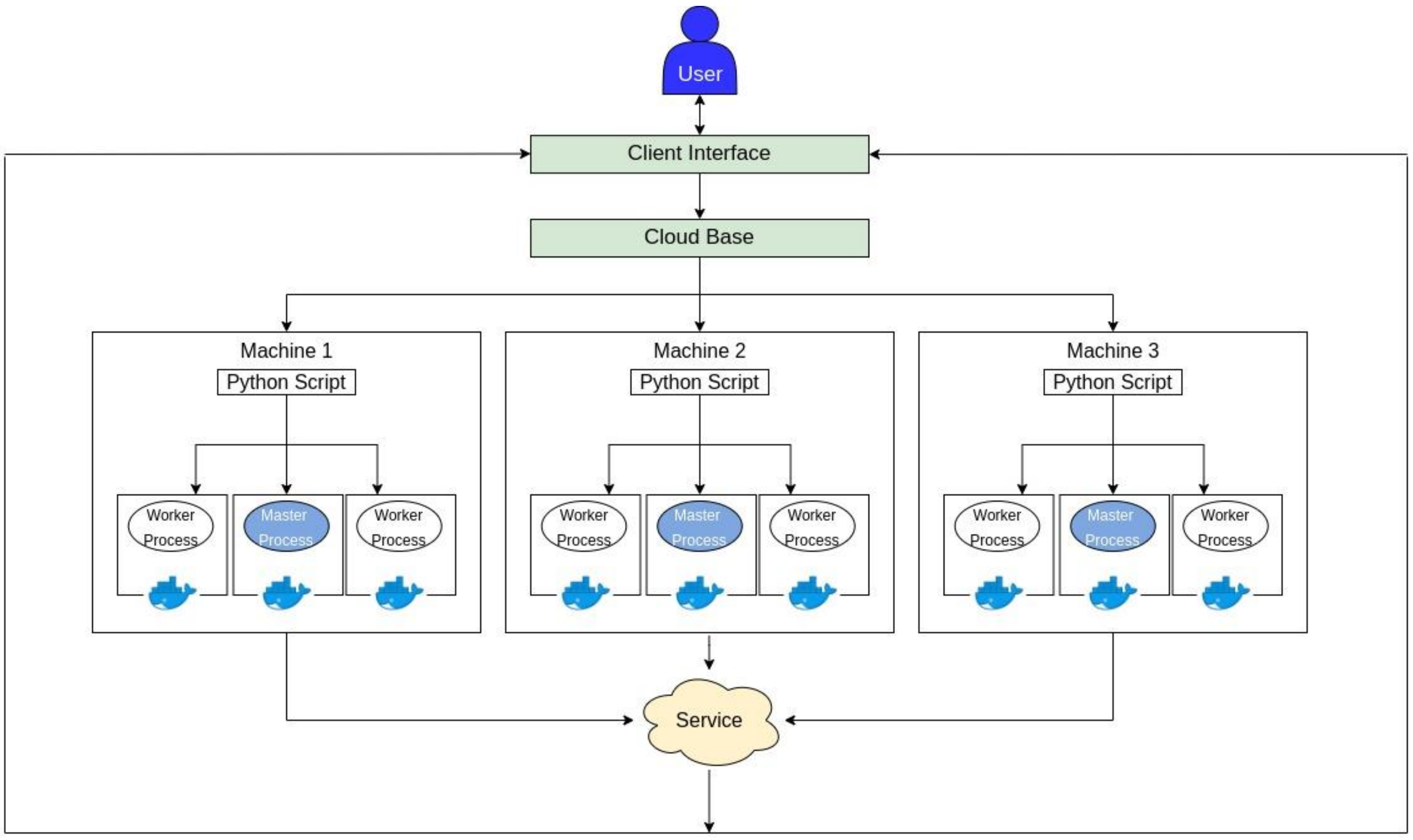
Cloud Service Design Doc

Abhishek Prasad	2018A7PS0331P
Aditya Anand Bangalore	2018A7PS0265P
Chirag CD	2018A7PS0277P
Rishiraj Acharyya	2018A7PS0168P
Yash Jain	2018A3PS0333P

Birla Institute of Technology and Science, Pilani

CLOUD COMPUTING
Group Number 21
28th November 2021

Architecture



Workflow

Service Start

- The Client requests the cloud base to start a new service.
- The Cloud Base uses an internal Distributor module to find machines with the fewest number of containers running and selects those to run a new service server and 2 worker processes.
- The Main server sends regular heartbeats to the cloud base regarding the health of the underlying containers (infra monitoring).
- The worker processes send their heartbeat to the main server (platform monitoring).

- The main server collates the entire data of all the workers and sends it to the client showing the number of healthy machines being used for that service (service monitoring), which also doubles as a means to let the client know which main server he should contact to answer his queries.

Client Query

Initially, the python scripts are running in their respective machines.

- The user sends a request for a service to the main server that is registered to that particular service.
- The main server selects a free worker to run the query and forwards the query to the selected worker.
- The worker computes the answer to the query and sends it back to the server.
- The main server forwards the result back to the client.

Auto-scaling

- The scaler is located on the cloud base and is called whenever the machine_base sends an update regarding the health and status of the containers on it.
- The scaler decides whether or not to allocate more containers to the service depending on whether the fraction of running containers upon total containers crosses 80%.
- The scaler then tells the distributer to allocate a certain number of containers for the particular overburdened service.
- The distributor checks the number of containers on each machine and picks the one with the least load to add a container.
- De-scaling happens in the same manner.

Usage

```
$ python cloud_base.py
```

```
$ python machine_base.py <machine_id> <machine_ip> <cloud_base_address>
```

```
$ python random_gen_server.py <server_id> <service_id> <server_ip>
<cloud_base_address> <client_address>
```

```
$ python random_gen_worker.py <worker_id> <service_id> <worker_ip>
<main_server_address>
```

```
$ python random_gen_client.py
```

```
$ docker build -t <process.py> .
```