DESIGN DOCUMENTATION
A TINY SOCIAL NETWORK SERVICE (SNS)

Date: 24th October, 2023
Written By: Vaibhav Pundir

Introduction

The objective of this document is to describe how to approach the requirements of the Machine Problem 2.1 and come up with their solution.

System Overview

The Tiny Social Networking Service will have the following components:

- Coordinator
- Server
- Client

Design Considerations

- **Assumptions**
  - As described in the assignment, there are no coordinator failures.
- **Dependencies**
  - The Tiny SNS will be developed on Ubuntu 22.04.3
  - Google Protocol Buffers - v3 and gRPC
  - g++ version 11.4.0

Architectural Strategies

The architecture here is GRPC architecture where we have a coordinator, server and a client side. The client contacts the coordinator to get the server details. The server registers itself to the coordinator and sends keep-alive messages to the coordinator.
Tech Stack: C++ | Makefile | Bash

System Architecture

The system consists of a coordinator, servers & clients.

Coordinator

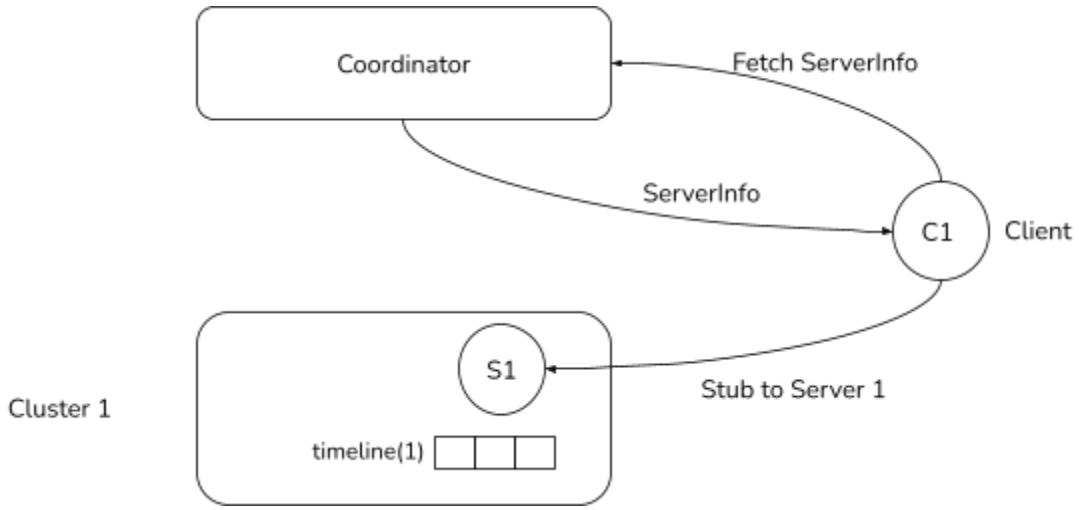
I. Responsible for mapping each client to a server. The mapping logic is $(clientID - 1) \% 3 + 1$.
II. Registering each server and keeping their status via heartbeat.

Server

I. Invokes Create api on the coordinator side to register itself.
II. Sends heartbeat messages to the Coordinator.
III. Handles client's requests.

Client

I. Requests server details from the Coordinator via coordinator stub.
II. Creates server stub via the ServerInfo received from Coordinator and executes user entered commands.



Detailed System Design

1. The coordinator has the following modules:

   Create

   Each server registers itself to the Coordinator via the Create api.

Heartbeat

Each server sends heartbeat messages to the coordinator every three seconds to let the coordinator know that it is still alive.

GetServer

Each client requests the server details from the Coordinator by calling this api.

Exists

Checks if the master has been elected in a cluster or not.

2. The server has two new functions:

keep_alive()

Sends heartbeat messages to the coordinator every three seconds.

register_server_with_coordinator()

Invokes Create api from the Coordinator and registers itself.

3. The client now connects to the server in the following way:
   - Create coordinator stub
   - Use GetServer api to get server details
   - Create server stub to connect to the server

Commands to run test cases

Test case 1 (Open 3 terminals)

1st terminal → ./coordinator -p 9090
2nd terminal → ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
Wait for 5 seconds
3rd terminal → ./tsc -h localhost -k 9090 -u 1
3rd terminal → list
3rd terminal → timeline

Test case 2 (Open 3 terminals)

1st terminal → ./coordinator -p 9090
2nd terminal → ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
//Wait for 5 seconds
2nd terminal → ctrl+c
//Wait for 1 second
3rd terminal → ./tsc -h localhost -k 9090 -u 1
//Wait for 5 second
3rd terminal → ./tsc -h localhost -k 9090 -u 1
2nd terminal → ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
//Wait for 5 second
3rd terminal → ./tsc -h localhost -k 9090 -u 1
3rd terminal → list
3rd terminal → timeline

Test case 3 (Open 5 terminals)

1st terminal → ./coordinator -p 9090
2nd terminal → ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
3rd terminal → ./tsd -c 2 -s 1 -h localhost -k 9090 -p 9290
//Wait for 5 seconds
4th terminal → ./tsc -h localhost -k 9090 -u 1
5th terminal → ./tsc -h localhost -k 9090 -u 2
3rd terminal → ctrl+c
//Wait for 1 second
5th terminal → list
5th terminal → timeline
//Wait for 5 seconds
4th terminal → list
4th terminal → timeline
5th terminal → list
5th terminal → timeline
// Make sure there is a total wait time of ~10 seconds b/w crashing
// and restarting cluster 2's server, since coordinator marks server
// in-active after 10 seconds of missing heartbeat and attempting to
// re-connect server before coordinator marks it as in-active should
// fail.
3rd terminal → ./tsd -c 2 -s 1 -h localhost -k 9090 -p 9290
//Wait for 5 seconds
5th terminal → list
5th terminal → timeline

Test Screenshots (Test case 1)

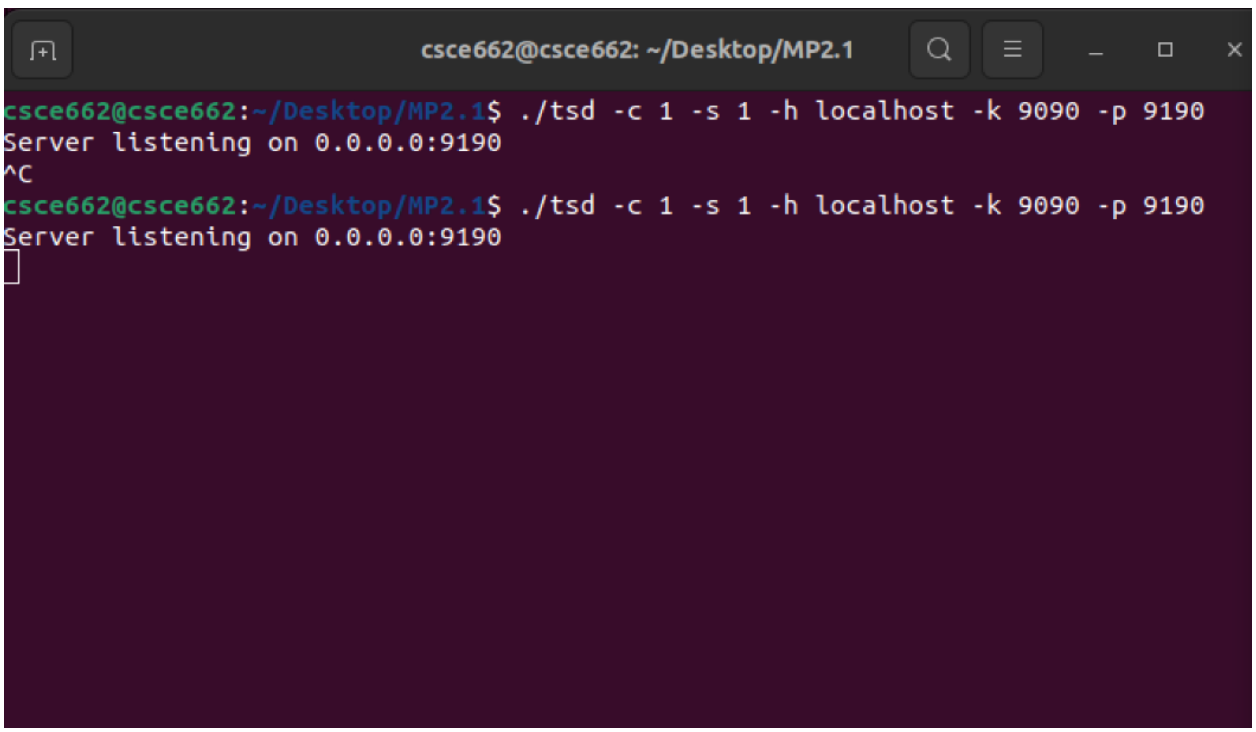
```
csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ ./coordinator -p 9090
Server listening on 127.0.0.1:9090
```

```
csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
Server listening on 0.0.0.0:9190
```

```
csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ sleep 5
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 1
Logging Initialized. Client starting...REPLY MESSAGE: Login Successful!
========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
====================================
Cmd> list
Command completed successfully
All users: 1,
Followers:
Cmd> timeline
Command completed successfully
Now you are in the timeline
p1
p2
p3
```

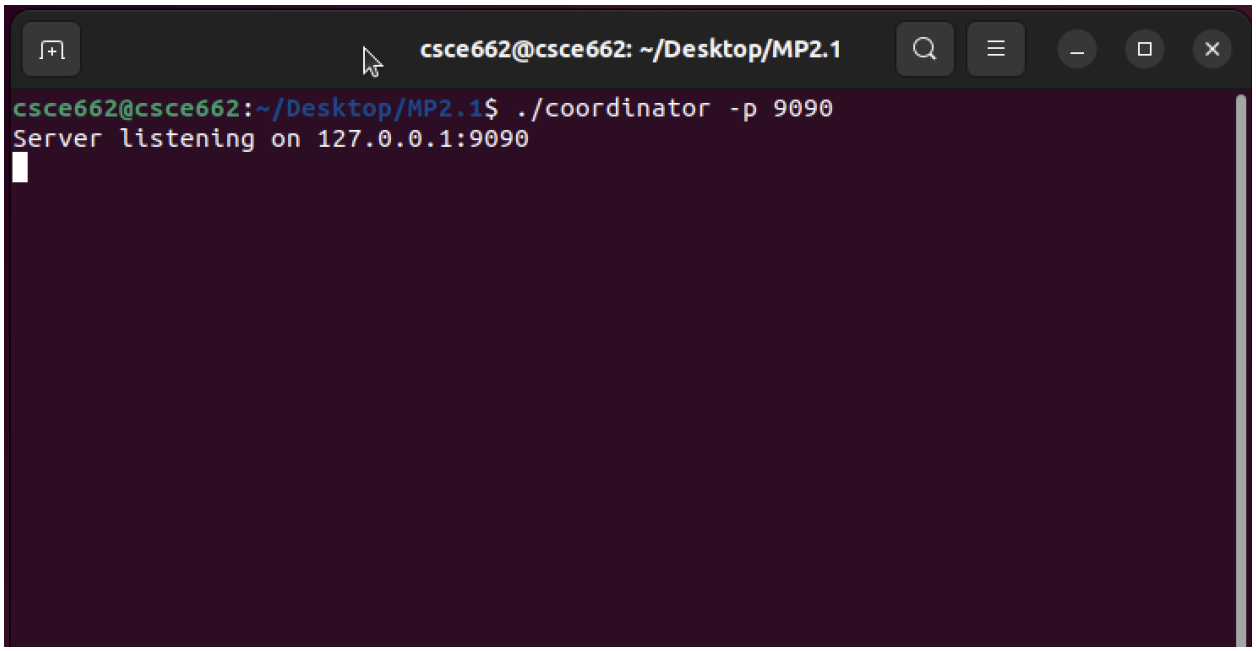
Test case 2

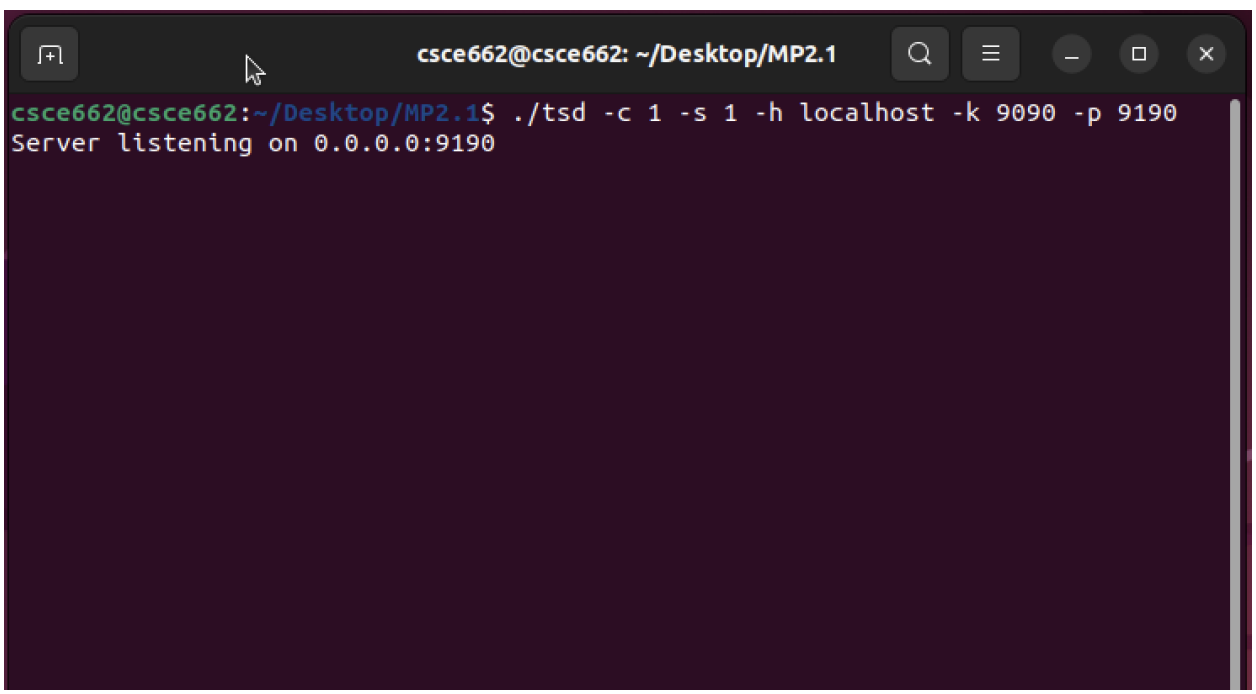

csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ ./coordinator -p 9090
Server listening on 127.0.0.1:9090


csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
Server listening on 0.0.0.0:9190
^C
csce662@csce662:~/Desktop/MP2.1$ ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
Server listening on 0.0.0.0:9190

```
csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ sleep 5
csce662@csce662:~/Desktop/MP2.1$ echo "kill server 1"
kill server 1
csce662@csce662:~/Desktop/MP2.1$ sleep 1
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 1
Logging Initialized. Client starting...
REPLY MESSAGE: connection failed: -1
csce662@csce662:~/Desktop/MP2.1$ sleep 5
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 1
Logging Initialized. Client starting...
connection failed: -1
csce662@csce662:~/Desktop/MP2.1$ echo "restart server 1"
restart server 1
csce662@csce662:~/Desktop/MP2.1$ sleep 5
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 1
Logging Initialized. Client starting...
REPLY MESSAGE: Login Successful!
========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
====================================
Cmd> list
Command completed successfully
All users: 1,
Followers:
Cmd> timeline
Command completed successfully
Now you are in the timeline
p1
p2
p3
```

Test case 3


csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ ./coordinator -p 9090
Server listening on 127.0.0.1:9090


csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~/Desktop/MP2.1$ ./tsd -c 1 -s 1 -h localhost -k 9090 -p 9190
Server listening on 0.0.0.0:9190

csce662@csce662: ~/Desktop/MP2.1

```
csce662@csce662:~/Desktop/MP2.1$ ./tsd -c 2 -s 1 -h localhost -k 9090 -p 9290
Server listening on 0.0.0.0:9290
^C
csce662@csce662:~/Desktop/MP2.1$ ./tsd -c 2 -s 1 -h localhost -k 9090 -p 9290
Server listening on 0.0.0.0:9290
```

csce662@csce662: ~/Desktop/MP2.1

```
csce662@csce662:~/Desktop/MP2.1$ sleep 5
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 1
Logging Initialized. Client starting...
REPLY MESSAGE: Login Successful!
========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
====================================
Cmd> list
Command completed successfully
All users: 1,
Followers:
Cmd> timeline
Command completed successfully
Now you are in the timeline
p1
p2
p3
```

```
csce662@csce662: ~/Desktop/MP2.1
csce662@csce662:~$ cd Desktop/MP2.1/
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 2
Logging Initialized. Client starting...
REPLY MESSAGE: Login Successful!
========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
====================================
Cmd> list
Command failed
Cmd> timeline
Command failed
Cmd> list
Command failed
Cmd> timeline
Command failed
Cmd> ^C
csce662@csce662:~/Desktop/MP2.1$ ./tsc -h localhost -k 9090 -u 2
Logging Initialized. Client starting...
REPLY MESSAGE: Login Successful!
========= TINY SNS CLIENT =========
 Command Lists and Format:
 FOLLOW <username>
 UNFOLLOW <username>
 LIST
 TIMELINE
====================================
Cmd> list
Command completed successfully
All users: 2,
Followers:
Cmd> timeline
Command completed successfully
Now you are in the timeline
p4
p5
p6
```