protocol is XML stream.

Address Pattern of XMPP

XML of XMPP includes three top labels: Message, Presence and IQ. Their specific meanings are as follows:

1) Message is used to send message between two users, showing specific content of message that are to send. When user sending a new message, just add message label to XML and then add specific content under this label.

2) Presence is used to express the state of users. Not every message needs to add this label. We need to add this label and mark corresponding state in XML when the state of user is changed.

3) IQ is used to express semantic request and response, can also be considered as a request/response mechanism. This label manage the conversion of two users who connected to XMPP server, allowing them to query and response by XML format, of which source entity send request and target entity receive request and then response. The most important property of IQ is type, which can be assigned as Get, Set, Result and Error. Get refers to obtain current threshold. Set express to obtain current threshold. Set express set up or replace the value queried by Get. Result express having responded previous query. Error express emerged errors when querying or responding process.
The Design and Implementation of Push System

In this article, the push system based on XMPP is designed on the basis of the open source project Androidpn. This Article modified the push mechanism of Androidpn and added subscription model to Androidpn, accordingly set up personalized recommendation model that can push message to users according to their subscription topic, which improved the pertinence of pushing message. The whole system can be divided to part: one is server-side, which is used to push message, maintain persistent connection with client, manage the list of online and offline clients and the subscription topics from different clients; the other is client, which is used to register and login to server, subscribe to topics and receive push message that belong to corresponding topics and then show these message in notification bar of android client.

Deployment of Server-side

The sever-side of this push system is implemented by modifying the open source project openfire. This server includes two parts: one is the XMPP service that listening 5222 port, which is responsible for communicating with XMPPConnection. Its main role was managing registration and authentication of users and pushing message; the other is web server, which is a light weight http server that used spring framework. It is responsible for receiving the web requests from users.

As is shown in Figure 2 above, the framework of Androidpn server can be divided to four layers. The uppermost layer of above framework includes four parts: Session Manager, Auth Manager, Presence Manager and Notification Manager. Session Manager is in charge of managing the sessions between clients and server. Auth Manager is responsible for the authentication of users in client-side. Presence Manager is in charge of managing login status of users. Notification Manager is responsible for pushing message from server to clients.

Using this solution, we can only send text message. However, it is enough to general push demands. We can’t look forward to obtaining all of data and message by just one push message. As a general rule, we just need to notify clients that there are updated data in server-side. And then clients send request initiatively to server to obtain updated data when they received notification.

Design and Implementation of Client

Android client need to use asmask.jar, an open source XMPP packet based on Java. This packet is on the basis of smack, another open source project based on openfire. When developing client, we need to import asmask.jar to project as library. Client establishes persistent connection with server by XMPPConnection provided by asmask packet and then do registration and login authentication operation. After above operations, clients can receive push message from server.
Therefore, there are two part functions to implement: one is registration, login, connection, and a series of operations, which can make the client establish a persistent connection with the server; the other is managing subscription topics, including selecting topics and submitting subscription topics to the server so that the server can push messages according to corresponding topics.

We also need to configure this project to make it can establish connection with the server. We need to open the project and find the file `androidpn.properties` in the `res/raw/` directory and then configure `apikey` (the secret key to authenticate the client), `xmppHost` (IP address of the server), `xmppPort` (port number listened by the server) three parameters, of which if the client is installed on a real phone, we configure `xmppHost` as the real IP address of our computer that deployed the server, we need to configure `xmppHost` as 10.0.0.2 otherwise and `xmppPort` should be configured as 5222 both situations.

**Working Process of XMPP Server and Client**

Figure 3 shows the working process of XMPP server, of which Figure 3. (a) shows the message-sending process of XMPP server and Figure 3. (b) shows the message-receiving process of XMPP server. Figure 4 shows the establishing process of connection between client and server.

![Figure 3. Flow chart of sending and receiving message of server.](image1)

![Figure 4. Establishing process of Connection between client and server.](image2)

**Function Test of Push System**

Test environment: A computer with Win7 as server host; Tomcat 7 used to deploy server; 2 real Android phones and an emulator as clients.

**Result**

Figure 5 shows the main page of client-side, through which we can launch connection and commit subscription topics. Figure 6 shows notification-received page, which proved that clients can receive push messages from the server. Figure 7 shows notification detail page, including subject, title, and content of received notification. Messages belong to other subjects (topics that users never signed up for) can’t be received by clients.
Conclusion

This Pager expounded principle and working mechanism of XMPP and designed push system based on XMPP. Importing topic-subscribed model to push system, this pager implemented a more targeted push system. This system can be used in most personalized commend applications.

References