Balanceador de Carga para o PySnakes

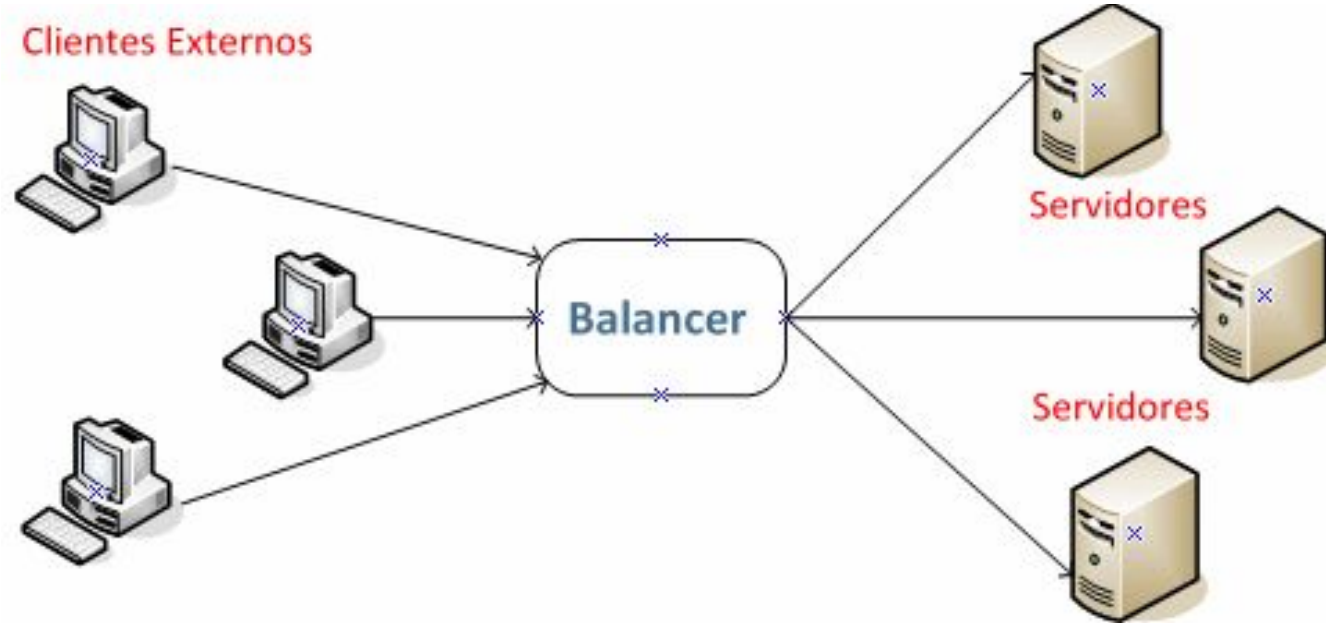
Alunos:

Daniel Guerra
Gabriel Igor
Paulo Augusto

1. Balanceamento de carga

- Solução para picos de tráfego de acesso;
- Técnica para distribuir a carga de trabalho uniformemente entre dois ou mais computadores, enlaces de rede, UCPs, discos rígidos ou outros recursos;
- Objetivo:
  - Otimizar a utilização de recursos;
  - Maximizar o desempenho;
  - Minimizar o tempo de resposta;
  - Evitar sobrecarga.

2. Balanceador de carga - Exemplo



3. Proposta

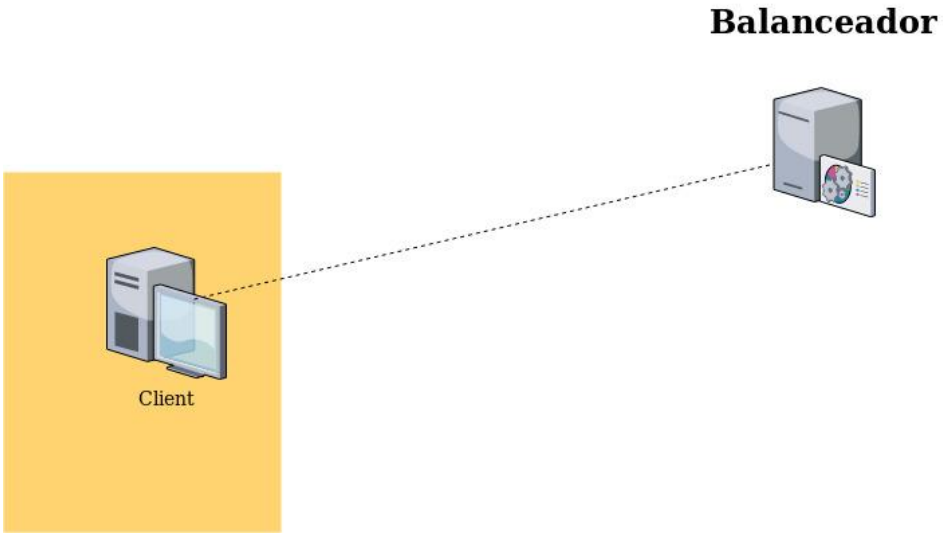
- Elaborar dois exemplos de balanceadores de carga para o jogo PySnakes elaborado na segunda unidade da disciplina:
  - Um balanceador de carga responsável por identificar quando o servidor está lotado, criando um novo servidor para alocar novos clientes;
  - Implementação de algoritmo de balanceamento de carga segundo um modelo da literatura.

3. Proposta

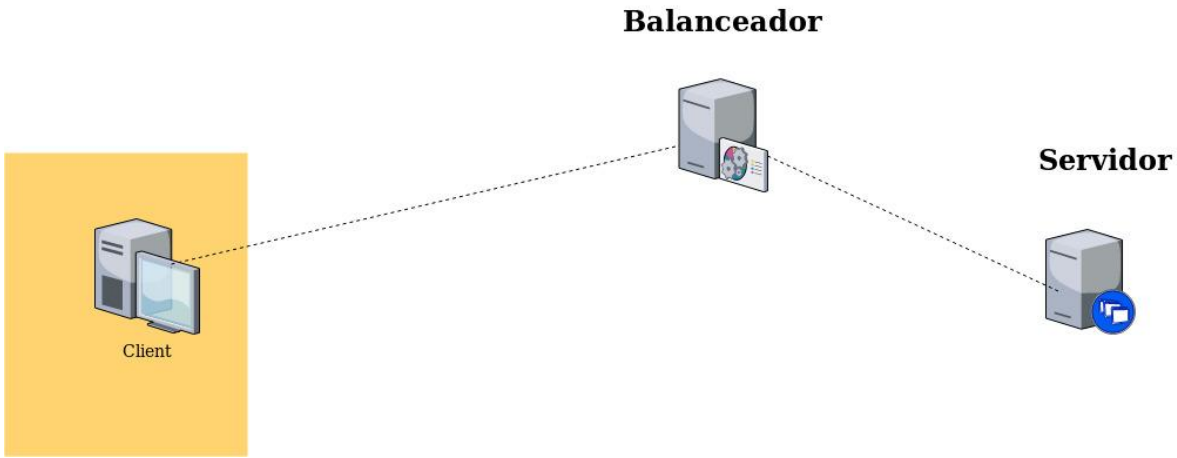
```python
def main():
    global balancer

    param = sys.argv[1:]
    port = int(param[0])

    # balancer = RoundRobin(port, 2)
    balancer = Balancer(port)
    # signal.signal(signal.SIGINT, balancer.stop)
    balancer.run()
```

```python
def run(self):
    # tem que ter o socket dos clientes
    # tem q um socket para cada servidor
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as clientsSocket:
        clientsSocket.setblocking(0)
        clientsSocket.bind(('', self.port))
        clientsSocket.listen(5)
        self.clientsReadList.append(clientsSocket)

        if debug:
            print('>>> Socket para clientes inicializado. Porta: {}'.format(self.port))

        while self.flag:
            # if debug:
                # print('>>> Processando conexoes de clientes')
            self.processClientsConns(clientsSocket)

    for thread in createdThreads:
        thread.join()
```

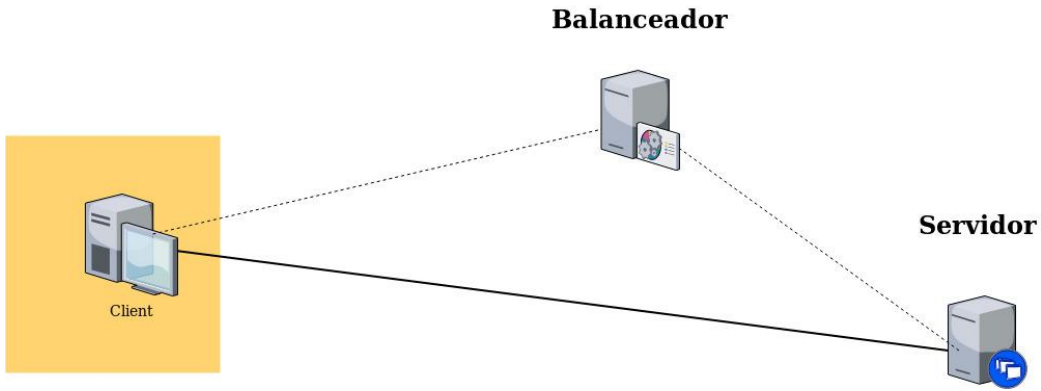
4. Primeira Solução - Balanceador iniciando servidores



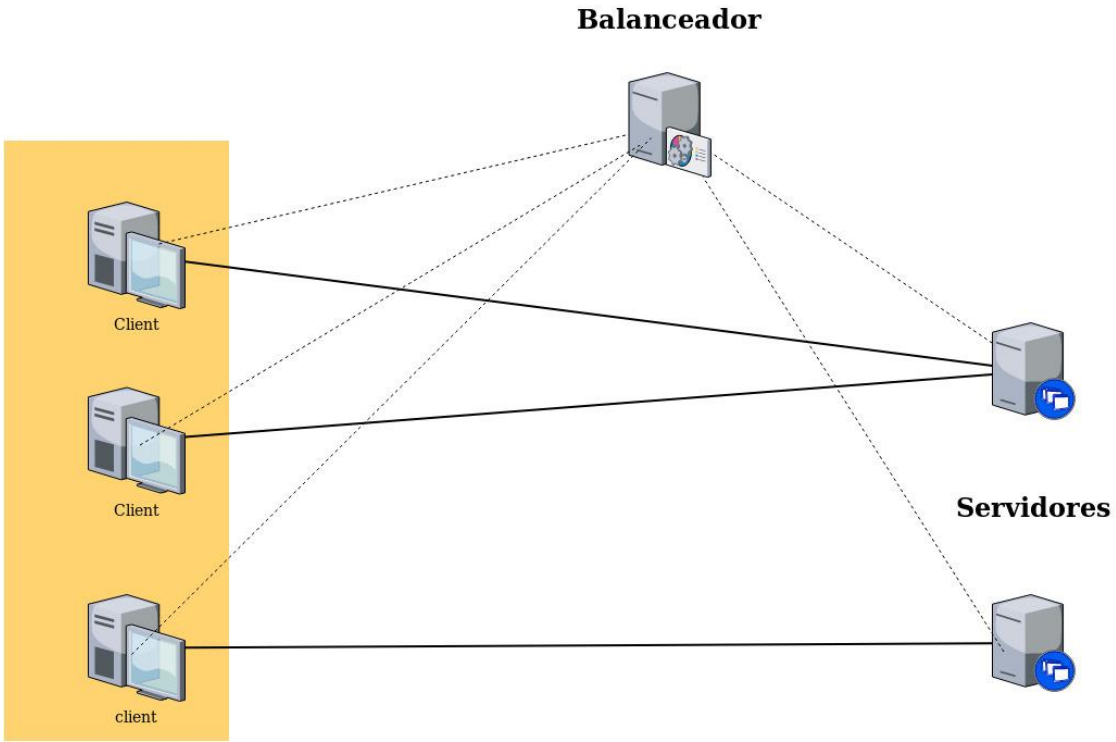
4. Primeira Solução - Balanceador iniciando servidores



4. Primeira Solução - Balanceador iniciando servidores



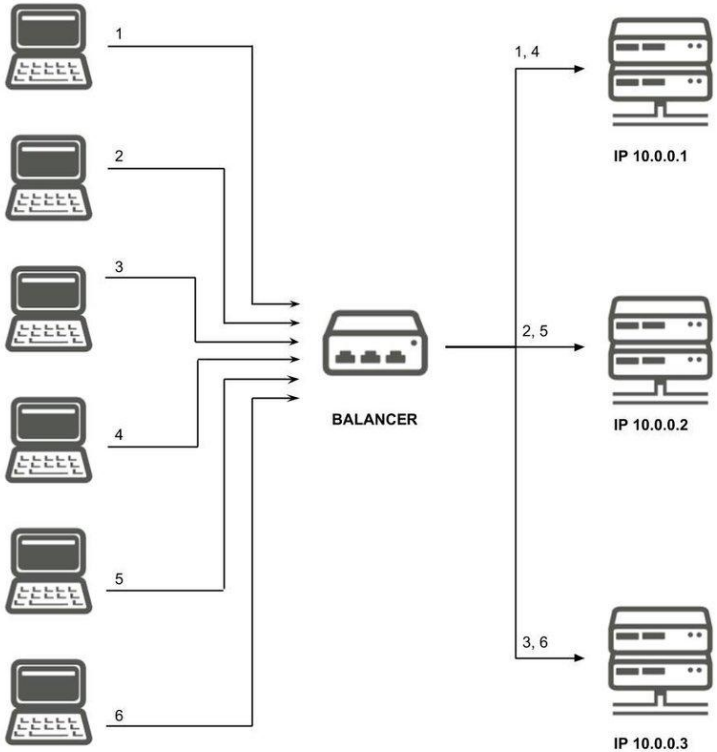
4. Primeira Solução - Balanceador iniciando servidores



4. Primeira Solução - Balanceador iniciando servidores

```
def findServer(self):
    if debug:
        print(">>> Tentando achar server")
    foundServer = -1

    for i in range(len(self.servers)):
        if not self.isServerFull(i):
            foundServer = self.servers[i]

    if(foundServer == -1):
        foundServer = self.initServer()
    return foundServer
```

5. Segunda Solução - Round Robin

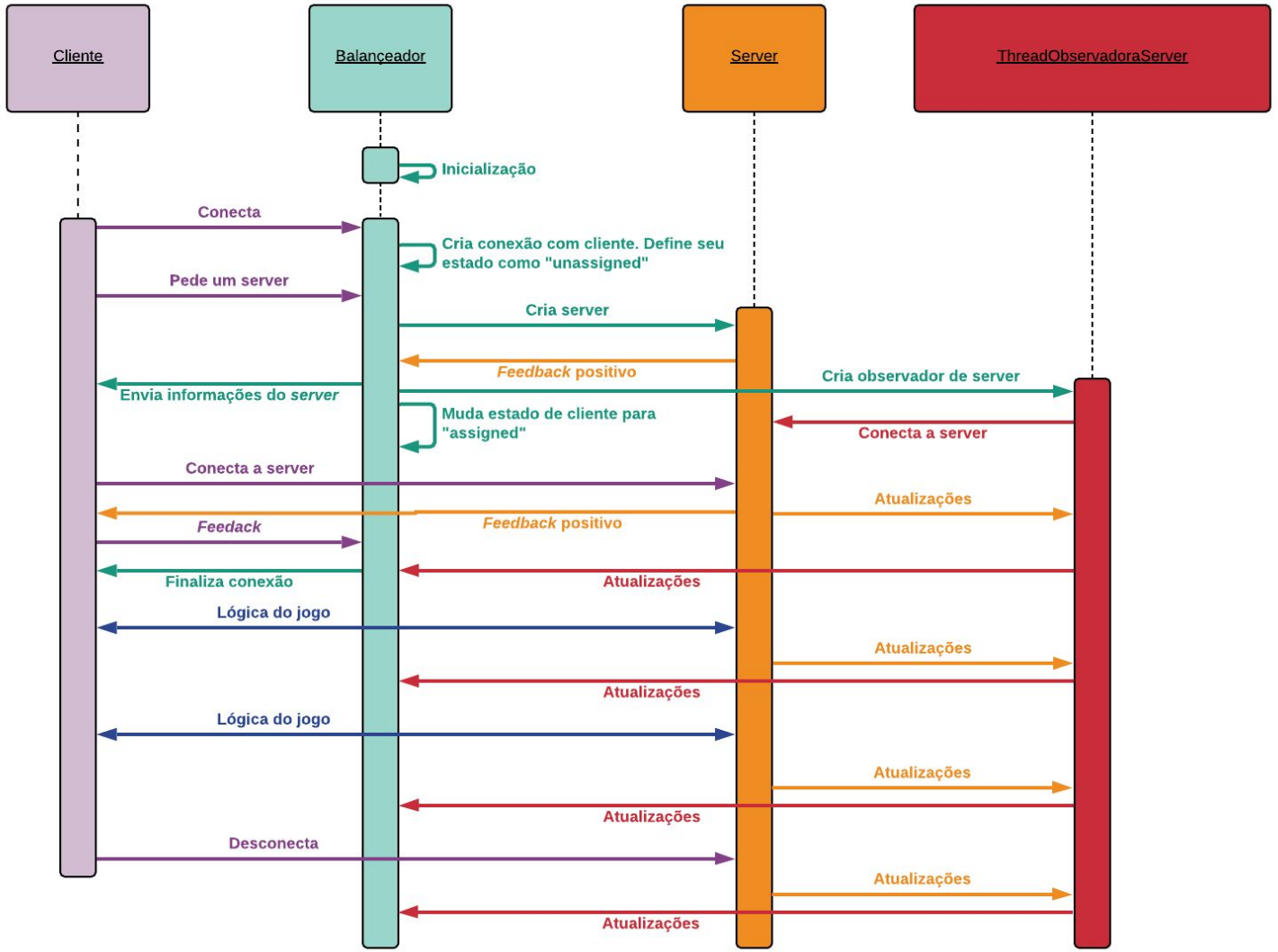


5. Segunda Solução - Round Robin

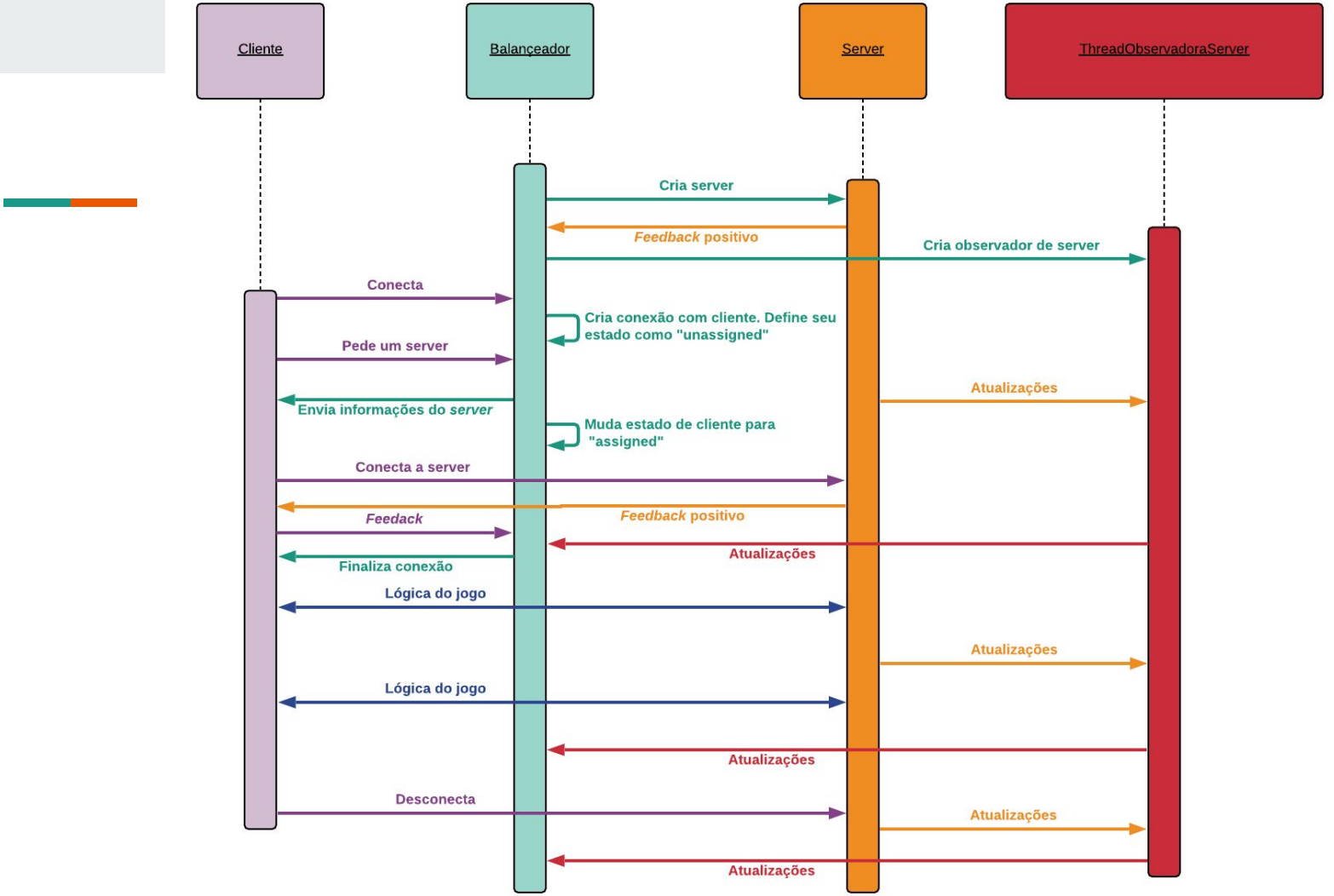
```python
class RoundRobin(Balancer):
    def __init__(self, port, numInitServers):
        super().__init__(port)

        self.chosenServer = 0

        self.initServers(numInitServers)

    def initServers(self, numInitServers):
        for i in range(numInitServers):
            self.initServer()
```

```python
def findServer(self):
    if debug:
        print(">>> Tentando achar server")
    foundServer = -1

    while True:
        if not self.isServerFull(self.chosenServer):
            foundServer = self.servers[self.chosenServer]
            self.chosenServer = (1 + self.chosenServer) % len(self.servers)
            break
        else:
            self.chosenServer = (1 + self.chosenServer) % len(self.servers)

    return foundServer
```

6. Diagrama de sequência - Primeira Solução


Cliente
Balançeador
Server
ThreadObservadoraServer
Inicialização
Conecta
Cria conexão com cliente. Define seu estado como "unassigned"
Pede um server
Cria server
Feedback positivo
Cria observador de server
Envia informações do server
Muda estado de cliente para "assigned"
Conecta a server
Conecta a server
Atualizações
Feedback positivo
Feedack
Finaliza conexão
Atualizações
Lógica do jogo
Atualizações
Atualizações
Lógica do jogo
Atualizações
Atualizações
Desconecta
Atualizações
Atualizações

6. Diagrama de sequência - Segunda Solução


Cliente
Balançeador
Server
ThreadObservadoraServer
Cria server
Feedback positivo
Cria observador de server
Conecta
Cria conexão com cliente. Define seu estado como "unassigned"
Pede um server
Atualizações
Envia informações do server
Muda estado de cliente para "assigned"
Conecta a server
Feedback
Feedback positivo
Atualizações
Finaliza conexão
Lógica do jogo
Atualizações
Lógica do jogo
Atualizações
Desconecta
Atualizações
Atualizações

Obrigado!