ISSN: 2688-8653

A Unified Chat Message Management Approach based on Channel Mechanisms

Xingze Wu

North China Electric Power University, Baoding, Hebei, China

Abstract

This paper provides a solution to the problem of unified message management for single chat and group chat in chat software. In this paper, we adopt a unified approach to manage the message forwarding channels.

Keywords

Channel; Message; Single Chat; Group Chat; User; User Group.

1. Introduction

With the rapid development of Internet, people's meanings and lifestyles have changed dramatically. Internet communication software based on instant messaging technology has become the most convenient communication tool in modern society, and is used by the general public. One of the most crucial functions is chatting. There are two types of chatting scenarios: single chat and group chat. This approach makes the design and management of the software more cumbersome and increases the cost of maintaining the software later.

In order to solve the above problems, this paper proposes a unified chat message management based on channel mechanism, by establishing a channel for each user to unify the management of single chat and group chat, and this mechanism does not need to increase the redundancy of information, simplify the design and management of the software, and increase the development efficiency.

2. System Scenarios

2.1. Chat Software Introduction

This scenario assumes that the software developed is a simple chat software, and there are two main functions of the software: user-to-user chat and user group chat. In the user-to-user scenario, a user's chat message is sent to a specific user, while in the group chat, a user's message is sent to multiple users, and a user can participate in multiple single and group chats at the same time.

2.2. Scenario Problem Description

According to the scenario analysis, users will prioritize unread messages, and also randomly view chat logs with a user or a group of users, messages will involve a lot of queries and additions, queries will be based on time and user's name to bulk queries, this scenario application should have the following basis:

- Sorting information according to time
- Bulk search information based on time
- Bulk query information based on users

ISSN: 2688-8653

This should be done as quickly as possible and with a unified design to avoid additional design and maintenance hassles.

2.3. Scene Modeling

We assume that each user has a unique ID that identifies the user, and each user group has a unique ID that identifies the user group, and that users and user groups are managed separately so that user and user group IDs can be duplicated. A user-group table is maintained between users and user groups to record which users are included in the user group.

The message has attributes of FROM_ID and TO_ID and TYPE, where FROM_ID denotes the sender of the message, TO_ID denotes the receiver of the message, and TYPE denotes the type of the message, where it is assumed that 0 denotes a single chat and 1 denotes a group chat. When TYPE=0, then TO_ID means user ID, and when TYPE=1, TO_ID means group chat ID.

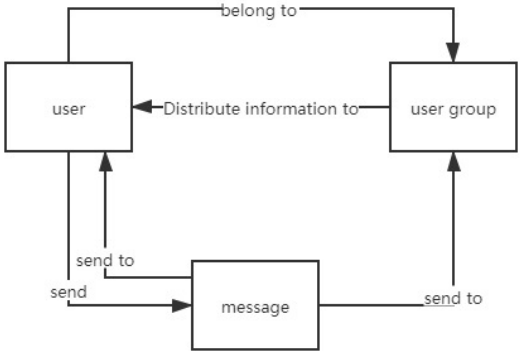

Fig 1. Scene simulation diagram

If the message is sent to a user, then the TO_ID of the message is recorded as the ID of the forwarding user and forwarded directly to the target user, If the message is sent to a user group, then the TO_ID of the message is recorded as the ID of the user group, and the user group is required to distribute the message to all users in the user group according to the user-group table.

3. Model Implementation Options

Analyzing the above models, this paper first briefly describes the other schemes and their advantages and disadvantages, and then gives the channel-based information management approach, introducing its logic model, building algorithms, and advantages and disadvantages compared to the above approaches.

3.1. Information Management Using Only the TYPE Field

This approach uses the TYPE field of the message to identify whether the message is sent to the user or to a group of users, in this way the user can send their own message based on their own ID, the algorithm to find all messages is given below:

For single chat messages, we can get the relevant information by sequentially querying the data, but for user group messages, we can only get the ID of the user group by querying the user-group table first, and then query the relevant information, which requires a second query. The advantage of this approach is that it is simple to implement, but its disadvantage is also obvious. It can be found that if this approach needs to sort the messages by their sending time, then it needs to sort the aggregated messages again, and it cannot sort the recent chat groups or users,

ISSN: 2688-8653

which is very first, and the two chat methods are managed in different ways, which increases the maintenance cost later.

Table 1. The algorithm to find all messages

Information management using only the TYPE field
1. Get user ID 2. According to the user ID, get the message with TO_ID as user ID or FROM_ID as user ID and TYPE as 0, which is the single chat message forwarded to or sent by the user. 3. Query user-group table based on user ID, get all user group IDs that the user is a member of, assign to IDs 4. for i in IDs 5. User group ID i, get the message with TO_ID i and TYPE 1, that is, the message forwarded to this user group 6. End for 7. Return the information obtained to the user

3.2. Ways to Increase Redundant Information

By analyzing the above approach, we can increase the redundancy of information, which not only reduces the number of queries, but also unifies the two chat methods, as given in the algorithm below:

Table 2. Ways to increase redundant information --send information section

1. User information 2. Get the user's ID 3. if the message sent by the user is a single chat message 4. Set the TO_ID of the message to the ID of the forwarding user, send the message 5. else The message sent by the user is a user group message 6. Get the user group ID of the user forwarding 7. Get the IDs of the users participating in this user group from the user-group table, assigning them as IDs 8. for i in IDs 9. Set the TO_ID of the message to i, forward to this user 10. end for 11. end if

ISSN: 2688-8653

Table 3. Ways to increase redundant information ---Access to Information section
1. Get the user's ID
2. Based on the user ID, query TO_ID for the information of ID
3. Return the information obtained to the user

The algorithm is divided into two parts, the first part is the sending algorithm, when the message is sent, the algorithm will send multiple copies of the message to the user group, each copy is sent to the users in the user group, by this way, it can simplify the complexity of obtaining information, the second part only needs to get the information of the user according to the user ID.

The advantage of this algorithm is that it simplifies the implementation of message acquisition and unifies the message management of single chat and group chat, but the disadvantage is also obvious, that is, when the number of users is very large, the database will store a lot of redundant information, we assume that each user group has 100 users, so when a user sends a message to this user group, then it is necessary to repeat the message into the database 100 times, the effective space utilization rate is only 1%. The effective space utilization rate is only 1%, if the group of users is more than one million, then the redundant information will occupy the storage terribly, which will affect the information query performance of the database.

3.3. Use of Channel Mechanisms

3.3.1. Introduction to the Channel

After combining and analyzing the advantages and disadvantages of the above two approaches, this paper proposes a channel-based message management approach, which unifies the management of single and group chats by means of a channel middle layer, in which users will first establish a channel with the time they need to chat, and then chat through the channel.

The channel has four attributes, which are as follows:

- USER_ID: The target user ID of the channel
- TO_ID: Connected user of the channel
- TYPE: The type of channel
- READ_TIME: Record the time of the last view
- READ_MSG; indicates if there are unread messages, 0 means no, 1 means yes

USER_ID and TO_ID indicate that the user with ID TO_ID can send messages to the user with USER_ID, TYPE has the same meaning as TYPE of the message, 0 means single chat channel, 1 means group chat channel, READ_TIME records the time when the user last read the channel.

The channel is unidirectional, when two users establish contact, then two channels need to be created to indicate that two users can communicate in both directions, when a user joins a user group, then a unidirectional channel needs to be created, TO_ID is the ID of the user group, The channel creation diagram is as follows:

ISSN: 2688-8653

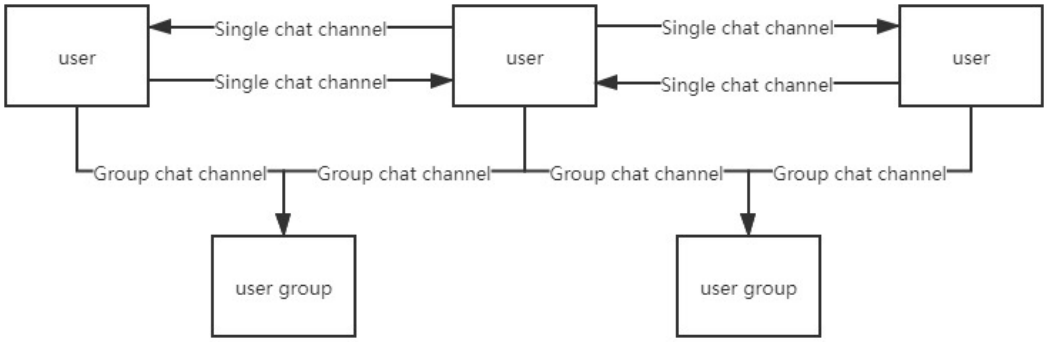

Fig 2. The channel creation diagram

3.3.2. Channel Establishment Algorithm

When users want to send a message, they need to establish a channel first, there are two types of channel establishment, i.e., single-chat channel and group-chat channel, and there are some differences between the two algorithms, which are listed here:

Table 4. Algorithm of Establishing a single chat channel

1. Get the ID of the user who established the channel and the ID of the target user, assign the value to TARGET_ID
2. Create channel, channel USER_ID and TO_ID are ID and TARGET_ID respectively, TYPE is 0, READ_TIME is current time
3. Create channel, channel USER_ID and TO_ID are TARGET_ID and ID respectively, TYPE is 0, READ_TIME is current time

Table 5. Algorithm of Establishing a group chat channel

1. Get the user ID and the user group ID, assign to ROOM_ID
2. Create channel, channel USER_ID and TO_ID are ID and ROOM_ID respectively, TYPE is 1, READ_TIME is current time

A two-way single-chat channel and a one-way group-chat channel are established in the above way, each of which stores the read state of the current user.

3.3.3. Algorithm for Sending Messages Using Channels

After the channel is established, the user can send the message in much the same way as in the above two scenarios, but with the addition of an attribute SEND_TIME, which indicates when the message will be sent.

Table 6. Message Delivery

1. Get the sending user ID and the target ID, assign the value to TARGET_ID
2. Create message, FROM_ID of message is ID, TO_ID is TARGET_ID, TYPE depends on message type

ISSN: 2688-8653

Both users and groups of users can be targeted and sent, so the two delivery methods are unified here.

3.3.4. Information Acquisition Algorithm Using Channels

Depending on the channel established by the user, we can use a uniform way to obtain information, and the algorithm for obtaining information is briefly described below:

Table 7. Access to information

1. Get all channels created by the user, assign the value to CHANNELS
2. For i in CHANNELS
3. Get information based on the status of the channel or TO_ID or TYPE
4. End for
5. Return the obtained information to the user

The user only needs to get all the information he needs through the pipe, or he can sort the information according to the READ_TIME of the channel.

3.4. Summary of the Programme

Compared with the first scheme, this scheme is more advantageous for the unified management of the management, which can manage the user's connection in a horizontal way by changing the status and properties of the channel, and also simplifies the fast distribution and reduces the maintenance cost.

In comparison with the second solution, this solution greatly reduces the storage of information, this solution does not store redundant information, and achieves unified management.

The advantage of the proposed scheme is that it is unified, the added middle layer separates the upper layer from the lower layer, the channel manages the details of sending and acquiring information, and provides a transparent service to the upper layer, unifying the management of information without significant performance degradation.

4. Conclusion

In this paper, we propose a unified way to manage information based on channels, through which not only can effectively simplify the development of information service code, but also unify the management of two scenarios, simplify the subsequent upgrade and maintenance, so that when the software is upgraded, by adding the service edge to the channel can suggest a quick addition of various functions to the software.

References

[1] Li, Tianshuang. Development and Application of Computer Software Development Technology [doi: 10. 16184/j.cnki.comprg.2021.04.11].

[2] Zou Yunlan, Liu Gaoping, Liu Qiao. Development of positioning and chat software based on wireless ad hoc network [doi:10.13777/j.cnki.issn1671-2250.2020.05.017].

[3] Sun Lei. Application of Layering Technology in Computer Software. [doi: 10.19 353/j. cnki. dzsj. 2021.19.065].