
Standard P1935

Introduction & Implementation

陳奐廷

Outline

- ❖ Testbed architecture
- ❖ Testbed actions
- ❖ Testbed apis
- ❖ Testbed implementation
- ❖ Testbed usage

Architecture



Edge / Fog Orchestrator

An orchestrator of the whole Edge / Fog system



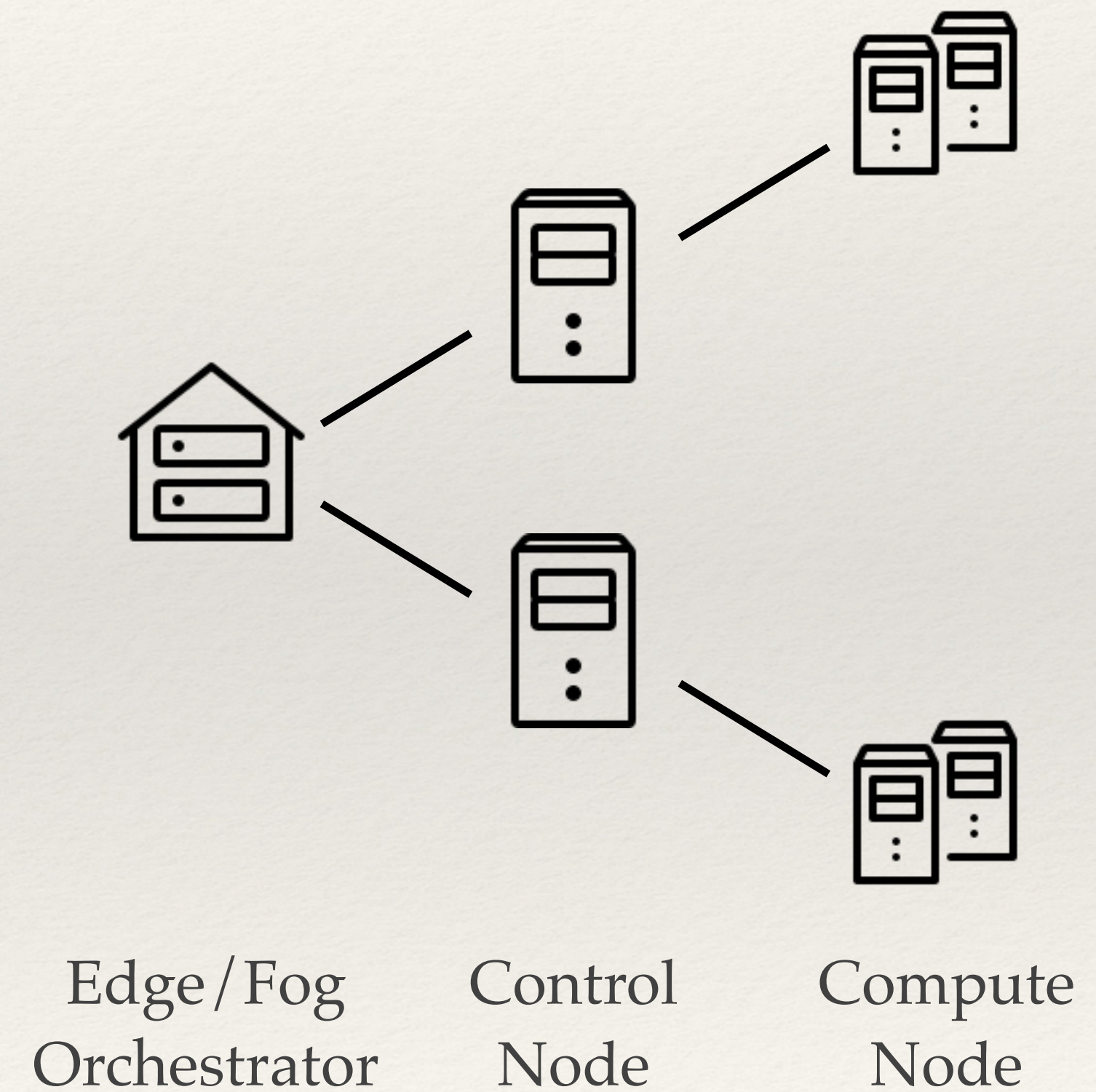
Control Node

Manage the related compute nodes



Compute Node

An entity that is able to handle the computing tasks on its own

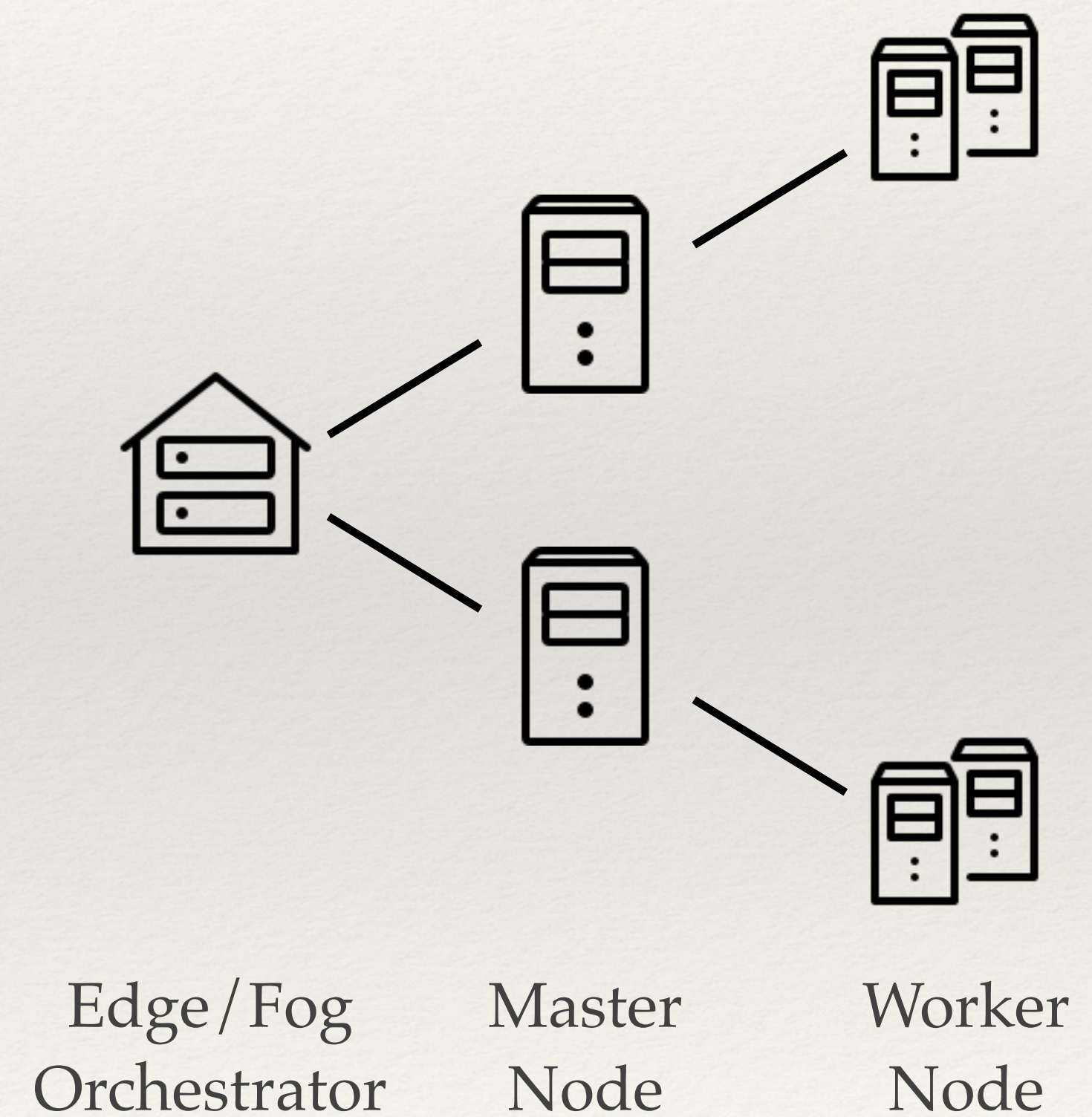


Architecture

 Edge / Fog Orchestrator
 python™  mongoDB®

 Control Node
 **kubernetes**

 Compute Node
 **kubernetes**

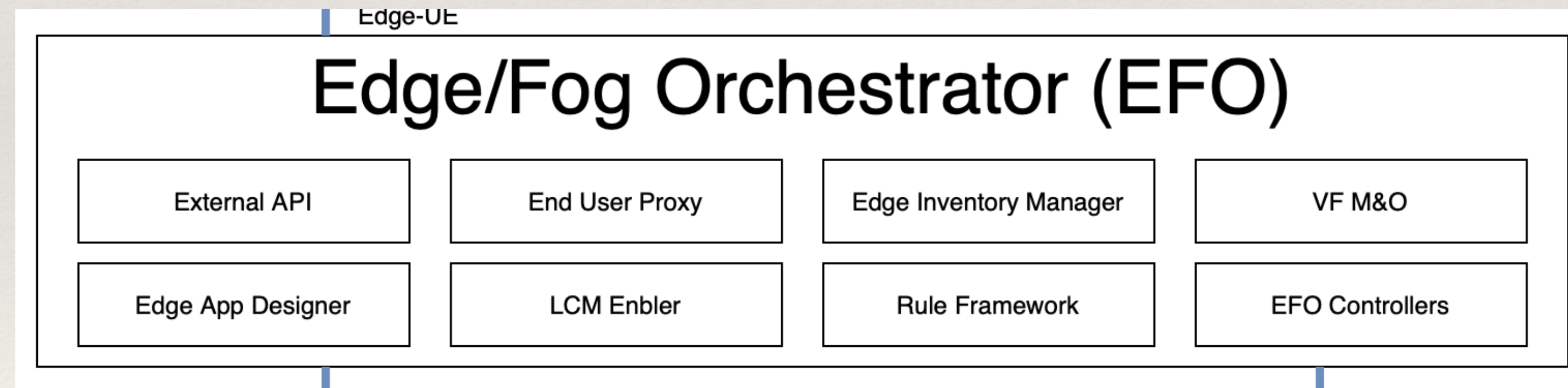


Architecture



Edge / Fog Orchestrator

An orchestrator of the whole Edge / Fog system

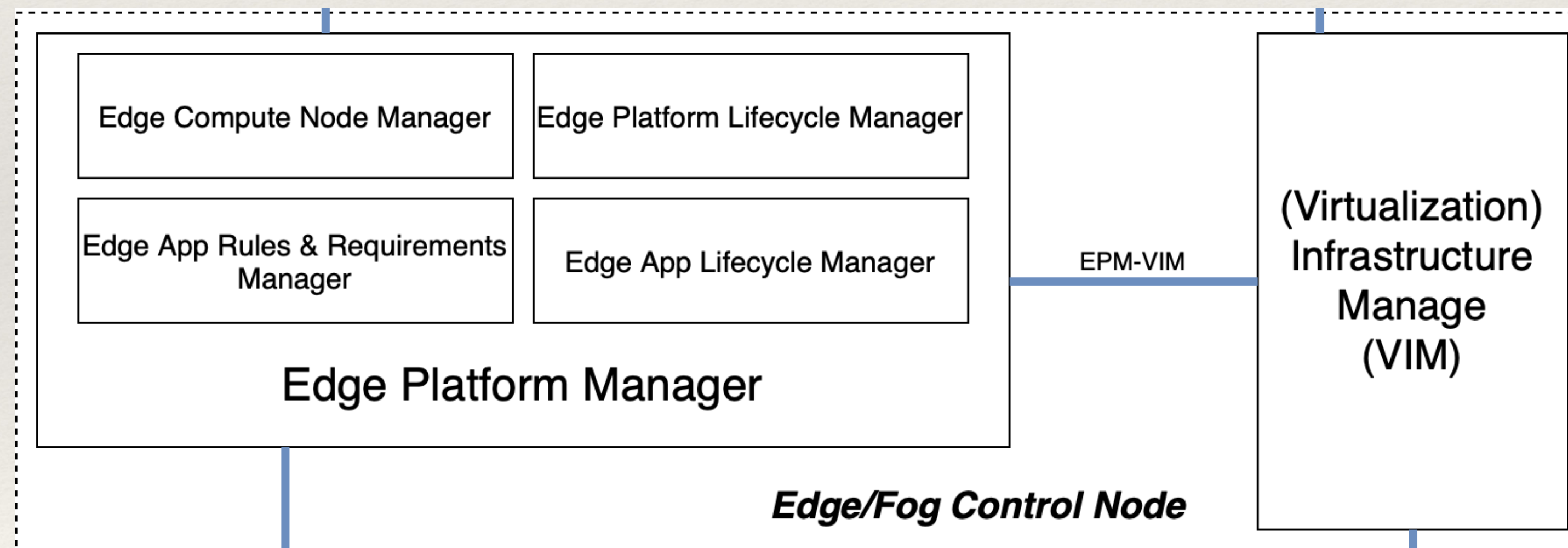


Architecture



Control Node

Manage the related compute nodes

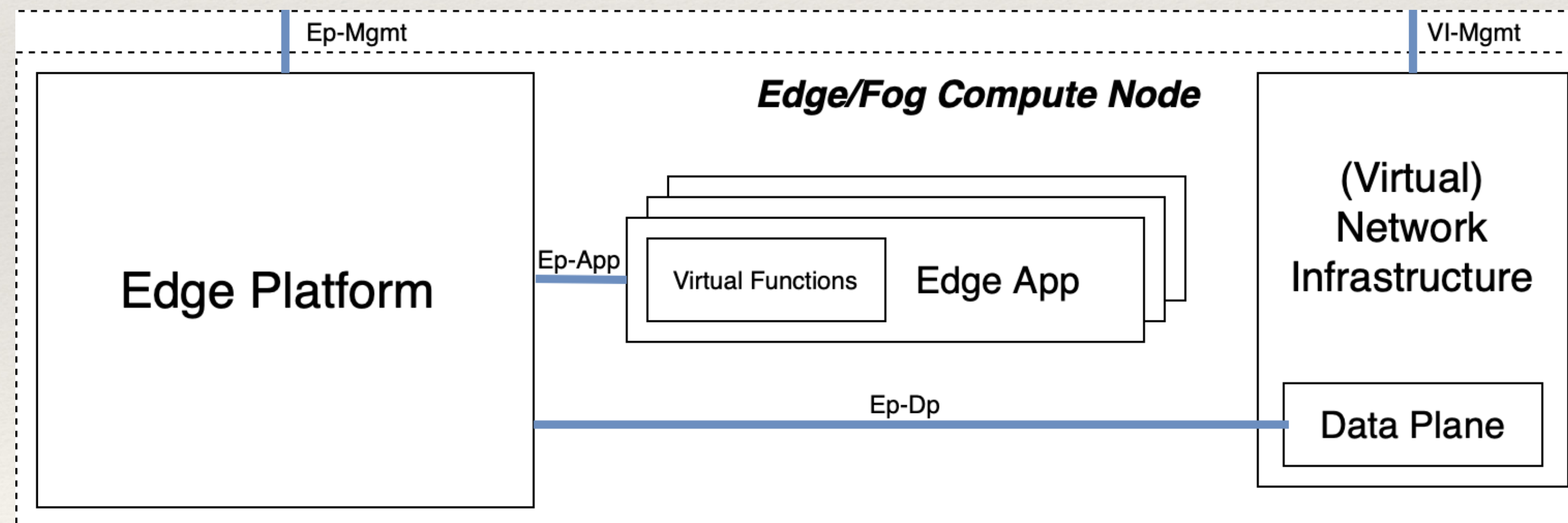


Architecture



Compute Node

An entity that is able to handle the computing tasks on its own



Users



Infrastructure Owner



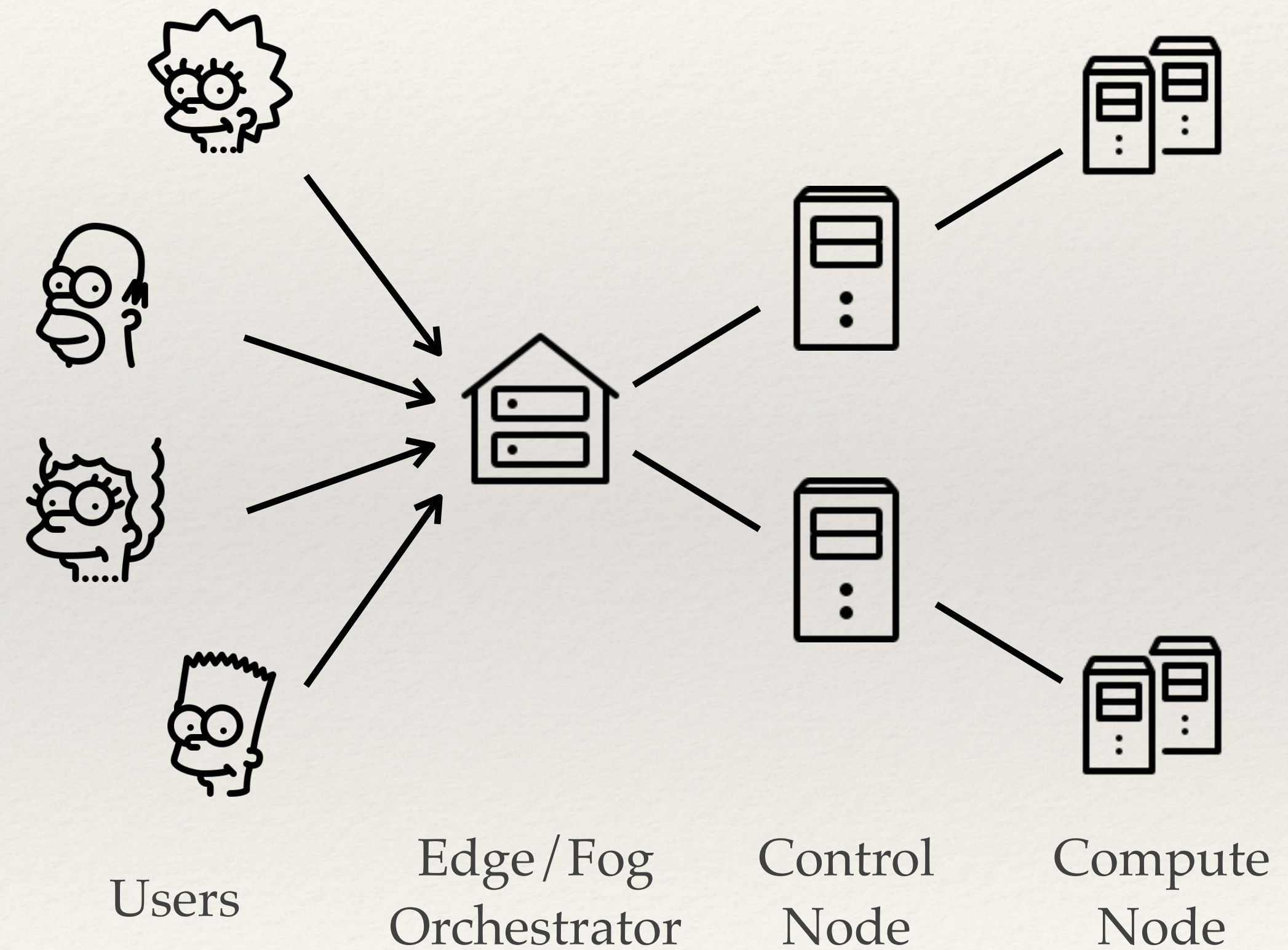
Edge Service Provider



Edge Service Operator



End User

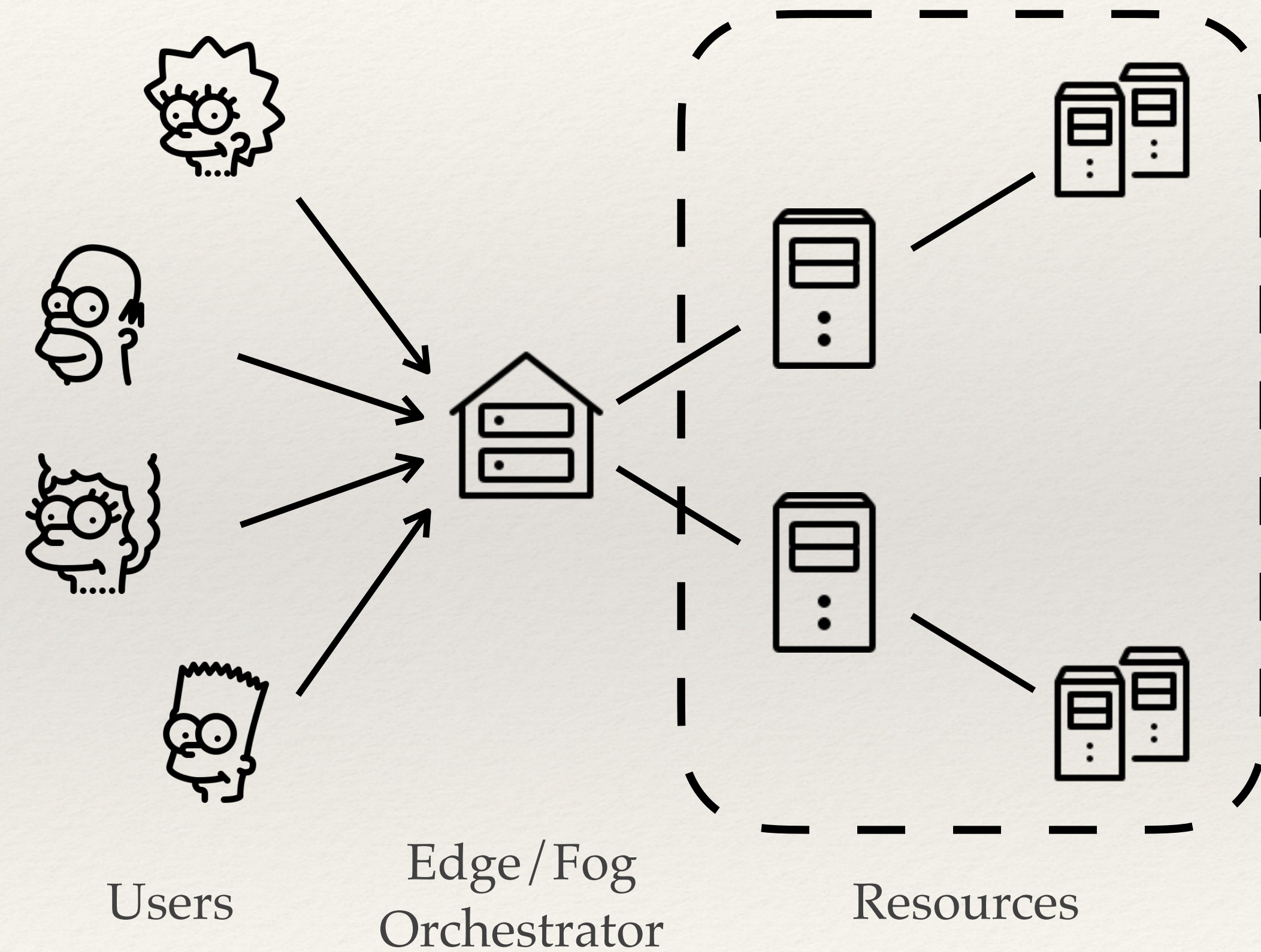


Actions

- ❖ Resource management
- ❖ Application management

Actions

- ❖ Resources
 - ❖ Physical machine
 - ❖ Virtual machine



Actions

- ❖ Applications
 - ❖ Run in container (Kubernetes)
 - ❖ Configured by manifest

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: helloworld
  labels:
    app: helloworld
spec:
  replicas: 1
  selector:
    matchLabels:
      app: helloworld
  template:
    metadata:
      labels:
        app: helloworld
    spec:
      containers:
      - name: helloworld
        image: dingyiyi0226/demoapp
        ports:
        - containerPort: 5678
```

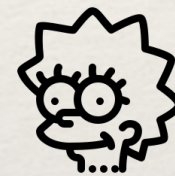
Actions

❖ Resource Management

❖ Creation



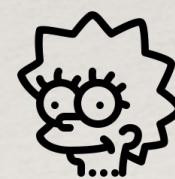
❖ Query / Discovery



❖ Reconfiguration



❖ Deletion



❖ Application Management

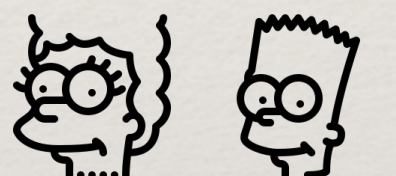
❖ Onboarding



❖ Instantiation



❖ Context Creation / Deletion



❖ Reconfiguration



❖ Termination



Infrastructure Owner



Edge Service Provider



Edge Service Operator



End User

REST APIs

❖ Resource Management

list_clusters	GET /api/clusters
create_cluster	POST /api/clusters
delete_cluster	DELETE /api/clusters/<clusterName>
list_nodes	GET /api/clusters/<clusterName>/nodes
create_node	POST /api/clusters/<clusterName>/nodes
list_machines	GET /api/machines
create_machine	POST /api/machines
delete_machine	DELETE /api/machines/<machineName>
create_vm	POST /api/vms

❖ Application Management

list_manifests	GET /api/manifests
create_manifest	POST /api/manifests
delete_manifest	DELETE /api/manifests/<manifestID>
patch_manifest	PATCH /api/manifests/<manifestID>
list_apps	GET /api/apps
create_app	POST /api/apps
delete_app	DELETE /api/apps/<appID>
post_app_contexts	POST /api/apps/<appID>/contexts
list_services	GET /api/services

See API.md for more information

Application Lifecycle

1. Service provider uploads the manifest

POST /api/manifests

2. Service operator authorizes and distributes the manifest

POST /api/apps

3. Service operator manages the application

GET /api/apps/<appID>/instantiate

GET /api/apps/<appID>/terminate

DELETE /api/apps/<appID>

See Actions.md, API.md, and demo videos (on NAS) for more information

User Interface

Smart Edge operator Log Out

Query Manifest

ID	App Name	Provider ID	Status	Action
provider-facial	facial	provider	Authorized	action
provider-helloworld	helloworld	provider	Authorized	Show Manifest
provider-helloworld-2	helloworld-2	provider	Authorized	Create App

Smart Edge operator Log Out

Query App List

ID	Ref. Manifest ID	Auth By	Host IP	Port	Status	Action
provider-facial	provider-facial	operator	N/A	N/A	Terminated	action
provider-helloworld	provider-helloworld	operator	192.168.50.236	30000	Running	Delete
provider-helloworld-2	provider-helloworld-2	operator	N/A	N/A	Terminated	Instantiate
						Terminate

User Interface

The screenshot displays the Smart Edge user interface. At the top left, there is a menu icon and the text "Smart Edge". At the top right, the user is identified as "owner" with a "Log Out" button.

The main content area is titled "Home Page" and is divided into several sections:

- Identity:** A section with a circular icon containing a house and a person, labeled "Identity". Below it, the role "Infrastructure Owner" is listed with three actions:
 - Create/Delete Machine
 - Create/Delete Cluster
 - Join/Delete Node
- Machine List:** A table listing the machines in the system.

Name	IP	Status	Type	Username	Userpwd
cluster_1_master	192.168.50.242	in use	Virtual	ubuntu	ubuntu
cluster_1_worker	192.168.50.236	in use	Virtual	ubuntu	ubuntu
EPC VM	192.168.50.141	in use	EPC		
Fii BS	192.168.50.70	in use	BS		
- CPU:** A donut chart showing 9% usage.
- Machine:** A donut chart showing 4 machines.
- Memory:** A donut chart showing 38% usage.

Testbed Location

- ❖ Server IP

140.112.187.109, 192.168.50.166

- ❖ Path

/home/1935/standard-p1935

- ❖ Github

dingyiyi0226/standard-p1935

System Structure

- ❖ API Server

Flask. Expose the api

- ❖ EFO

The core system

- ❖ Database

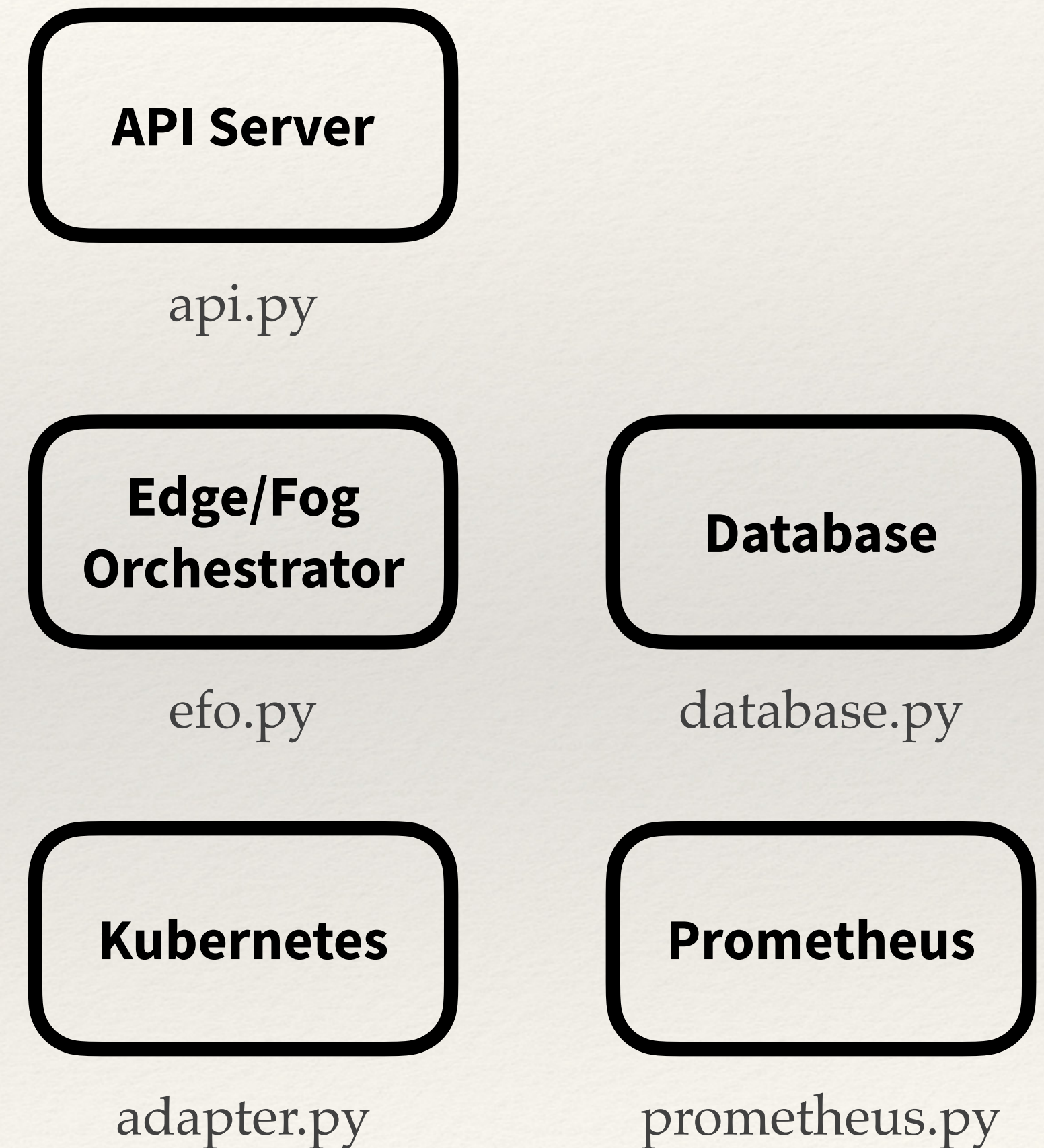
MongoDB. Store the manifests, apps, and nodes information

- ❖ Prometheus

Prometheus-operator. Monitor nodes and retrieve real time information

- ❖ Kubernetes

Kubernetes-python-client. Communicate with the kubernetes clusters



System Installation

- ❖ Install python 3.7 with requirements

```
pip install -r requirements.txt
```

- ❖ Install MongoDB

- ❖ Add users in database

```
mongo < init_db.js
```

Start the system

- ❖ (optional) Switch the user to wmlab (for VM related)

`su - wmlab`

- ❖ Activate the python environment

`conda activate 1935env`

- ❖ Start the system in development/production mode

`./run_dev.sh` or `./run_prod.sh`

- ❖ Go to `http://192.168.50.166:5000` in browser

Login Accounts

Name	Password	Identity
provider	prov	Service provider
operator	oper	Service operator
enduser	end	End User
owner	own	Infrastructure owner

Defined in init_db.js

Common Errors

- ❖ Starting script errors, address already in use
 - Kill the process or change the port in the starting script
- ❖ Package not found error
 - Activate the python environment
- ❖ Other errors
 - Check the VM/Kubernetes/MongoDB status

Conclusions

- ❖ Testbed introduction
- ❖ Testbed RESTful APIs
- ❖ Testbed Implementation and UI
- ❖ Testbed usage