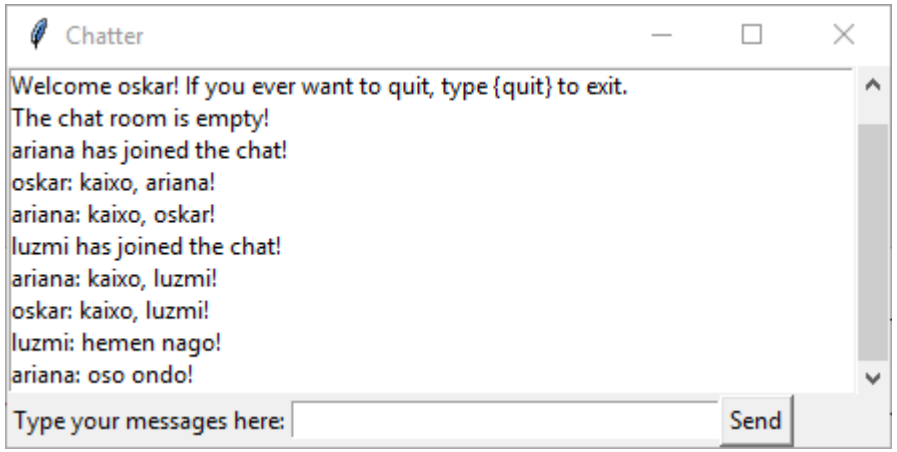
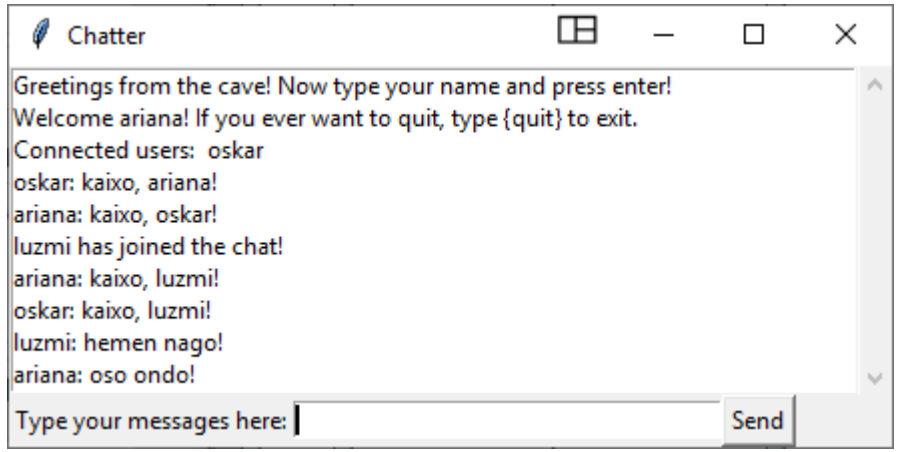
Konputagailu Sareen Oinarriak 2019/2020 ikasturtea

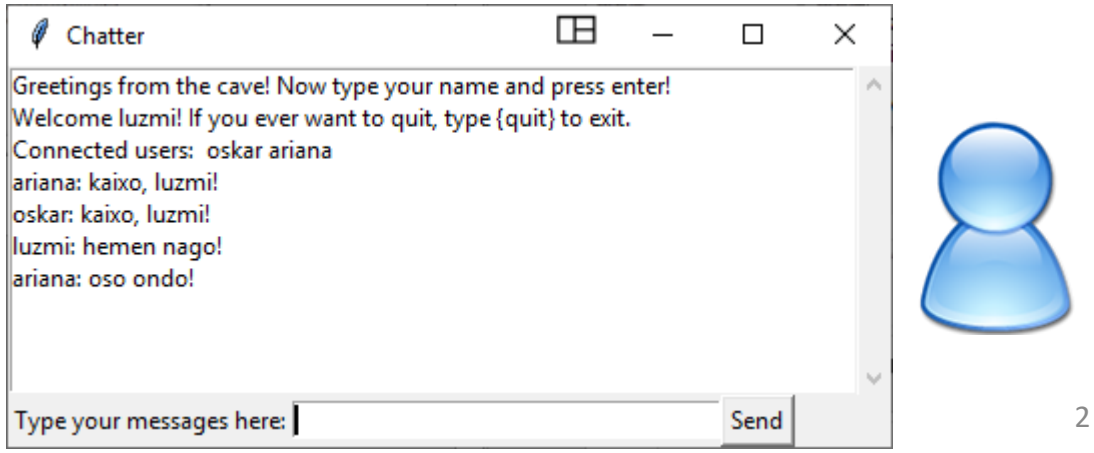
3. Praktika: Socket-ak Chat baten programazioa

Helburua

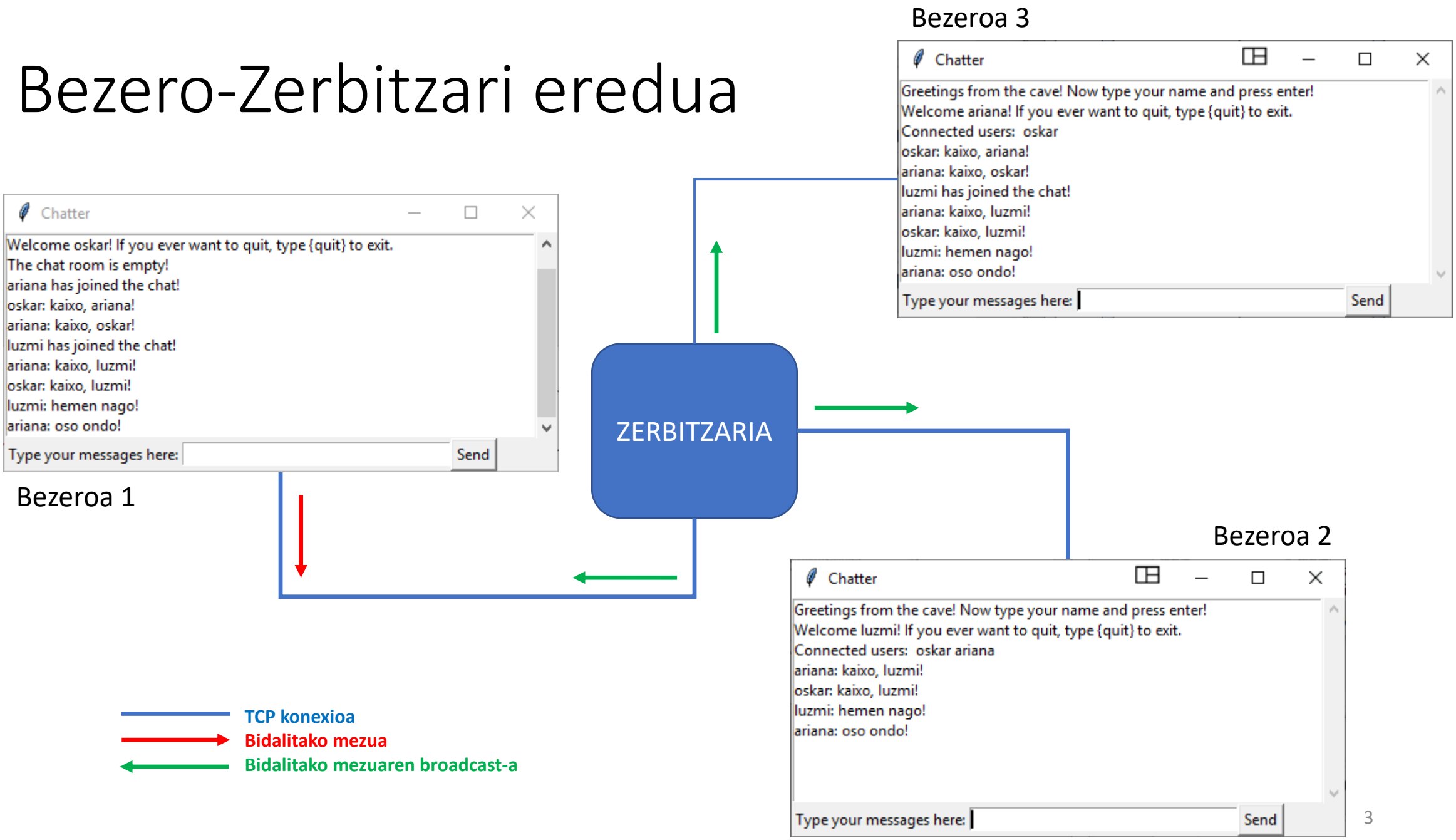

Chatter
Welcome oskar! If you ever want to quit, type {quit} to exit.
The chat room is empty!
ariana has joined the chat!
oskar: kaixo, ariana!
ariana: kaixo, oskar!
luzmi has joined the chat!
ariana: kaixo, luzmi!
oskar: kaixo, luzmi!
luzmi: hemen nago!
ariana: oso ondo!
Type your messages here:
Send


Live Chat

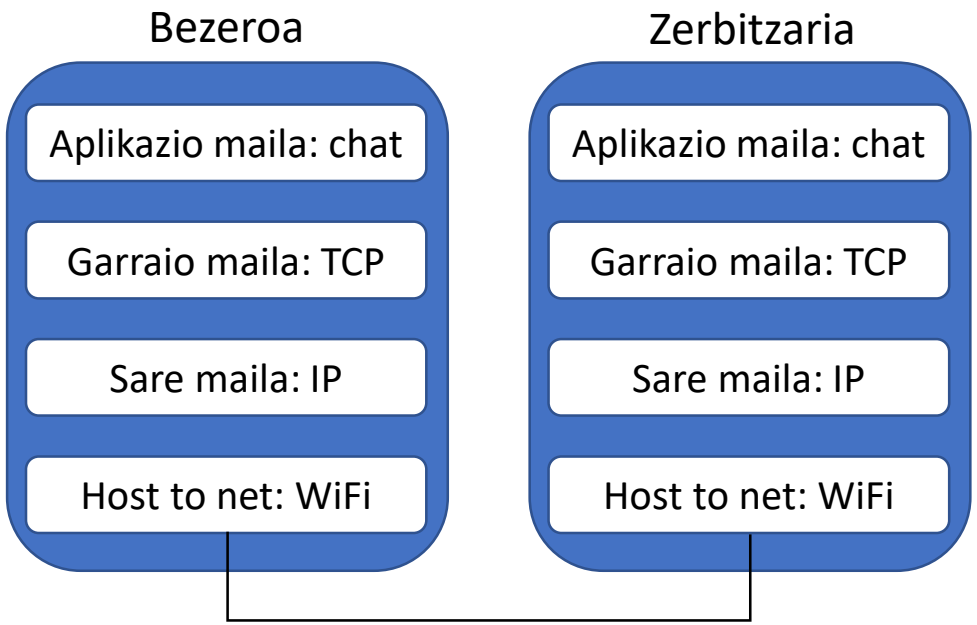

Chatter
Greetings from the cave! Now type your name and press enter!
Welcome ariana! If you ever want to quit, type {quit} to exit.
Connected users: oskar
oskar: kaixo, ariana!
ariana: kaixo, oskar!
luzmi has joined the chat!
ariana: kaixo, luzmi!
oskar: kaixo, luzmi!
luzmi: hemen nago!
ariana: oso ondo!
Type your messages here:
Send


Chatter
Greetings from the cave! Now type your name and press enter!
Welcome luzmi! If you ever want to quit, type {quit} to exit.
Connected users: oskar ariana
ariana: kaixo, luzmi!
oskar: kaixo, luzmi!
luzmi: hemen nago!
ariana: oso ondo!
Type your messages here:
Send

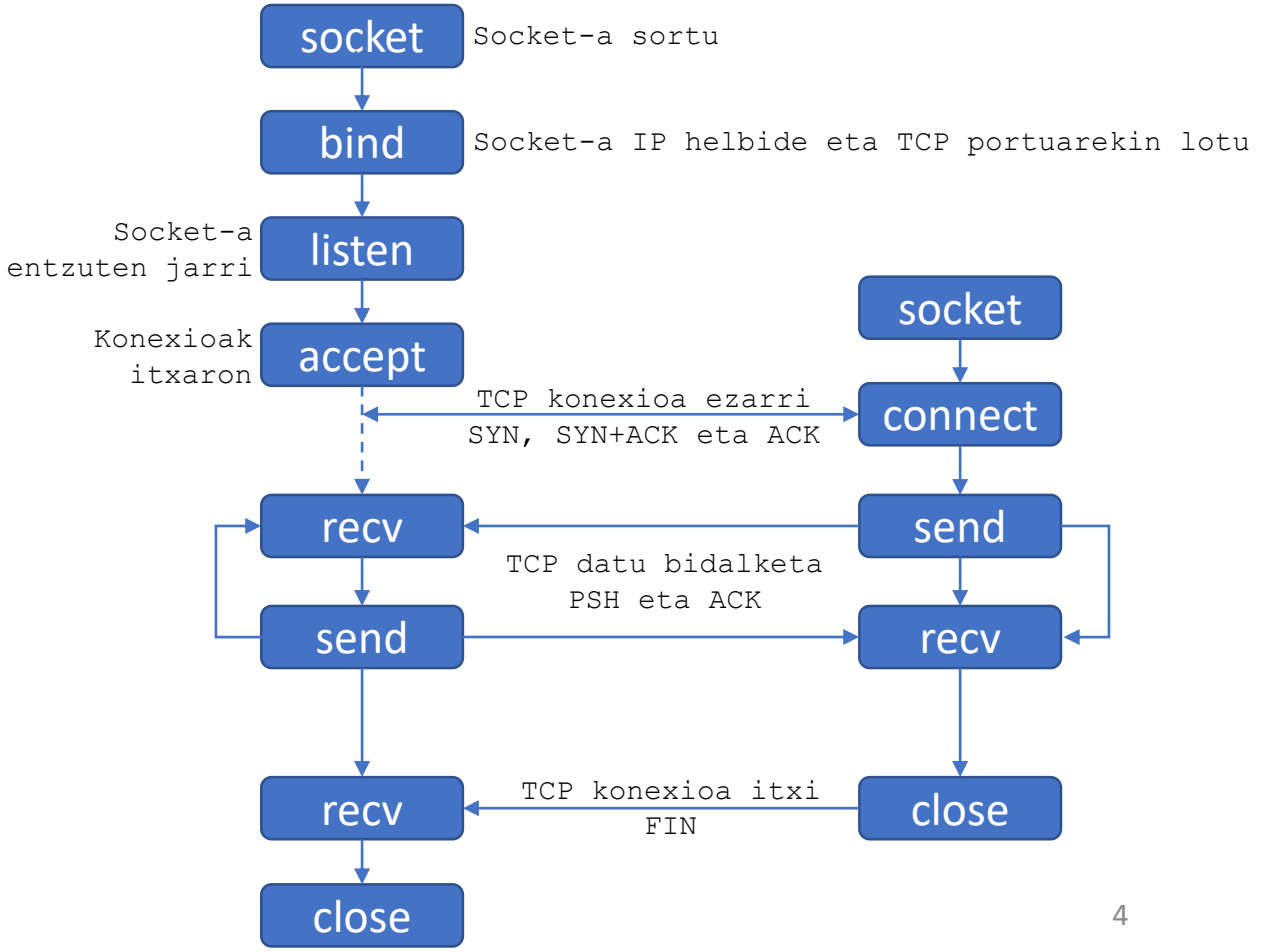
Bezero-Zerbitzari eredua



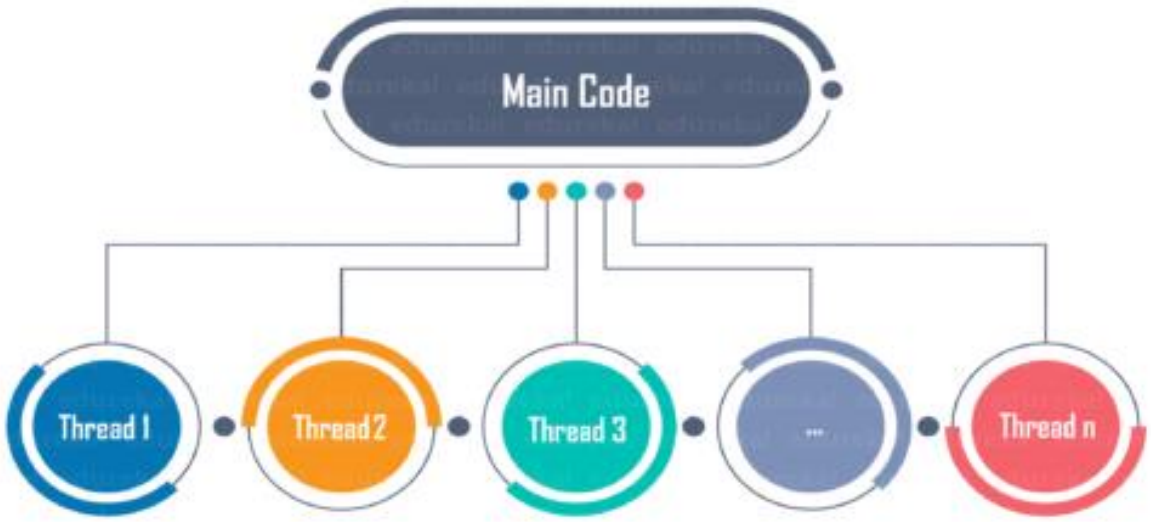
Bezero eta zerbitzariaren arteko komunikazioa



socket — Low-level networking interface
https://docs.python.org/3/library/socket.html



Bezeroa eta zerbitzaria hari-anitzdunak dira



threading — Thread-based parallelism
https://docs.python.org/3/library/threading.html

Hariak ataza baten exekuzioak itxoite denborak azaltzen dituen kasuetan erabilgarriak dira. Tarte horietan hariak beste kode bat exekutatu dezake prozesu nagusia edo beste hari bat itxaroten dagoen bitartean.

Haririk gabe: begiztaren iterazio bakoitzak aurrekoa amaitua daiten itxaron behar du

```
import time

for i in range(1, 11):
    delay = 11-i
    print("Thread-" + str(i) + " lotan " + str(delay) + " s")
    time.sleep(delay)
    print("Thread-" + str(i) + " finished")
```

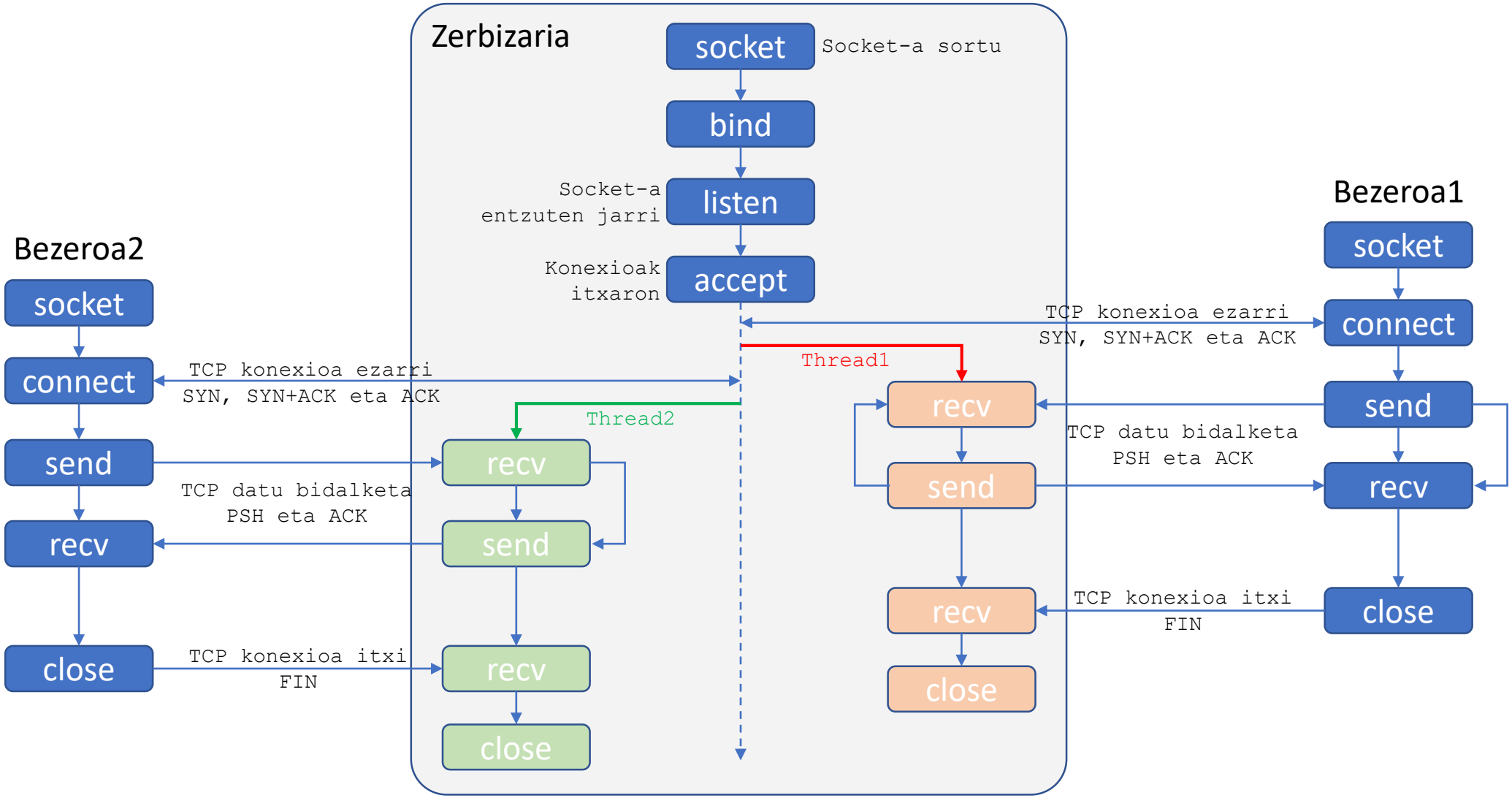
Hariekin: begiztaren iterazio baikotza aurrekoa amaitu gabe hasi daiteke

```
import threading
import time

def my_thread(i, delay):
    print("Thread-" + str(i) + " lotan " str(delay) + " s")
    time.sleep(delay)
    print("Thread-" + str(i) + " finished")

for i in range(1, 11):
    threading.Thread(target=my_thread, args=(i, 11-i, )).start()
```

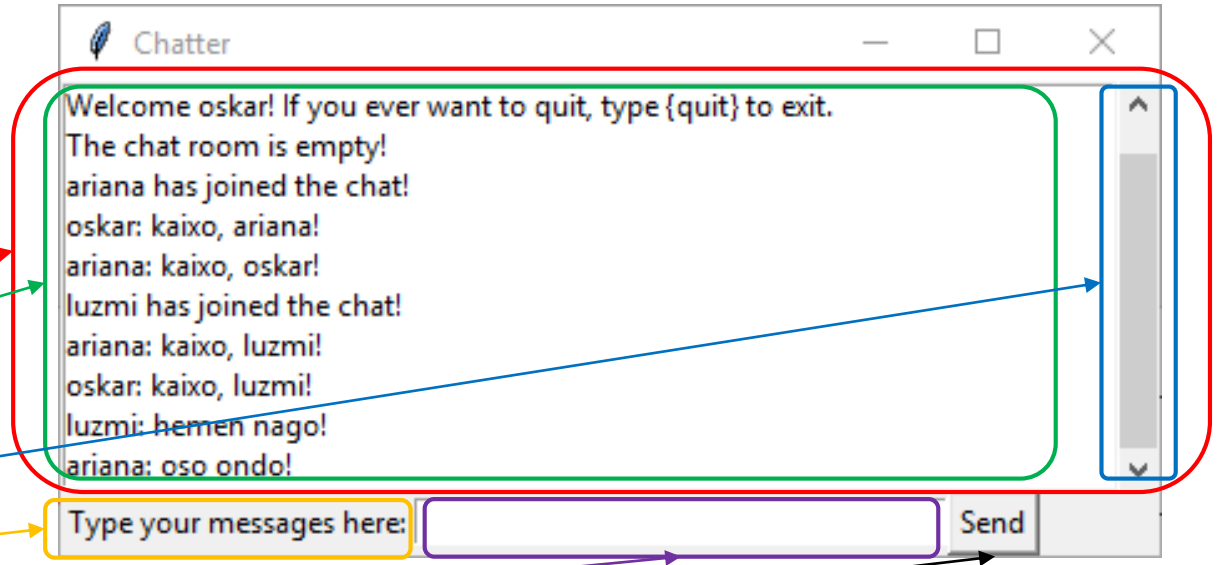
Bezeroa eta zerbitzaria hari-anitzdunak dira



Bezeroaren egitura

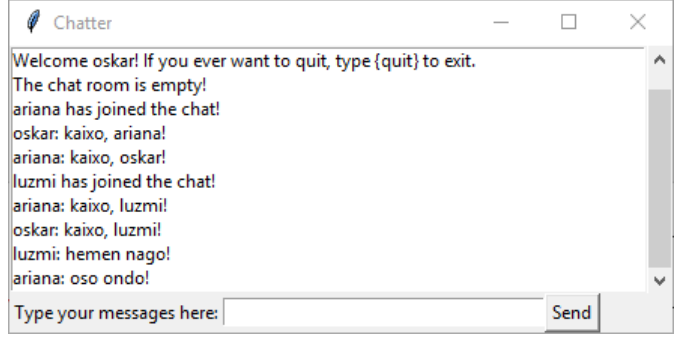
- Bi hariz osoturik dago:
  - GUI (datu bidalketa eta bistaratzea)
  - Datu jasotzea (GUI-an jartzen dira)

- GUI-aren zatiak:
  - leihoa
  - Frame
    - Listbox
    - Scrollbar
  - Label
  - Entry
  - Button



Entry atalean idazten dena, StringVar motako aldagai batean gordetzen da

Bezeroaren egitura



- Zerbitzaritik datu bat jasotzen denean, mezu zerrendaren amaieran sartzen da. Listbox-ean mezu gehiago sartzen ez badira, scroll-ak behera egiten du.

```
# mezua jaso arte itxaron
msg = client_socket.recv(buffer_size)

# mezua Listbox-aren amaieran sartu
msg_listbox.insert(tkinter.END, msg)

# mezua sartzen ez bada, scroll-a jeitsi
msg_listbox.yview(tkinter.END)
```

- Erabiltzaileak Entry atalean idatzi eta Return tekla edo Send botoia sakatzen baditu, send() metodoa exekutatzen da, mezua zerbitzarira bidaliz:

```
entry_field.bind("<Return>", send)
send_button = tkinter.Button(leihoa, text="Send", command=send)


def send():
    # StringVar aldagaitik mezua atera
    msg = my_msg.get()

    # mezua bidali
    client_socket.send(msg)
```

Zerbitzariaren egitura

- Hainbat hariz osoturik:
  - Hari nagusia:
    - Bezeroen TCP konexio eskaerak onartzen ditu.
    - Bezero bakoitzari zerbitzu emateko hari bana sortzen du.

```
# bezeroaren eskaera jaso eta TCP konexioa ezarri (SYN, SYN+ACK eta ACK)
client, client_address = server_socket.accept()

# bezeroarentzako haria sortu (harian "handle_client" metodoa exekutatuko da)
client_thread = Thread(target=handle_client, args=(client, ))

# haria abiarazi
client_thread.start()
```

    - Konektatutako bezeroen erabiltzaile izenak eta IP addr, TCP port bikoteak gordetzen ditu.

```
usernames = {client1: 'oskar', client2: 'ariana', client3: 'luzmi'}

addresses = {client1: (23.15.6.234, 6234), client2: (232.75.6.23, 26234), client3: (55.1.88.2, 45340)}
```

  - Bezeroen hariak: hari bakoitzak bezero batetik mezuak jaso eta mezu horien broadcast-a egiten du