Handin 3 - ChittyChat

Erik Malmqvist Jakobsen

November 1, 2022

1 Preface

This document accompanies the small application found at `https://github.com/emjakobsen1/dsys`, made as an assignment for the course Distributed Systems at IT-University of Copenhagen as part of BSc in Software Development.

ChittyChat is a console-based chat service that lets multiple clients communicate through a server. It is implemented in Go and uses gRPC for passing messages between participants. It computes and logs Lamport timestamps throughout the lifetime of clients.

2 System architecture

The chat application is a server-client architecture. Clients can join and communicate through the server which is prefixed to port:8080. The application uses server-side streaming.

The clients are calling the server to set up the connection to the server. The server provides a stream to read a sequence of messages. The clients then read the timestamps and messages from the returned stream, including messages that the client itself broadcasted.

The following messages and RPC methods are used:

- **User**. Represents the client node. Clients are made with a random ID, optionally a name and with and with an initial lamport timestamp set to 0.
- **Message**. Of type stream. Contains the ID associated with the client sending the message, the content of the message itself, the lamport timestamp at which the message was sent from the client node aswell as the name of the client.
- **Connect**. The message that server receives when a client connects to the server.
- **Close**. An empty message.
- **CreateStream**. Sets up the connection with the Connect message, and returns a stream of messages to the client.
- **BroadcastMessage**. Describes how we return the messages from the server to the client. Returns close when finished with broadcasting.

3 Lamport timestamps

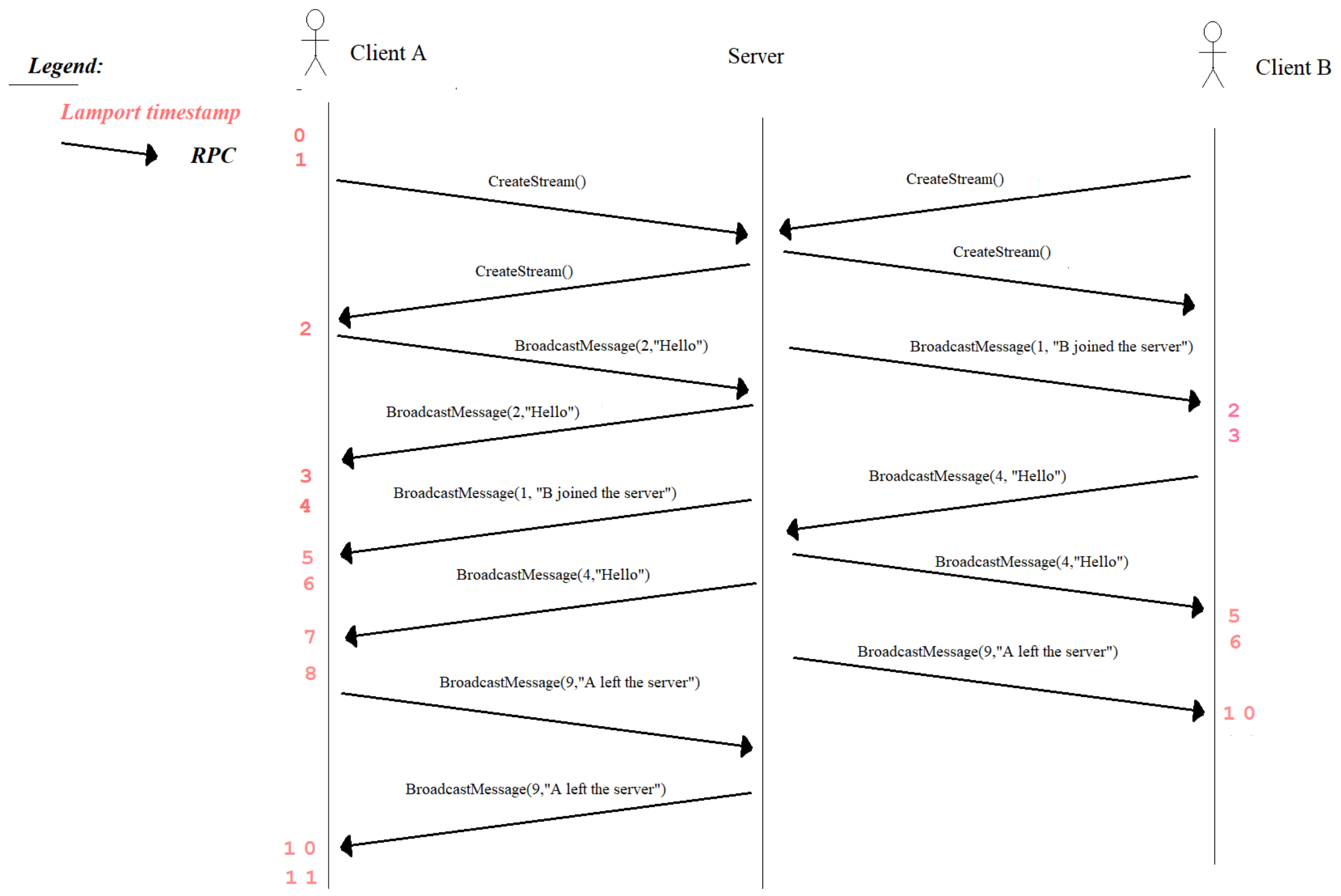
Each client node is initialised with a timestamp of `t := 0`. When the client joins, it announces itself with a message broadcast which increments its local, `t := t + 1`.

Now the client can start chatting. Whenever a client broadcasts a message its local `t` is updated to `t := t + 1` and that `t` is forwarded with the message.

The clients receives its own messages as well as other clients messages from the stream, at each such event it computes the Lamport timestamp to be the maximum t of either the received message or its local `t`, `t := max(t, t') + 1`.

4 Scenarios

A diagram illustrating the scenario: Client A joins, Client A publishes, Client B joins, Client B publishes, Client A leaves. The arrows represents RPC calls and the red numbers each clients Lamport timestamp. A system log is included in the Appendix (6.1: Scenario 1).


Legend:
Lamport timestamp
RPC
Client A
Server
Client B
0
1
CreateStream()
CreateStream()
CreateStream()
CreateStream()
2
BroadcastMessage(2,"Hello")
BroadcastMessage(1, "B joined the server")
BroadcastMessage(2,"Hello")
2
3
3
4
BroadcastMessage(1, "B joined the server")
BroadcastMessage(4, "Hello")
5
6
BroadcastMessage(4,"Hello")
BroadcastMessage(4,"Hello")
7
5
6
8
BroadcastMessage(9,"A left the server")
BroadcastMessage(9,"A left the server")
10
BroadcastMessage(9,"A left the server")
10
11

Another scenario: 4 Clients join, 1 leaves is played out. Four participants, Anders, Bo Cesie and Dirk joins the chat service. Bo types "Hey" and Cesie leaves. Whenever a participant joins the server, participants already connected to the server gets notified. When a participant leaves, all other participants receives the message. See Appendix (6.2) for a system log of the event.

5 Limitations

- **R5**. The application frequently crashes when new clients connects to the server and old clients leave. gRPC outputs the error: rpc error: code = Unavailable desc = transport is closing. It is possible that the error is caused by keeping the connections in a slice on the server side, but it is also possible the error happens on the client side. Thus, the requirement ”Chat clients can join at any time” is not met.
- **R6**. When a new participant join the chat, it is broadcasted to all other participants including the new participant. Although, sometimes the new participant cannot see the broadcasted message, but all the others can. For the requirement ”A ”Participant X joined Chitty-Chat at Lamport time L” message is broadcast to all Participants when client X joins, including the new Participant.”, the application does not always include the new participant.

6 Appendix

6.1 Scenario 1: Client A joins, Client A publishes, Client B joins, Client B publishes, Client A leaves

6.1.1 Server log

```
2022/11/01 10:48:36 Starting Chitty-Chat at port :8080
2022/11/01 10:48:36 RegisterBroadcastServer:  &{{<nil> <nil> <nil> <nil> <nil> <nil
    > [] [] <nil> <nil> [] 0 4194304 2147483647 <nil> {0 0 0 0 0} {0 false} 0 0
    32768 32768 120000000000 <nil> <nil> 0} {0 0} map[] map[] false false 0xc00006c9
    c0 map[proto.Broadcast:0xc000196b10] <nil> 0xc0000628e0 0xc000062900 {0 {0 0}}
    {{} 0 0} 0xc000196a20 0xc000024540 []} &{[] {}}
2022/11/01 10:48:42 Close:
2022/11/01 10:48:42 CreateStream:  &{0xc000061f00 85 true 0xc00002e480}
2022/11/01 10:48:44 Broadcasting message to:  &{0xc0001be000} id:"85"  content:":
    Hello"  timestamp:2  userName:"A"
2022/11/01 10:48:44 Close:
2022/11/01 10:48:50 CreateStream:  &{0xc000210040 65 true 0xc00021c000}
2022/11/01 10:48:50 Broadcasting message to:  &{0xc0000a60e0} id:"65"  content:"
    joined Chitty-Chat server."  timestamp:1  userName:"B"
2022/11/01 10:48:50 Broadcasting message to:  &{0xc0001be000} id:"65"  content:"
    joined Chitty-Chat server."  timestamp:1  userName:"B"
2022/11/01 10:48:50 Close:
2022/11/01 10:48:56 Broadcasting message to:  &{0xc0000a60e0} id:"65"  content:":
    Hello"  timestamp:4  userName:"B"
2022/11/01 10:48:56 Broadcasting message to:  &{0xc0001be000} id:"65"  content:":
    Hello"  timestamp:4  userName:"B"
2022/11/01 10:48:56 Close:
2022/11/01 10:49:06 Broadcasting message to:  &{0xc0000a60e0} id:"85"  content:"
    left the server."  timestamp:9  userName:"A"
2022/11/01 10:49:06 Broadcasting message to:  &{0xc0001be000} id:"85"  content:"
    left the server."  timestamp:9  userName:"A"
2022/11/01 10:49:06 Close:
2022/11/01 10:49:06 Broadcasting message to:  &{0xc0001be000} id:"85"  content:":-
    close"  timestamp:9  userName:"A"
2022/11/01 10:49:06 Broadcasting message to:  &{0xc0000a60e0} id:"85"  content:":-
    close"  timestamp:9  userName:"A"
```

```
2022/11/01 10:49:06 Close:
```

6.1.2 Client A Standard out

```
go run main.go -name A
Hello
A #85 (T = 4):Hello
B #65 (T = 6) joined Chitty-Chat server.
B #65 (T = 8):Hello
-close
A #85 (T = 11) left the server.
```

6.1.3 Client B Standard out

```
go run main.go -name B
B #65 (T = 3) joined Chitty-Chat server.
Hello
B #65 (T = 6):Hello
A #85 (T = 10) left the server.
A #85 (T = 12):-close
```

6.2 Scenario 2: 4 Clients join the server, 1 leaves

6.2.1 System log

```
2022/11/01 18:30:32 Starting Chitty-Chat at port :8080
2022/11/01 18:30:32 RegisterBroadcastServer:  &{{<nil> <nil> <nil> <nil> <nil> <nil
   > [] [] <nil> <nil> [] 0 4194304 2147483647 <nil> {0 0 0 0 0} {0 false} 0 0
   32768 32768 120000000000 <nil> <nil> 0} {0 0} map[] map[] false false 0xc0000e
   0980 map[proto.Broadcast:0xc000164ae0] <nil> 0xc0000a08c0 0xc0000a08e0 {0 {0 0}}
    {{} 0 0} 0xc0001649f0 0xc0000a4400 []} &{[] {}}
2022/11/01 18:30:36 Close:
2022/11/01 18:30:36 CreateStream:  &{0xc000060350 89 true 0xc00002e1e0}
2022/11/01 18:30:41 CreateStream:  &{0xc000202370 81 true 0xc00022a060}
2022/11/01 18:30:41 Broadcasting message to:  &{0xc0000780e0} id:"81"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Bo"
2022/11/01 18:30:41 Broadcasting message to:  &{0xc0000782a0} id:"81"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Bo"
2022/11/01 18:30:41 Close:
2022/11/01 18:30:44 CreateStream:  &{0xc000202460 81 true 0xc00022a240}
2022/11/01 18:30:44 Broadcasting message to:  &{0xc0000780e0} id:"81"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Cesie"
2022/11/01 18:30:44 Broadcasting message to:  &{0xc0000782a0} id:"81"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Cesie"
2022/11/01 18:30:44 Broadcasting message to:  &{0xc0003ae000} id:"81"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Cesie"
2022/11/01 18:30:44 Close:
2022/11/01 18:31:28 CreateStream:  &{0xc0001b2f40 42 true 0xc000086780}
2022/11/01 18:31:28 Broadcasting message to:  &{0xc0000782a0} id:"42"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Dirk"
2022/11/01 18:31:28 Broadcasting message to:  &{0xc0003ae000} id:"42"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Dirk"
```

```
2022/11/01 18:31:28 Broadcasting message to:  &{0xc0000780e0} id:"42"  content:"
   joined Chitty-Chat server."  timestamp:1  userName:"Dirk"
2022/11/01 18:31:28 Close:
2022/11/01 18:31:49 Broadcasting message to:  &{0xc0000780e0} id:"81"  content:":
   Hey"  timestamp:8  userName:"Bo"
2022/11/01 18:31:49 Broadcasting message to:  &{0xc0000782a0} id:"81"  content:":
   Hey"  timestamp:8  userName:"Bo"
2022/11/01 18:31:49 Broadcasting message to:  &{0xc0003ae000} id:"81"  content:":
   Hey"  timestamp:8  userName:"Bo"
2022/11/01 18:31:49 Broadcasting message to:  &{0xc00025c0e0} id:"81"  content:":
   Hey"  timestamp:8  userName:"Bo"
2022/11/01 18:31:49 Close:
2022/11/01 18:32:14 Broadcasting message to:  &{0xc0003ae000} id:"81"  content:"
   left the server."  timestamp:10  userName:"Cesie"
2022/11/01 18:32:14 Broadcasting message to:  &{0xc00025c0e0} id:"81"  content:"
   left the server."  timestamp:10  userName:"Cesie"
2022/11/01 18:32:14 Broadcasting message to:  &{0xc0000782a0} id:"81"  content:"
   left the server."  timestamp:10  userName:"Cesie"
2022/11/01 18:32:14 Broadcasting message to:  &{0xc0000780e0} id:"81"  content:"
   left the server."  timestamp:10  userName:"Cesie"
2022/11/01 18:32:14 Close:
```

6.2.2 Client Anders Standard out

```
go run main.go -name Anders
Bo #81 (T = 3) joined Chitty-Chat server.
Cesie #81 (T = 5) joined Chitty-Chat server.
Dirk #42 (T = 7) joined Chitty-Chat server.
Bo #81 (T = 9):Hey
Cesie #81 (T = 11) left the server.
```

6.2.3 Client Bo Standard out

```
go run main.go -name Bo
Bo #81 (T = 3) joined Chitty-Chat server.
Cesie #81 (T = 5) joined Chitty-Chat server.
Dirk #42 (T = 7) joined Chitty-Chat server.
Hey
Bo #81 (T = 10):Hey
Cesie #81 (T = 12) left the server.
```

6.2.4 Client Cesie Standard out

```
go run main.go -name Cesie
Cesie #81 (T = 3) joined Chitty-Chat server.
Dirk #42 (T = 5) joined Chitty-Chat server.
Bo #81 (T = 9):Hey
-close
Cesie #81 (T = 12) left the server.
```

6.2.5 Client Dirk Standard out

```
go run main.go -name Dirk
Bo #81 (T = 9):Hey
Cesie #81 (T = 11) left the server.
```