Mandatory Activity 3 Report

IT-University of Copenhagen 28/10/2024

Group “Peter Parker”

Markus Sværke Staael - msvs@itu.dk

Patrick Shen - pash@itu.dk

Nicky Chengde Ye - niye@itu.dk

Git repository with source code:
https://github.com/BDIS2024/Distributed_Systems/tree/master/Assignment_3

System Requirements

Here we will shortly discuss if and how our implementation fulfills the given system requirements.

R1 - Fulfilled

- Chitty-Chat is a distributed service, that enables its clients to chat. The service is using gRPC for communication. You have to design the API, including gRPC methods and data types.

This requirement is fulfilled and is best demonstrated by the reader reading the code.

R2 - Fulfilled

- Clients in Chitty-Chat can Publish a valid chat message at any time they wish. A valid message is a string of UTF-8 encoded text with a maximum length of 128 characters. A client publishes a message by making a gRPC call to Chitty-Chat.

Clients can try to publish any valid message whenever they have joined the chat. When a client publishes a message to the server, the server will check to see if the message is valid (Message length is less or equal to 128).
If the message is valid it will distribute the message among all of the connected clients.
If the message is invalid it will send a feedback message to the client trying to publish a message.

R3 - Fulfilled

- The Chitty-Chat service has to broadcast every published message, together with the current logical timestamp, to all participants in the system, by using gRPC. It is an implementation decision left to the students, whether a Vector Clock or a Lamport timestamp is sent.

This implementation of Chitty-Chat uses Lamport Timestamp to record logical timestamps. The timestamp is distributed together with the message.

R4 - Fulfilled

- When a client receives a broadcasted message, it has to write the message and the current logical timestamp to the log

The client does record any received message as evident by the logs later in this document.

R5 - Fulfilled

- Chat clients can join at any time.

There are no restrictions on when clients are allowed to join the server.

R6 - Fulfilled

- A "Participant X joined Chitty-Chat at Lamport time L" message is broadcast to all Participants when client X joins, including the new Participant.

All clients in the chat receive a join message whenever any participant joins. This message is not formatted as given but rather the form:
“X : has joined the chat. (L)”
Where X is the client joining and L is the lamport time in which they join.

R7 - Fulfilled

- Chat clients can drop out at any time.

Clients can drop out any time by publishing the message “leave” or getting a connection error.

R8 - Fulfilled

- A "Participant X left Chitty-Chat at Lamport time L" message is broadcast to all remaining Participants when Participant X leaves.

All clients in the chat receive a leave message whenever any participant leaves. This message is not formatted as given but rather the form:
“X : has left the chat. (L)”
Where X is the client leaving and L is the lamport time in which they join.

Technical Requirements

Here we will shortly discuss if and how our implementation fulfills the given technical requirements.

1 - Fulfilled

- Use gRPC for all messages passing between nodes

gRPC is used for all messages passing between nodes.

2 - Fulfilled

- Use Golang to implement the service and clients

Golang is used for the implementation of service clients.

3 - Fulfilled

- Every client has to be deployed as a separate process

Every client is a separate process.

4 - Fulfilled

- Log all service calls (Publish, Broadcast, ...) using the log package

All messages and errors are recorded using the log package. The log can be viewed by opening the logs file.

5, 6, 7 - Fulfilled

- Demonstrate that the system can be started with at least 3 client nodes
- Demonstrate that a client node can join the system
- Demonstrate that a client node can leave the system

This demonstration can be seen in the “Chitty-Chat Demonstration” chapter.

8 - Optional and unfulfilled

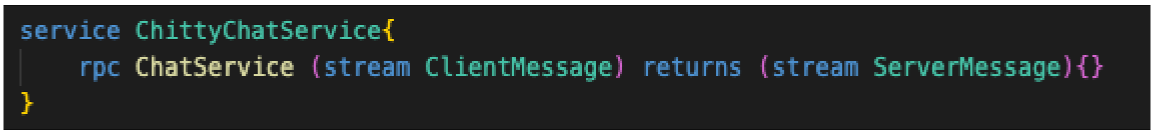
Hand-in Discussions

Stream Type

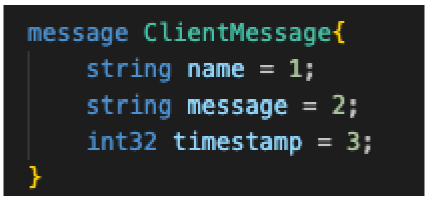
The implementation of the chat service uses bidirectional streaming to achieve a live chatting experience, so that when one user sends a message the other users receive it immediately. By extension of this the system architecture used is a server-client type. This means that the chat service has clients and a server. The client will send a request to send a message to the server. Then the server will respond with a server response and broadcast the message to all clients including the one that sent the request.

RPC Method

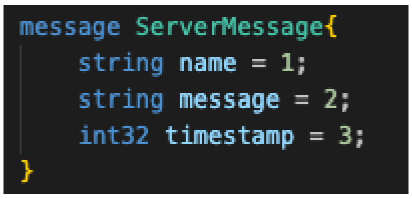
The chatservice only has one rpc method which is the bidirectional stream. The stream takes ClientMessages, and returns ServerMessages which are in essence the same thing, but with different names to differentiate between the usage. They both have a name field that correlates to the client name that sends the message, a message field for the chat message and a timestamp field for the lamport timestamp.



```
service ChittyChatService{
    rpc ChatService (stream ClientMessage) returns (stream ServerMessage){}
}
```



```
message ClientMessage{
    string name = 1;
    string message = 2;
    int32 timestamp = 3;
}
```



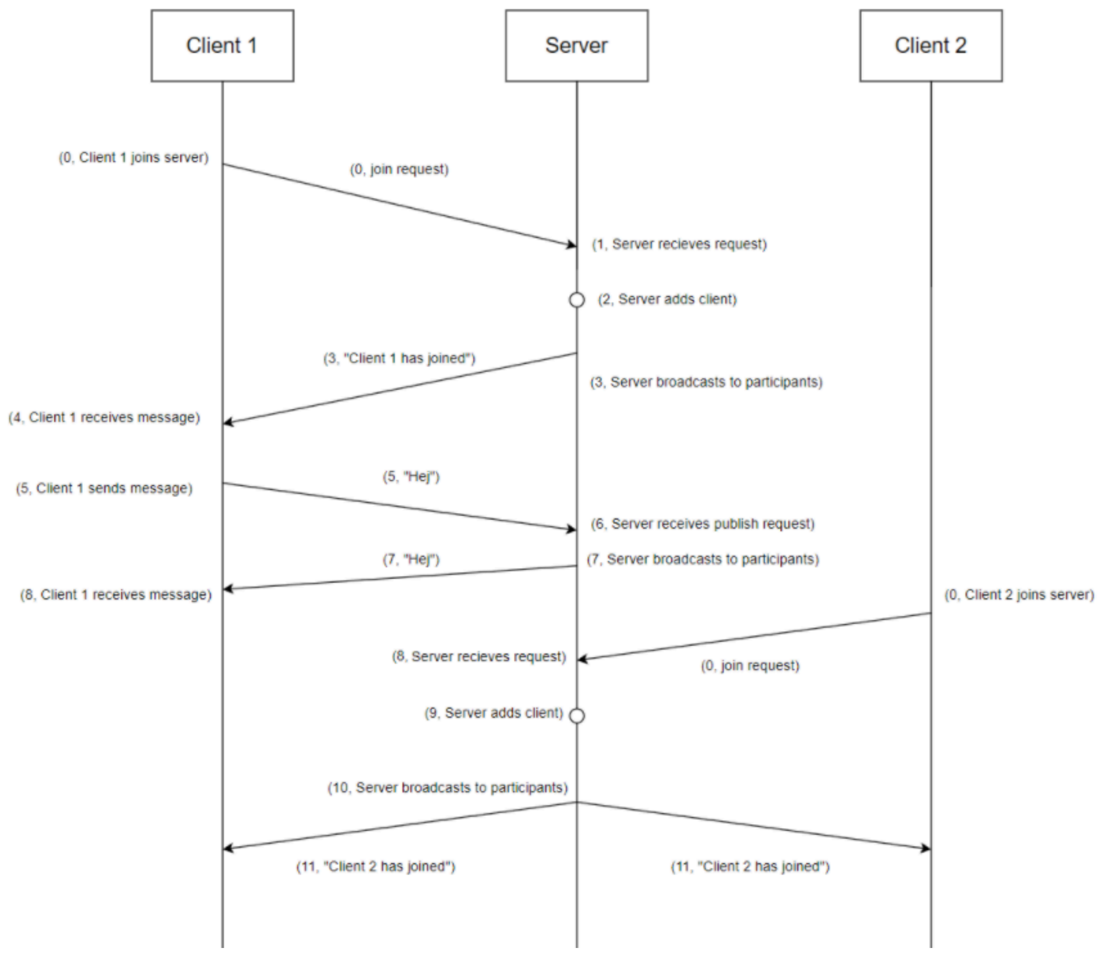
```
message ServerMessage{
    string name = 1;
    string message = 2;
    int32 timestamp = 3;
}
```

Lamport Timestamp

The logical timestamp the chatservice implements is a lamport timestamp, where each client and server has an initial counter set to 0, then at each local event this counter is incremented by 1. On sending a message request the node increments the counter and sends the counter along with the message via the underlying network link. On receiving a request take the max value of the counter sent via the network link the the local counter, and increment the counter by one.

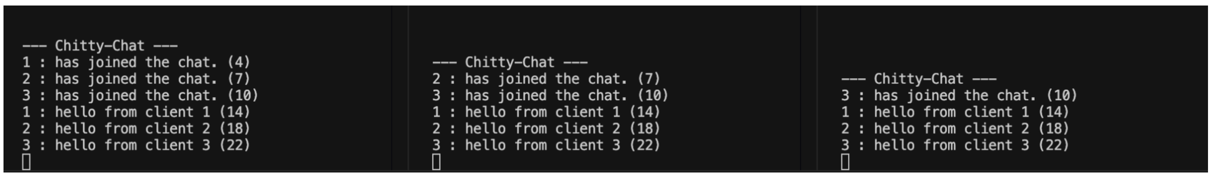
Sequence of RPC calls

Below is a diagram that traces the RCP calls between a server and two clients with lamport timestamps. The appendix includes a log of the RCP calls. The vertical lines represent the local timeline of events occurring on the different processes. An occurring event is represented by a tuple. The tuples consist of a lamport timestamp and a message/action. The arrows pointing to the other timelines represent an outgoing message/request, while a dot shows an isolated local event.


Client 1
Server
Client 2
(0, Client 1 joins server)
(0, join request)
(1, Server recieves request)
(2, Server adds client)
(3, "Client 1 has joined")
(3, Server broadcasts to participants)
(4, Client 1 receives message)
(5, "Hej")
(5, Client 1 sends message)
(6, Server receives publish request)
(7, "Hej")
(7, Server broadcasts to participants)
(8, Client 1 receives message)
(0, Client 2 joins server)
(8, Server recieves request)
(0, join request)
(9, Server adds client)
(10, Server broadcasts to participants)
(11, "Client 2 has joined")
(11, "Client 2 has joined")

Chitty-Chat Demonstration

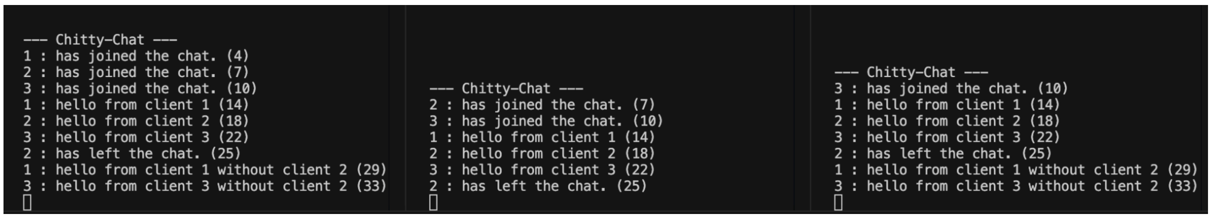
Below is a demonstration of the Chat Service in use with 3 clients. In the first figure it is demonstrated that three clients are able to join, send and retrieve messages. In the second figure it is demonstrated that a client can leave at any time and that the two other clients can continue to chat. In the last figure it is demonstrated that the client that left the chat can rejoin and continue chatting with the other clients.



```
--- Chitty-Chat ---
1 : has joined the chat. (4)
2 : has joined the chat. (7)
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
```

```
--- Chitty-Chat ---
2 : has joined the chat. (7)
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
```

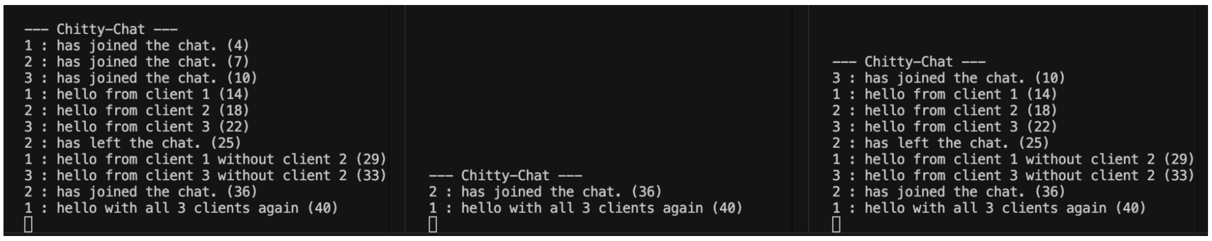
```
--- Chitty-Chat ---
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
```



```
--- Chitty-Chat ---
1 : has joined the chat. (4)
2 : has joined the chat. (7)
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
2 : has left the chat. (25)
1 : hello from client 1 without client 2 (29)
3 : hello from client 3 without client 2 (33)
```

```
--- Chitty-Chat ---
2 : has joined the chat. (7)
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
2 : has left the chat. (25)
```

```
--- Chitty-Chat ---
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
2 : has left the chat. (25)
1 : hello from client 1 without client 2 (29)
3 : hello from client 3 without client 2 (33)
```



```
--- Chitty-Chat ---
1 : has joined the chat. (4)
2 : has joined the chat. (7)
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
2 : has left the chat. (25)
1 : hello from client 1 without client 2 (29)
3 : hello from client 3 without client 2 (33)
2 : has joined the chat. (36)
1 : hello with all 3 clients again (40)
```

```
--- Chitty-Chat ---
2 : has joined the chat. (36)
1 : hello with all 3 clients again (40)
```

```
--- Chitty-Chat ---
3 : has joined the chat. (10)
1 : hello from client 1 (14)
2 : hello from client 2 (18)
3 : hello from client 3 (22)
2 : has left the chat. (25)
1 : hello from client 1 without client 2 (29)
3 : hello from client 3 without client 2 (33)
2 : has joined the chat. (36)
1 : hello with all 3 clients again (40)
```

Logs for demonstration with 3 clients.

```
2024/10/22 14:14:57 Server started on :5050
2024/10/22 14:15:21 Client sent request: Name: 1, Message: has joined the chat., Timestamp: (0)
2024/10/22 14:15:21 Server added client: 1, (1)
2024/10/22 14:15:21 Server received request: Name: 1, Message: has joined the chat., Timestamp: (0) at 2
2024/10/22 14:15:21 Server sent response: Name: 1, Message: has joined the chat., Timestamp: (3)
2024/10/22 14:15:21 Client received response: Name: 1, Message: has joined the chat., Timestamp: (3) at: 4
2024/10/22 14:15:23 Client sent request: Name: 2, Message: has joined the chat., Timestamp: (0)
2024/10/22 14:15:23 Server added client: 2, (4)
2024/10/22 14:15:23 Server received request: Name: 2, Message: has joined the chat., Timestamp: (0) at 5
2024/10/22 14:15:23 Server sent response: Name: 2, Message: has joined the chat., Timestamp: (6)
2024/10/22 14:15:23 Client received response: Name: 2, Message: has joined the chat., Timestamp: (6) at: 7
2024/10/22 14:15:23 Client received response: Name: 2, Message: has joined the chat., Timestamp: (6) at: 7
2024/10/22 14:15:24 Server added client: 3, (7)
2024/10/22 14:15:24 Server received request: Name: 3, Message: has joined the chat., Timestamp: (0) at 8
2024/10/22 14:15:24 Client sent request: Name: 3, Message: has joined the chat., Timestamp: (0)
2024/10/22 14:15:24 Server sent response: Name: 3, Message: has joined the chat., Timestamp: (9)
2024/10/22 14:15:24 Client received response: Name: 3, Message: has joined the chat., Timestamp: (9) at: 10
2024/10/22 14:15:24 Client received response: Name: 3, Message: has joined the chat., Timestamp: (9) at: 10
2024/10/22 14:15:24 Client received response: Name: 3, Message: has joined the chat., Timestamp: (9) at: 10
2024/10/22 14:15:30 Client sent request: Name: 1, Message: hello from client 1, Timestamp: (11)
2024/10/22 14:15:30 Server received request: Name: 1, Message: hello from client 1, Timestamp: (11) at 12
2024/10/22 14:15:30 Server sent response: Name: 1, Message: hello from client 1, Timestamp: (13)
2024/10/22 14:15:30 Client received response: Name: 1, Message: hello from client 1, Timestamp: (13) at: 14
2024/10/22 14:15:30 Client received response: Name: 1, Message: hello from client 1, Timestamp: (13) at: 14
```

```
2024/10/22 14:15:30 Client received response: Name: 1, Message: hello from client 1, Timestamp: (13) at: 14
2024/10/22 14:15:33 Client sent request: Name: 2, Message: hello from client 2, Timestamp: (15)
2024/10/22 14:15:33 Server received request: Name: 2, Message: hello from client 2, Timestamp: (15) at 16
2024/10/22 14:15:33 Server sent response: Name: 2, Message: hello from client 2, Timestamp: (17)
2024/10/22 14:15:33 Client received response: Name: 2, Message: hello from client 2, Timestamp: (17) at: 18
2024/10/22 14:15:33 Client received response: Name: 2, Message: hello from client 2, Timestamp: (17) at: 18
2024/10/22 14:15:33 Client received response: Name: 2, Message: hello from client 2, Timestamp: (17) at: 18
2024/10/22 14:15:37 Client sent request: Name: 3, Message: hello from client 3, Timestamp: (19)
2024/10/22 14:15:37 Server received request: Name: 3, Message: hello from client 3, Timestamp: (19) at 20
2024/10/22 14:15:37 Server sent response: Name: 3, Message: hello from client 3, Timestamp: (21)
2024/10/22 14:15:37 Client received response: Name: 3, Message: hello from client 3, Timestamp: (21) at: 22
2024/10/22 14:15:37 Client received response: Name: 3, Message: hello from client 3, Timestamp: (21) at: 22
2024/10/22 14:15:37 Client received response: Name: 3, Message: hello from client 3, Timestamp: (21) at: 22
2024/10/22 14:17:05 Server received request: Name: 2, Message: has left the chat., Timestamp: (22) at 23
2024/10/22 14:17:05 Server sent response: Name: 2, Message: has left the chat., Timestamp: (24)
2024/10/22 14:17:05 Server removed client: 2, (25)
2024/10/22 14:17:05 Client received response: Name: 2, Message: has left the chat., Timestamp: (24) at: 25
2024/10/22 14:17:05 Client received response: Name: 2, Message: has left the chat., Timestamp: (24) at: 25
2024/10/22 14:17:05 Client received response: Name: 2, Message: has left the chat., Timestamp: (24) at: 25
2024/10/22 14:17:15 Client sent request: Name: 1, Message: hello from client 1 without client 2, Timestamp: (26)
2024/10/22 14:17:15 Server received request: Name: 1, Message: hello from client 1 without client 2, Timestamp: (26) at 27
2024/10/22 14:17:15 Server sent response: Name: 1, Message: hello from client 1 without client 2, Timestamp: (28)
2024/10/22 14:17:15 Client received response: Name: 1, Message: hello from client 1 without client 2, Timestamp: (28) at: 29
```

```
2024/10/22 14:17:15 Client received response: Name: 1, Message: hello from client 1 without client 2, Timestamp: (28) at: 29
2024/10/22 14:17:24 Client sent request: Name: 3, Message: hello from client 3 without client 2, Timestamp: (30)
2024/10/22 14:17:24 Server received request: Name: 3, Message: hello from client 3 without client 2, Timestamp: (30) at 31
2024/10/22 14:17:24 Server sent response: Name: 3, Message: hello from client 3 without client 2, Timestamp: (32)
2024/10/22 14:17:24 Client received response: Name: 3, Message: hello from client 3 without client 2, Timestamp: (32) at: 33
2024/10/22 14:17:24 Client received response: Name: 3, Message: hello from client 3 without client 2, Timestamp: (32) at: 33
2024/10/22 14:18:22 Client sent request: Name: 2, Message: has joined the chat., Timestamp: (25)
2024/10/22 14:18:22 Server added client: 2, (33)
2024/10/22 14:18:22 Server received request: Name: 2, Message: has joined the chat., Timestamp: (25) at 34
2024/10/22 14:18:22 Server sent response: Name: 2, Message: has joined the chat., Timestamp: (35)
2024/10/22 14:18:22 Client received response: Name: 2, Message: has joined the chat., Timestamp: (35) at: 36
2024/10/22 14:18:22 Client received response: Name: 2, Message: has joined the chat., Timestamp: (35) at: 36
2024/10/22 14:18:22 Client received response: Name: 2, Message: has joined the chat., Timestamp: (35) at: 36
2024/10/22 14:18:33 Client sent request: Name: 1, Message: hello with all 3 clients again, Timestamp: (37)
2024/10/22 14:18:33 Server received request: Name: 1, Message: hello with all 3 clients again, Timestamp: (37) at 38
2024/10/22 14:18:33 Server sent response: Name: 1, Message: hello with all 3 clients again, Timestamp: (39)
2024/10/22 14:18:33 Client received response: Name: 1, Message: hello with all 3 clients again, Timestamp: (39) at: 40
2024/10/22 14:18:33 Client received response: Name: 1, Message: hello with all 3 clients again, Timestamp: (39) at: 40
2024/10/22 14:18:33 Client received response: Name: 1, Message: hello with all 3 clients again, Timestamp: (39) at: 40
```

Logs for sequence of rpc calls diagram with 2 clients:

```
2024/10/24 14:49:24 Server started on :5050
2024/10/24 14:50:26 Server added client: Client 1, (1)
2024/10/24 14:50:26 Server received request: Name: Client 1, Message: has joined the chat., Timestamp: (0) at 2
2024/10/24 14:50:26 Server sent response: Name: Client 1, Message: has joined the chat., Timestamp: (3)
```

```
2024/10/24 14:50:26 Client sent request: Name: Client 1, Message: has joined the chat., Timestamp: (0)
2024/10/24 14:50:26 Client received response: Name: Client 1, Message: has joined the chat., Timestamp: (3) at: 4
2024/10/24 14:50:36 Client sent request: Name: Client 1, Message: hej, Timestamp: (5)
2024/10/24 14:50:36 Server received request: Name: Client 1, Message: hej, Timestamp: (5) at 6
2024/10/24 14:50:36 Server sent response: Name: Client 1, Message: hej, Timestamp: (7)
2024/10/24 14:50:36 Client received response: Name: Client 1, Message: hej, Timestamp: (7) at: 8
2024/10/24 14:50:49 Server added client: Client 2, (8)
2024/10/24 14:50:49 Server received request: Name: Client 2, Message: has joined the chat., Timestamp: (0) at 9
2024/10/24 14:50:49 Server sent response: Name: Client 2, Message: has joined the chat., Timestamp: (10)
2024/10/24 14:50:49 Client received response: Name: Client 2, Message: has joined the chat., Timestamp: (10) at: 11
2024/10/24 14:50:49 Client sent request: Name: Client 2, Message: has joined the chat., Timestamp: (0)
2024/10/24 14:50:49 Client received response: Name: Client 2, Message: has joined the chat., Timestamp: (10) at: 11
```