Sistemas Distribuidos

TAREA #1

FRANCISCO ABARCA

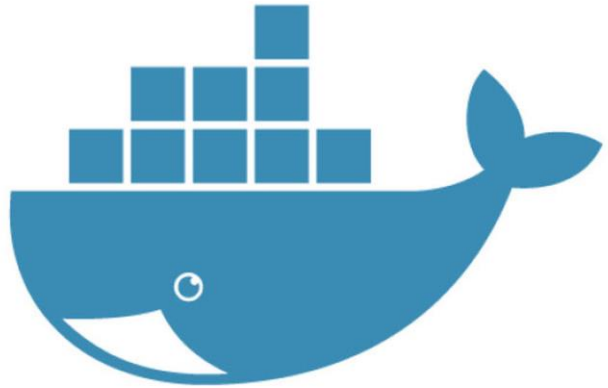
ALEXANDER RUZ

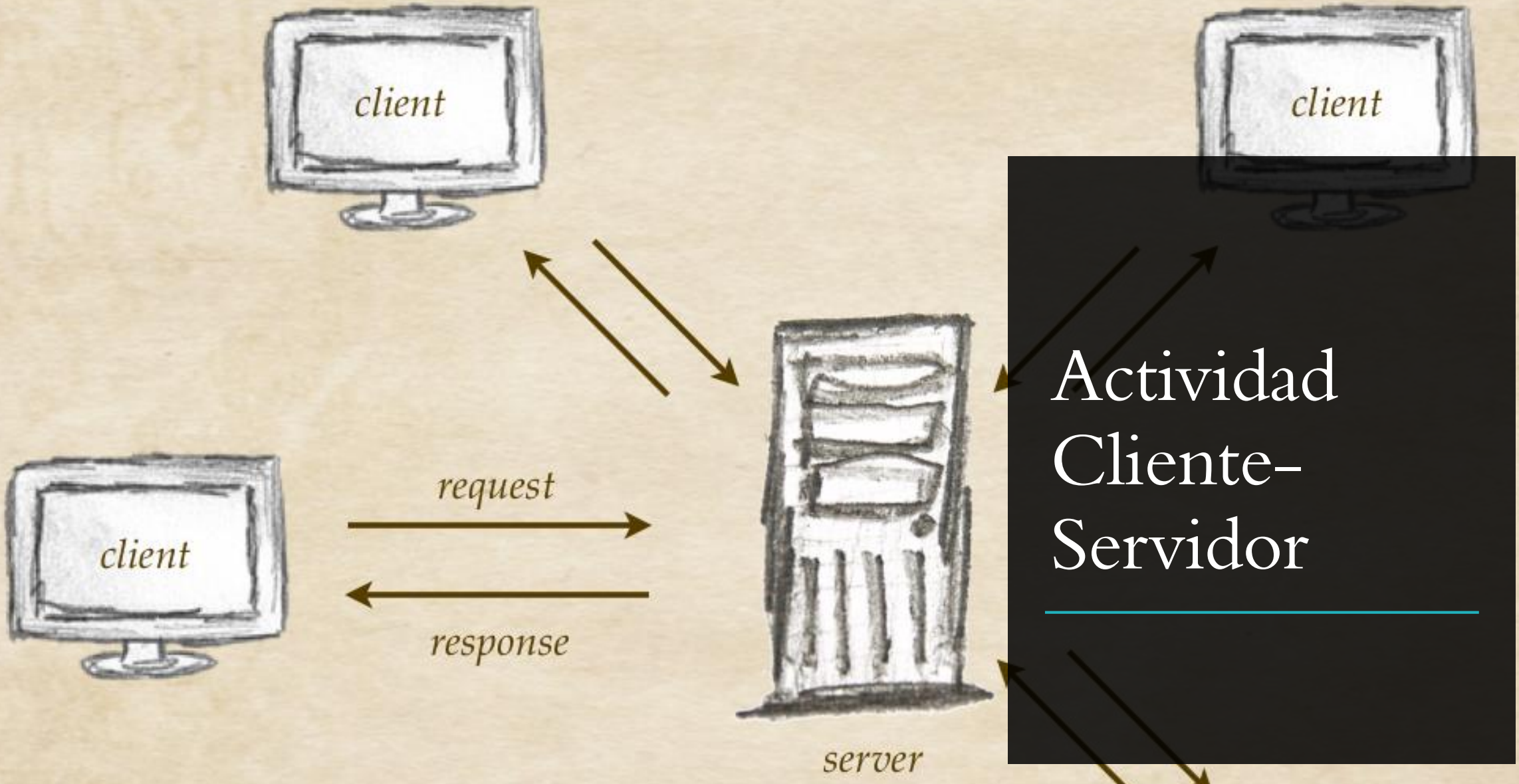
Temario

- Arquitectura
- Implementación
  - Diseño conceptual
  - Comportamiento de sockets
    - Envío de datos
    - Heartbeat
  - Dockerfiles
  - Docker-compose

Arquitectura

LAS TECNOLOGÍAS BASES SON PYTHON (LÓGICA) Y DOCKER (DESPLIEGUE)





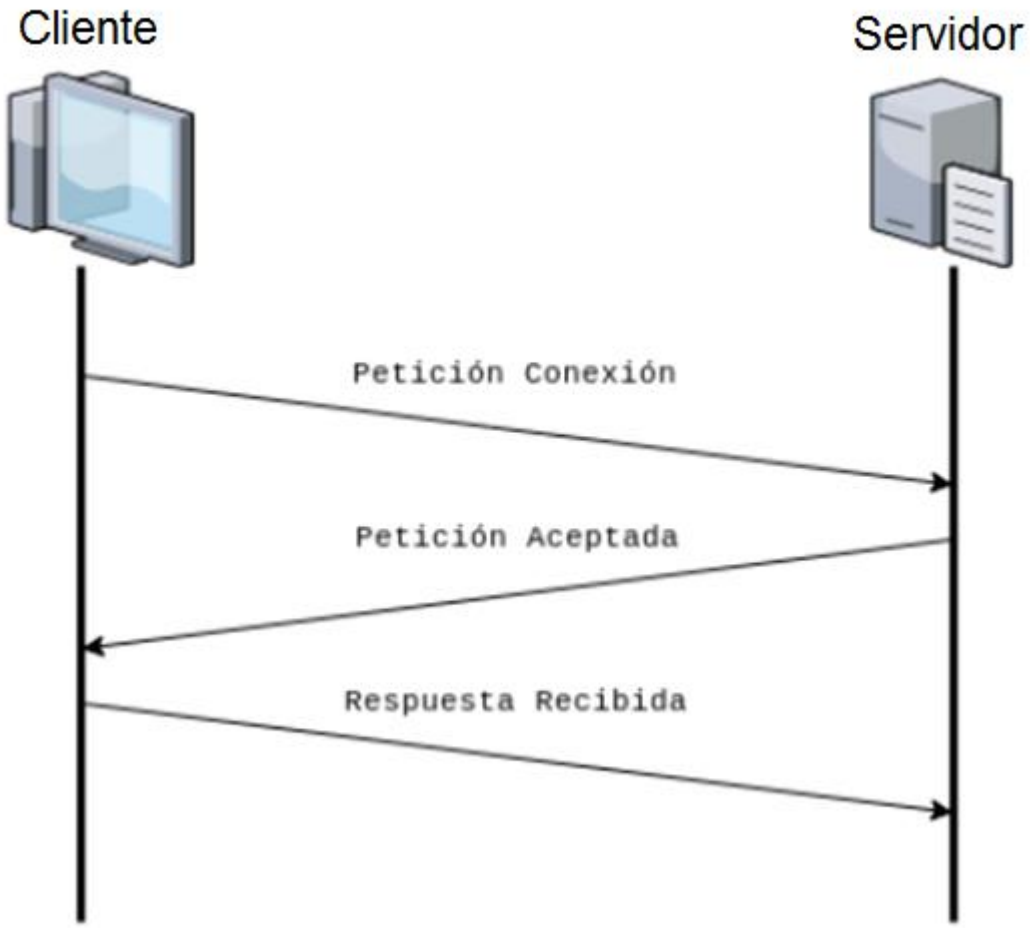
Actividad Cliente-Servidor

Cliente-Servidor

Se realiza una conexión en el puerto 5000.

Hay comunicación entre ambos.

Ambos tienen un archivo con información.



Dockerfiles

El ejecutable de cliente y servidor se empaquetan en un contenedor diferente para cada uno.

Cliente

```
1  FROM python:latest
2
3  ADD cliente.py /cliente/
4
5  WORKDIR /cliente/
```

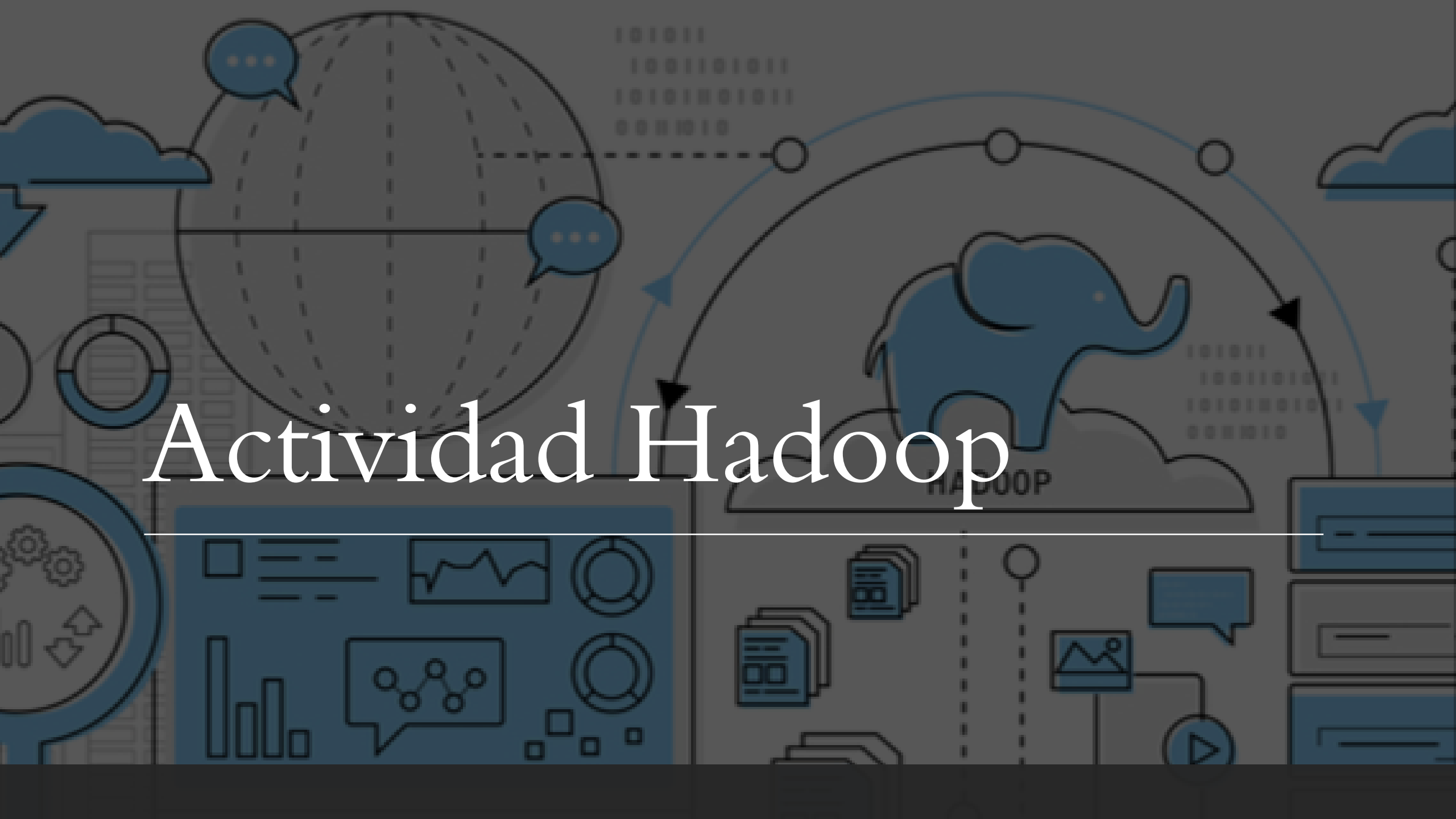
Servidor

```
1  FROM python:latest
2
3  ADD servidor.py /servidor/
4
5  WORKDIR /servidor/
```

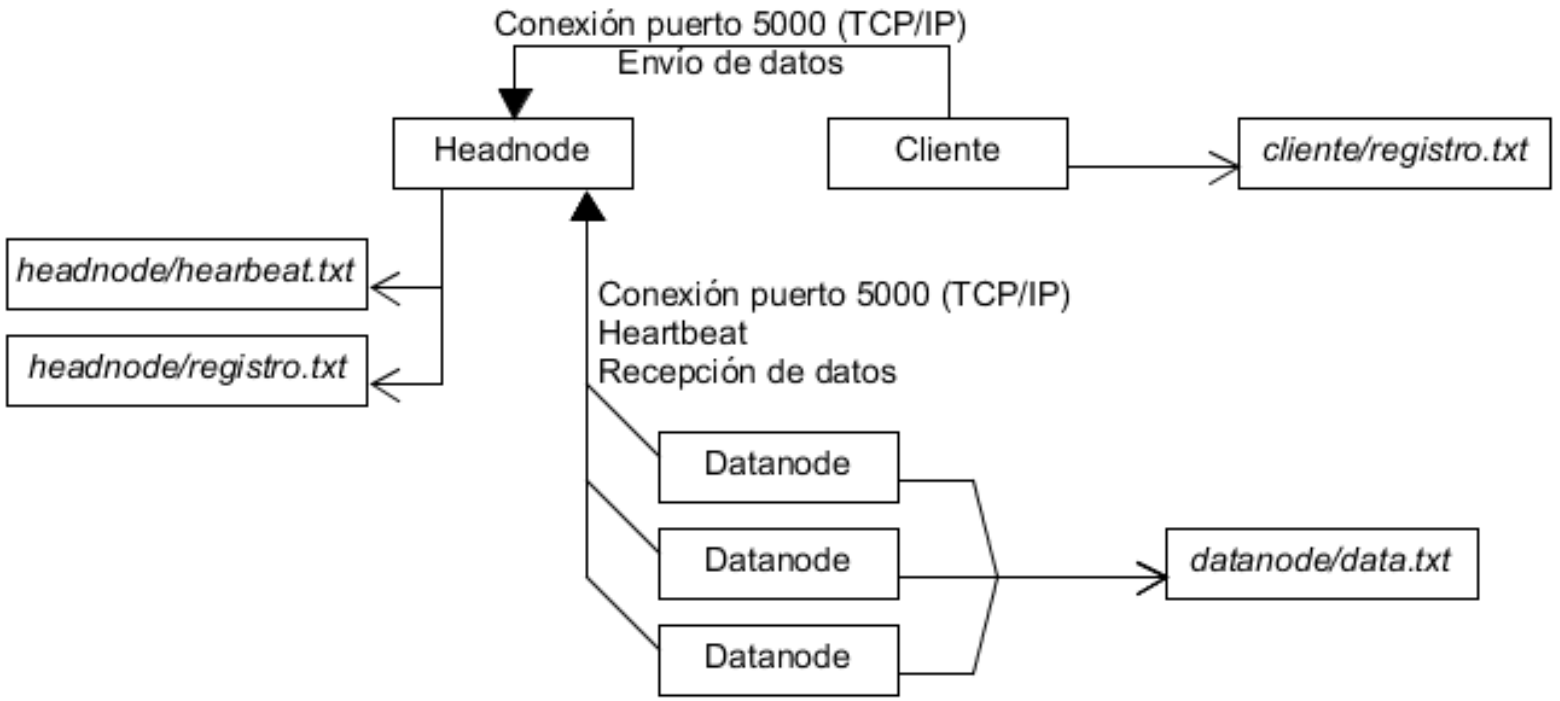
Docker-compose

Docker-compose busca instrucciones en este archivo, que luego ejecuta según lo que se busca.

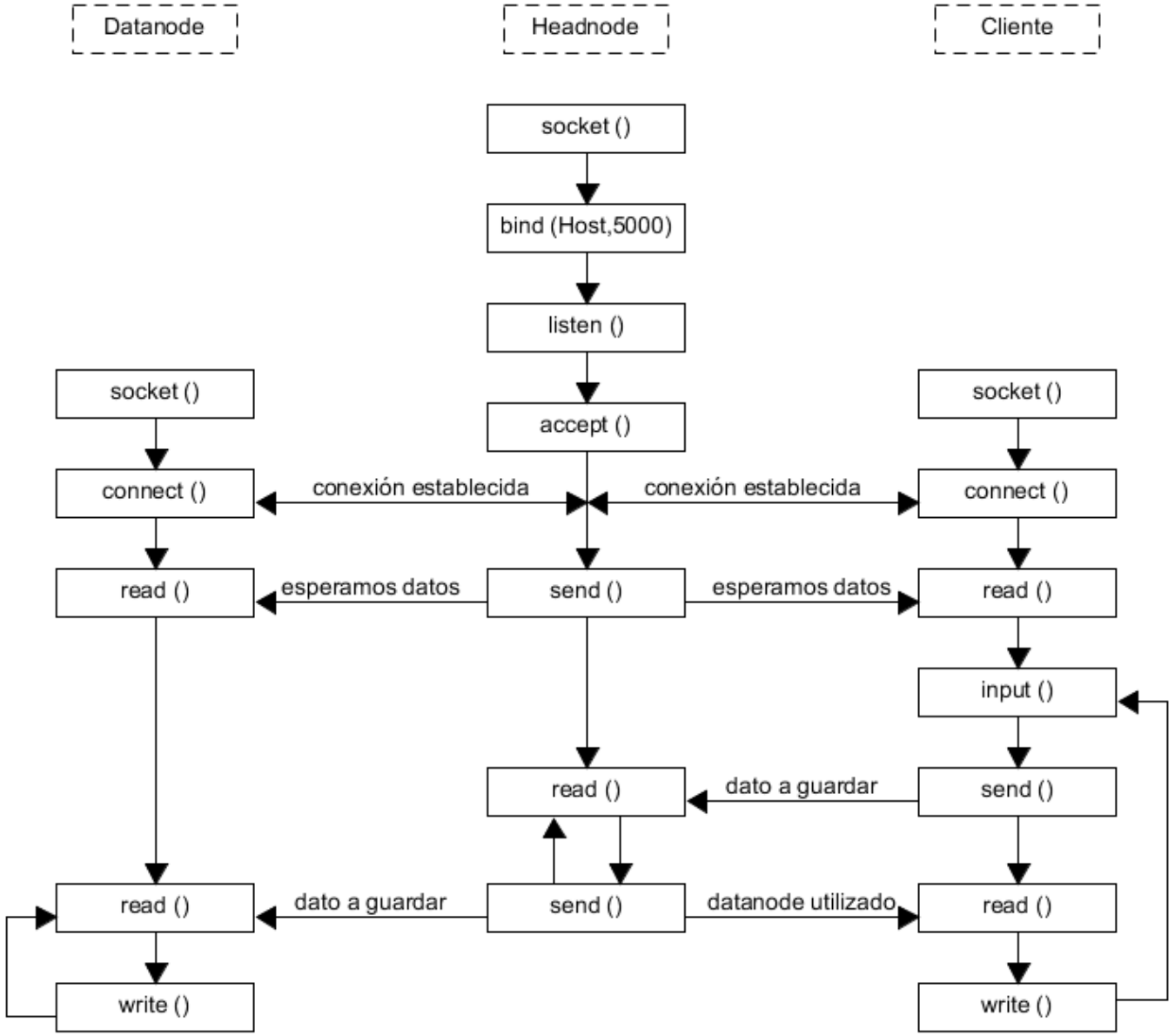
```yaml
version: "3"

services:
  server:
    build: servidor/
    command: python3 servidor.py
    ports:
      - "5000:5000"
    volumes:
      - ./servidor:/servidor/

  client:
    build: cliente/
    command: python3 cliente.py
    volumes:
      - ./cliente:/cliente/
    depends_on:
      - server
```



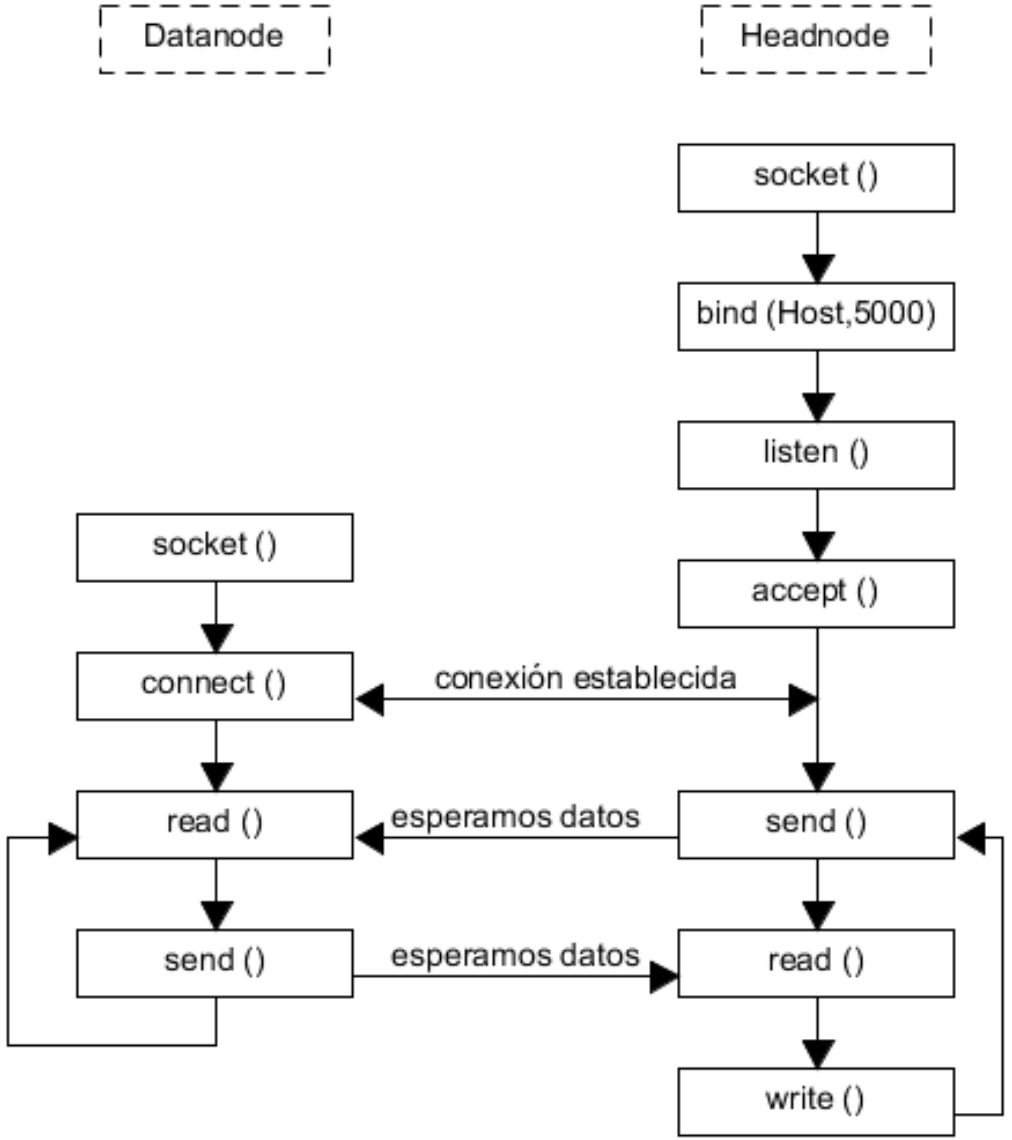
Actividad Hadoop



Diseño conceptual



Comportamiento de sockets (Envío de datos)



Comportamiento de sockets (Heartbeat)

Dockerfiles

Headnode

FROM python:latest

ADD Headnode.py /headnode/

WORKDIR /headnode/

EXPOSE 5000

Datanode

FROM python:latest

ADD Datanode.py /datanode/

WORKDIR /datanode/

Cliente

FROM python:latest

ADD Cliente.py /cliente/

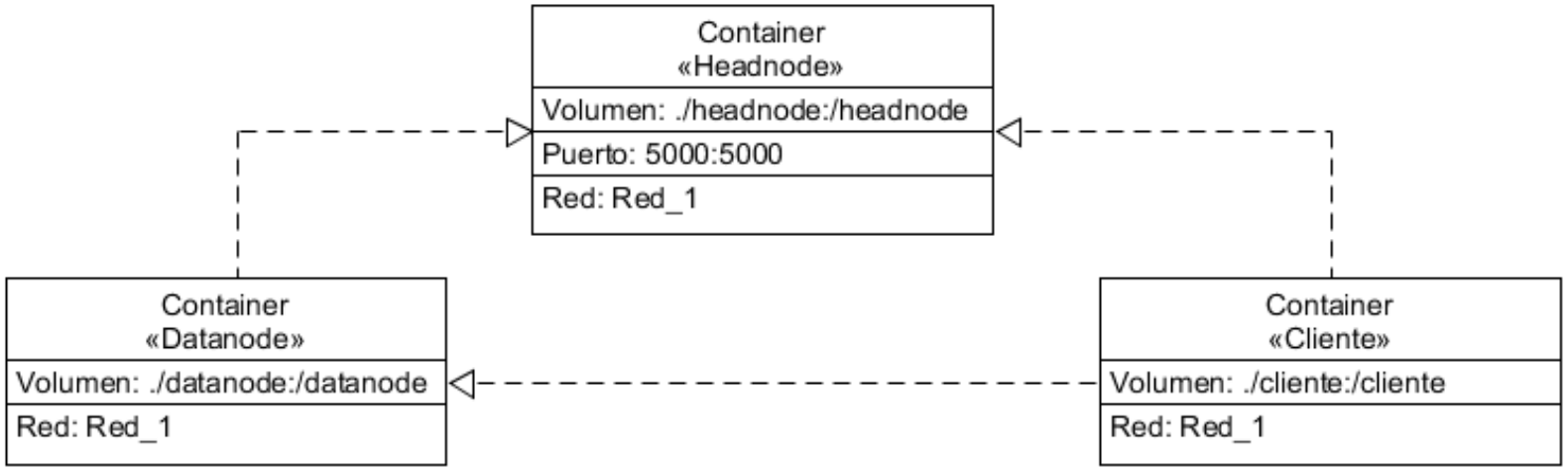
WORKDIR /cliente/

Docker-compose

```
cliente:
  container_name: Cliente
  stdin_open: true
  tty: true
  build: cliente/
  command: python ./Dummy.py
  network_mode: host
  volumes:
    - ./cliente:/cliente
  depends_on:
    - headnode
    - datanode
```

```
datanode:
  build: datanode/
  command: python ./Datanode.py
  network_mode: host
  volumes:
    - ./datanode:/datanode
  depends_on:
    - headnode
```

```
headnode:
  container_name: Headnode
  stdin_open: true
  tty: true
  build: headnode/
  command: python ./Headnode.py
  ports:
    - "5000:5000"
  network_mode: host
  volumes:
    - ./headnode:/headnode
```



Modelo de containers