

A Simple Peer To Peer Network Implementation

Hoor Abootalebi
Nariman Aryan
Amin Isaai
Amirhossein Khajepour
Mahdis Tajdari
Ali Zeynali

November 2018

Contents

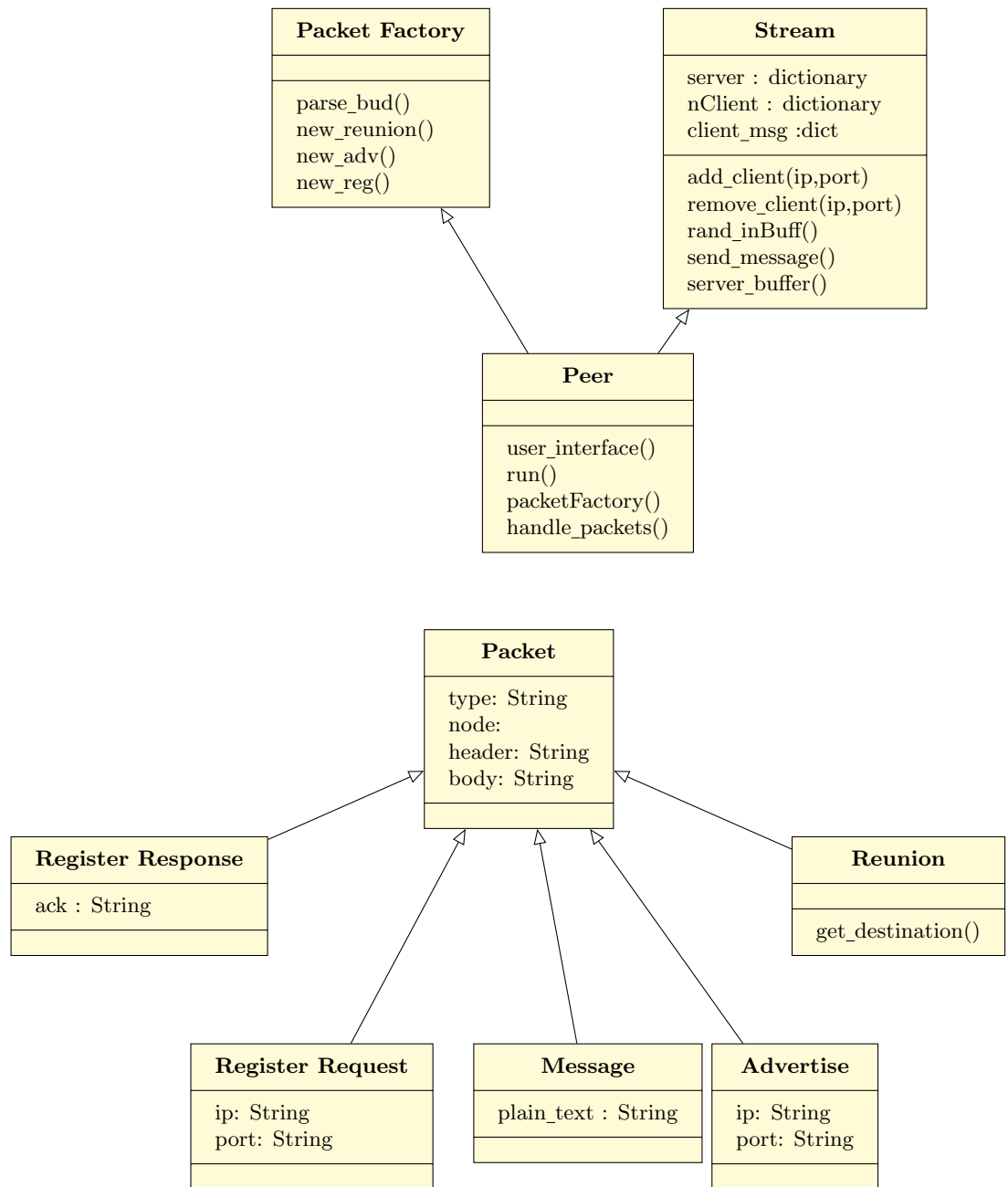
1	Introdeuction	3
2	UML model	3
3	Objects	4
3.1	Streem	5
3.2	Peer	5
3.3	Packet Factory	5
3.4	Packet	5
3.5	Reunion	6
3.6	Node	6
3.7	Resgister Request	6
3.8	Register Response	6
3.9	Advertise	6
3.10	Mesasge	6

1 Introdeuction

This project aims to implement a peer to peer network. In the first step we design UML model and then we are going to explain each objects' attributes and methods.

2 UML model

We design the UML model in order to make the project more understandable, clearer and professional.



3 Objects

Now it's time to explain every object's duty.

3.1 Stream

```
#stream()
add_client(ip,port)
remove_client(ip,port)
send_message()
rand_inBuff()
server_buffer()
```

We also need to add a dictionary to specify every client's message(s)

3.2 Peer

```
#Peer()
stream()
user_interface() #Which the user or client sees and works with.
run()           #This method runs every time to see
                #whether there is new messages or not.
packetFactory()
handle_packets()
```

3.3 Packet Factory

`packetFactory()` would generate the packets every node needs to connect another with.

```
#packetFactory
parse_buf()
new_reunion()
new_adv() #makes a new advertise packet.
new_reg() #makes a new register packet.
```

3.4 Packet

Every packet consists seven different parts: **plain_text** which is the raw text message in the packet.

Node: Specifies to which node the packet sent to. **Sender** specifies who sent the packet **Validator** which makes the packet valid.

Header where the information such as type of the packet and etc.

are going to be there.

Body body of our packet **Action** says what the packet does.

3.5 Reunion

`reunion(packet)` checks the connection of the nodes to the root.

```
#reunion(packet)
get_destination()
```

3.6 Node

Every node has two parameters: **IP** and **Port**.

3.7 Register Request

`reg_req()` sends IP/Port of a node to the root to ask if it can register it.

3.8 Register Response

`reg_res()` should just send an from the root *Ack* to inform a node that it has been registerd in the root if the `reg_req()` was successful.

3.9 Advertise

`adv(packet)`

3.10 Mesasge

`msg(packet)`