DISCA
DEPARTAMENT D'INFORMÀTICA DE SISTEMES I COMPUTADORS

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

GRC GRUPO DE REDES DE COMPUTADORES

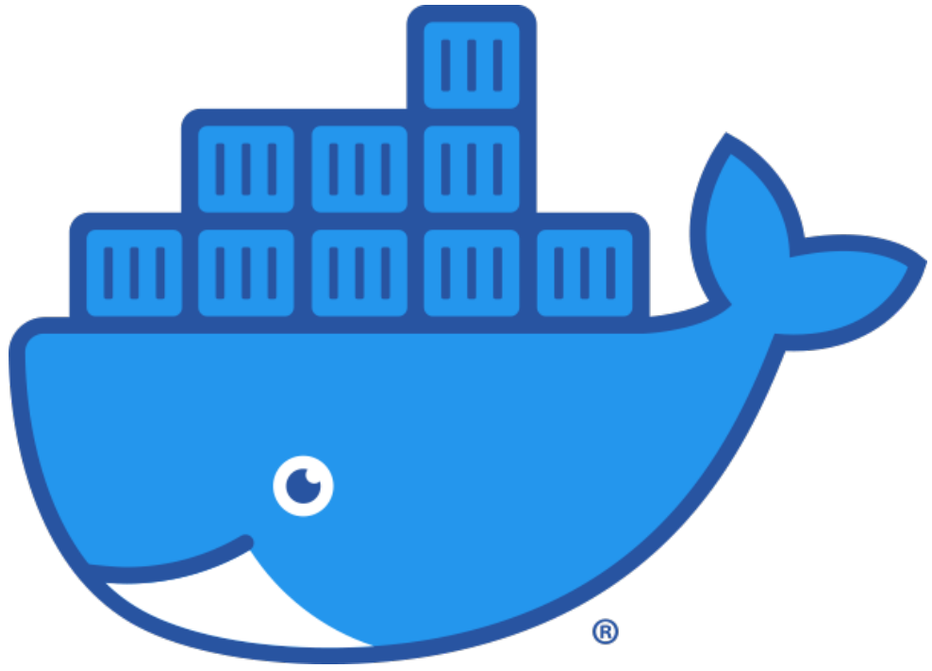
Docker for IoT

1DT305. Summer 2021

Course outline

- This course will focus on the use of Docker for the Development of microservices for the IoT.
- The course is organized as a succession of theoretical and experimental (hands-on) parts.

https://www.docker.com/get-started

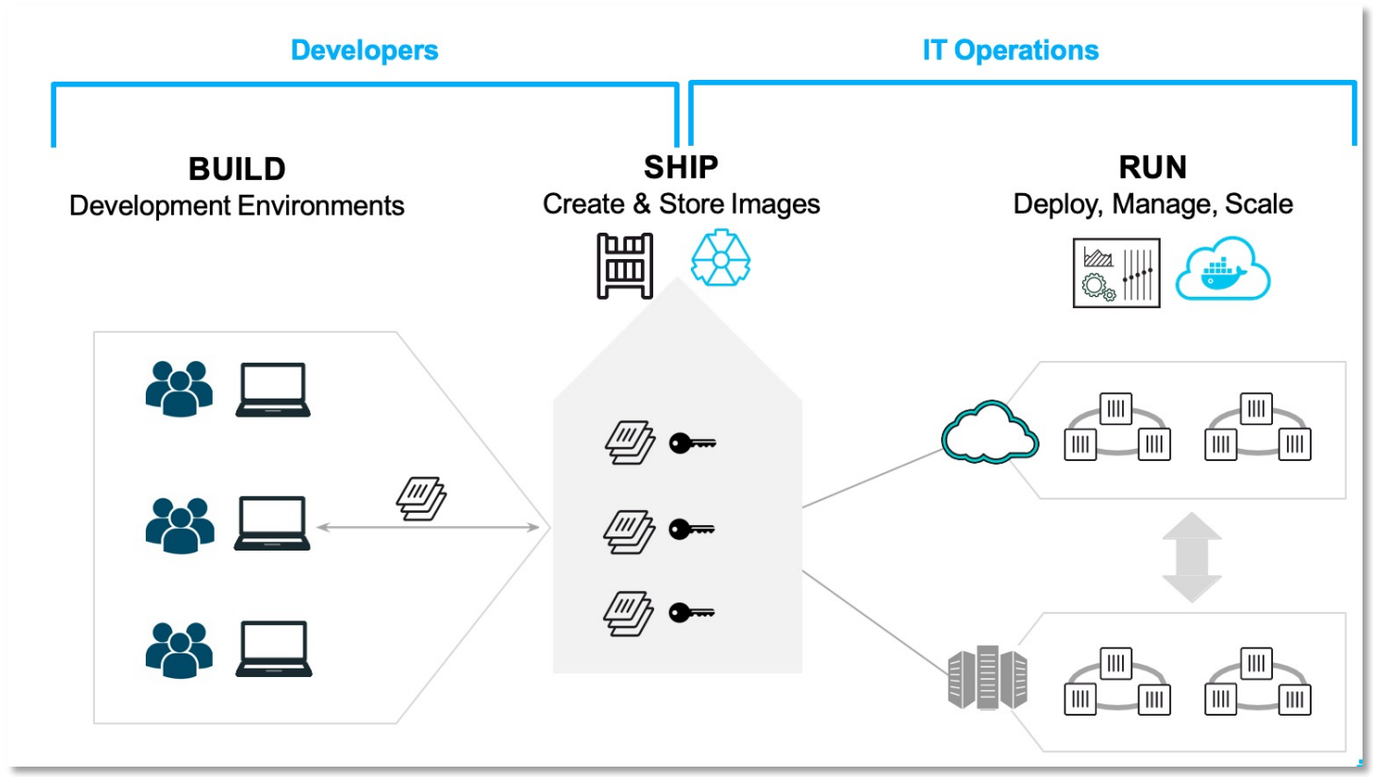


How to follow the course

- Option 1:
  - You can download and install Docker on multiple platforms.
  - Refer to the following link: https://docs.docker.com/get-docker/ and choose the best installation path for you.
- Option 2:
  - You can use a virtual machine that you can download here:
  - https://www.dropbox.com/s/frur6hfwanc9fwl/ubudocker.ova?dl=0
  - You have to use VirtualBox with the option: “Files / Import virtualized service”.
  - The VM has user “`docker`” with password “`docker`”
- Option 3:
  - You can execute it online: https://labs.play-with-docker.com/

Docker overview

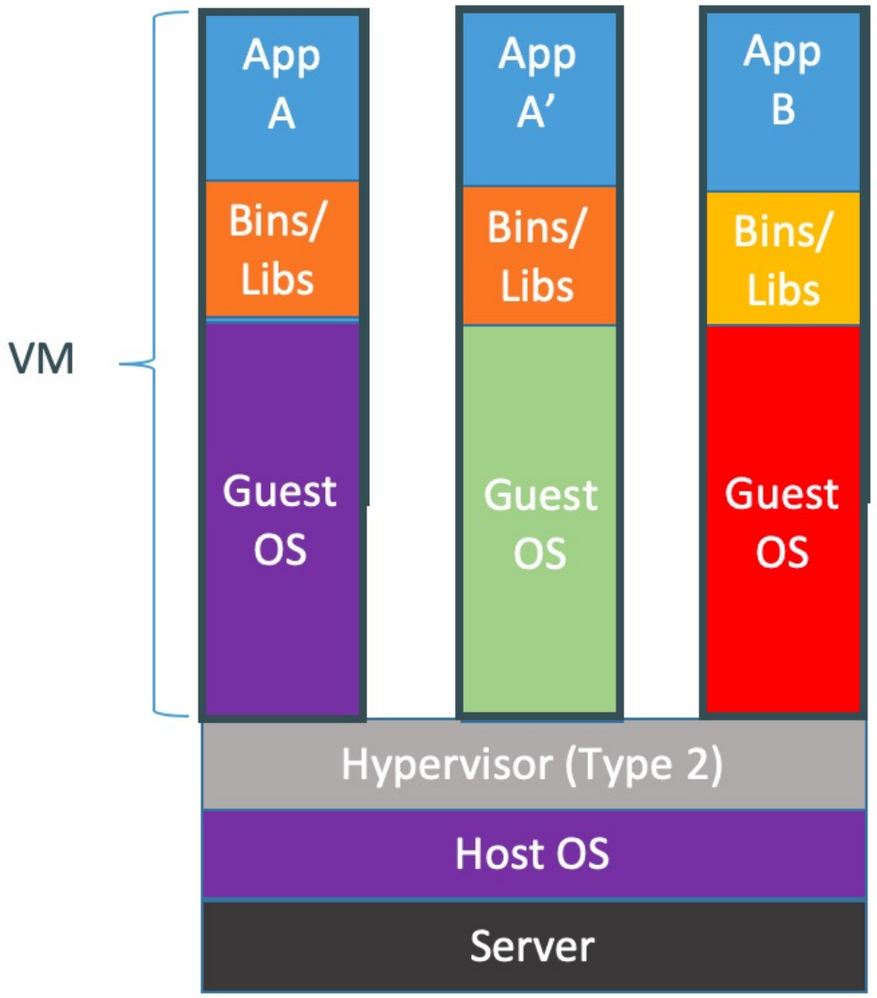
- Docker is an open platform for developing, shipping, and running applications.



- Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.
- https://docs.docker.com/get-started/overview/

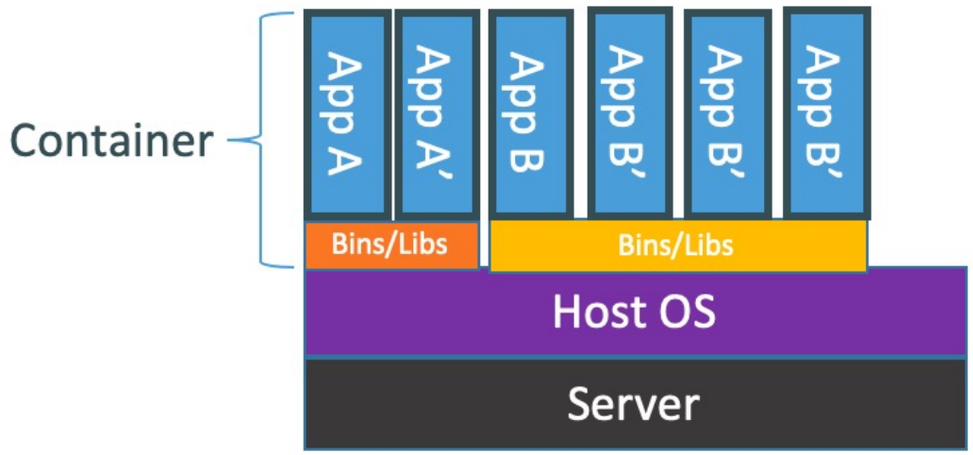
GRC GRUPO DE REDES DE COMPUTADORES

Docker vs. Virtual Machine



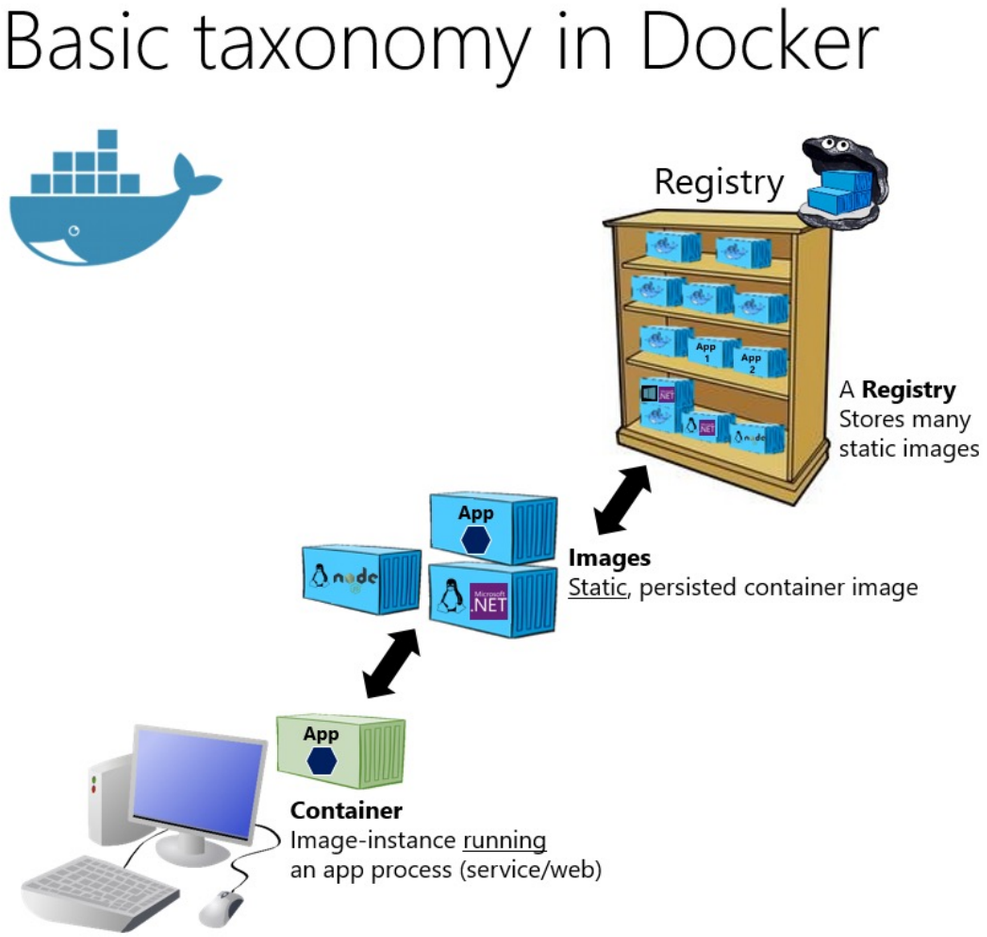
Containers are isolated, but share OS kernel and, where appropriate, bins/libraries

…result is significantly faster deployment, much less overhead, easier migration, faster restart



Deployment and scaling

- Docker containers can run on a developer's local laptop, on physical or virtual machines in a data center, on cloud providers, or in a mixture of environments.
- Docker's portability and lightweight nature also make it easy to **dynamically manage workloads**, scaling up or tearing down applications and services as business needs dictate, in near real time.

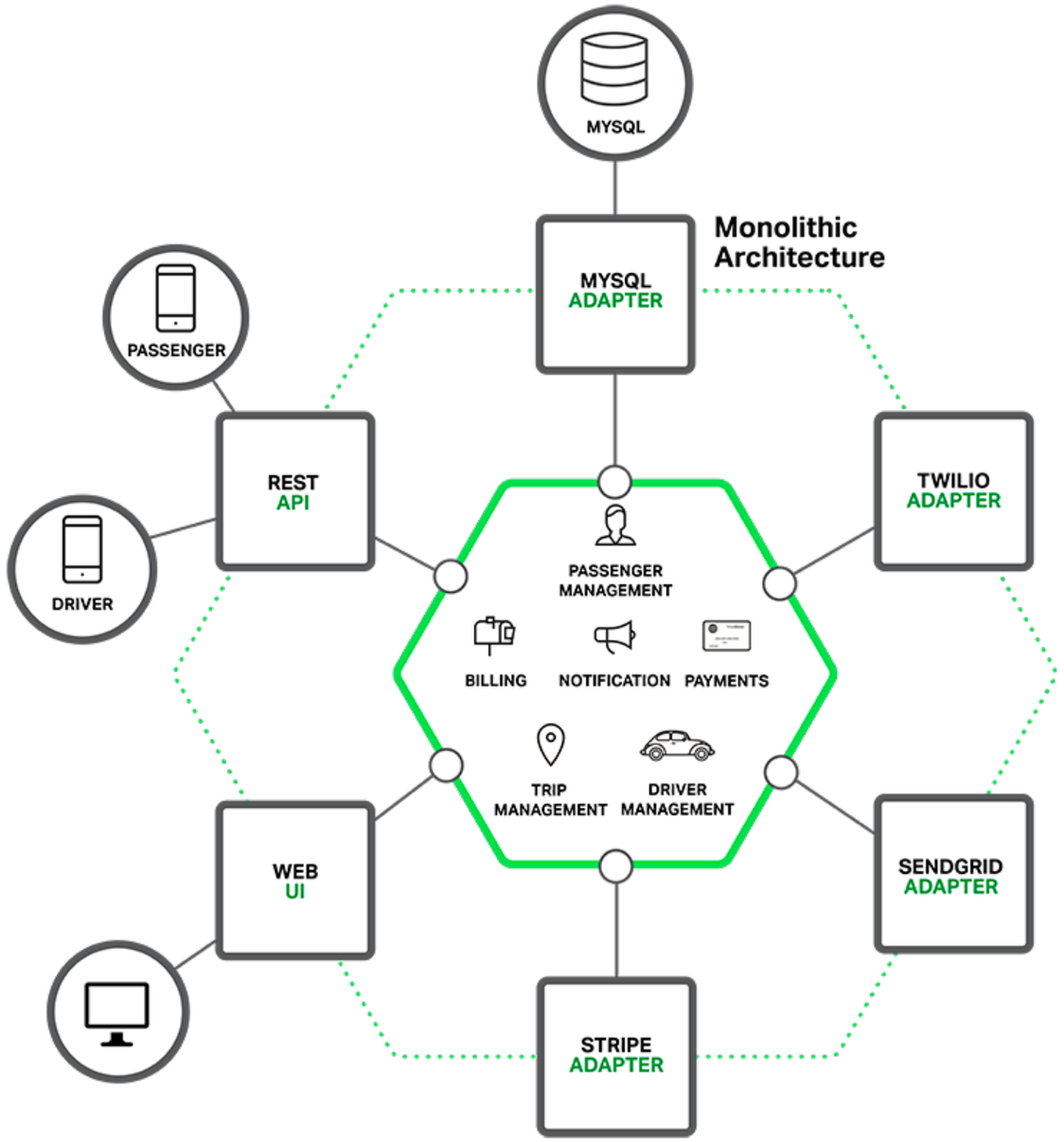


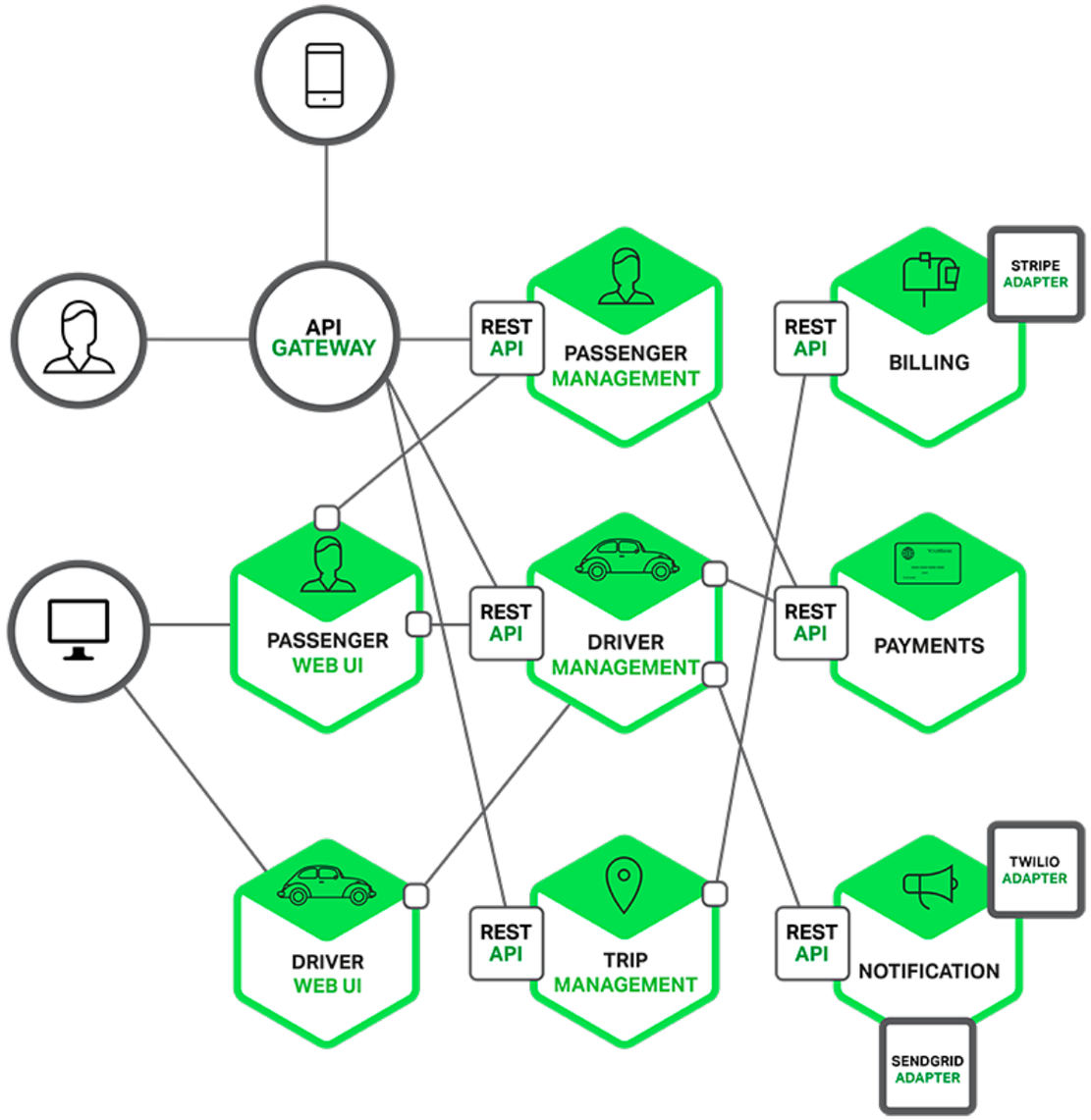
GRC GRUPO DE REDES DE COMPUTADORES

Multi-Cloud Platforms

- One of Docker's greatest benefits is portability. You may create your IoT application locally and deploy it in Public Cloud platform of your choice. Docker containers can be run inside an any Cloud instance provided that the host OS supports Docker.
- Docker Cloud: Docker's official cloud service for Swarm provisioning, managed Registry Service.
- Amazon EC2 Container Service: A highly scalable, high-performance container management service that supports Docker containers and allows to run applications on a managed cluster of Amazon EC2 instances.
- Microsoft Azure Container Service: Offers Docker container orchestration and scale operations. Allows the use of the Mesos-based DC/OS, Kubernetes, or the basic Docker Swarm and Compose.
- Google Container Engine for Docker Containers: Kubernetes clusters, managed by Google. Container Engine is a powerful cluster manager and orchestration system for running Docker containers.

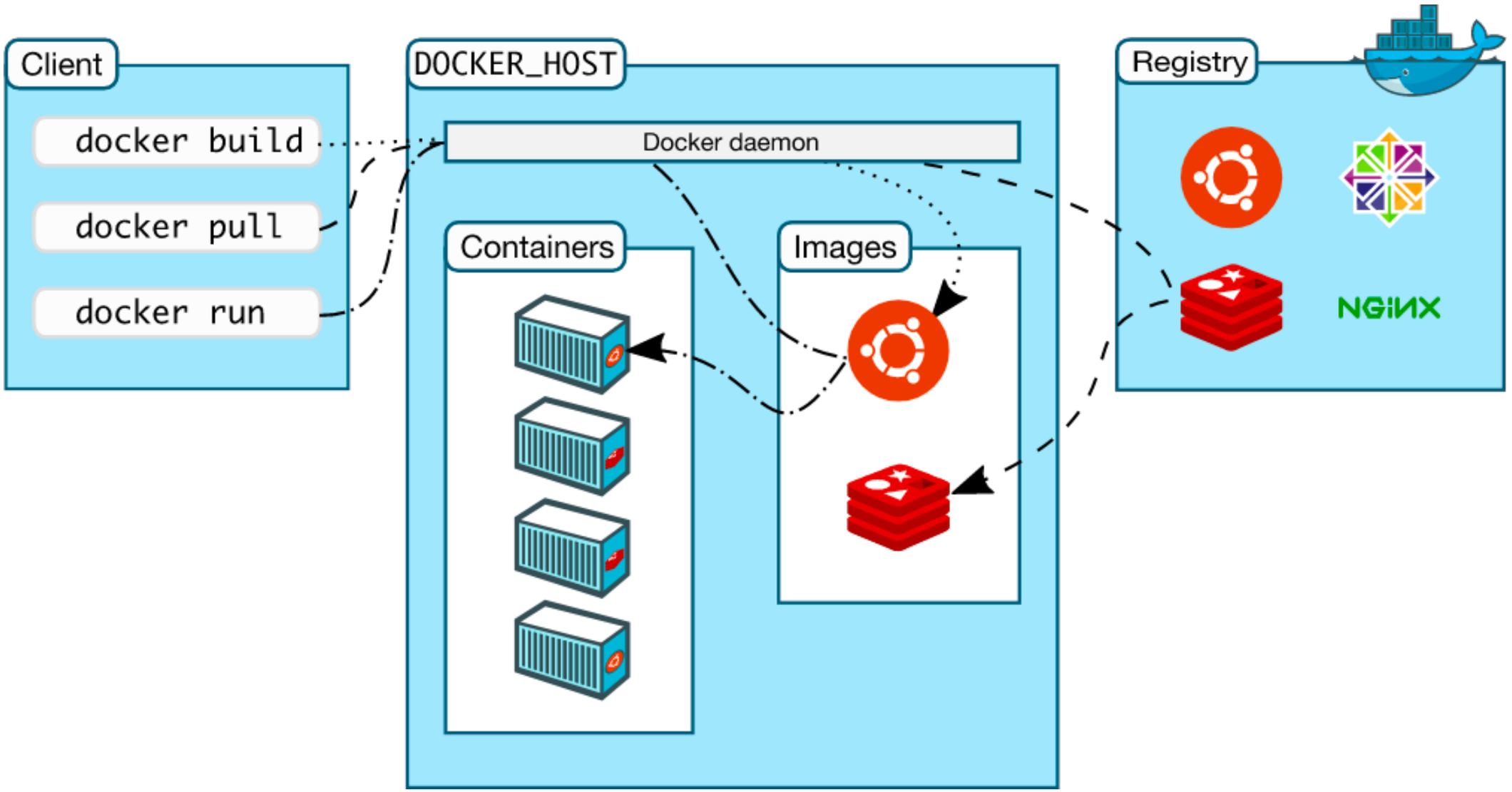
Monolithic vs. Microservices





GRC GRUPO DE REDES DE COMPUTADORES

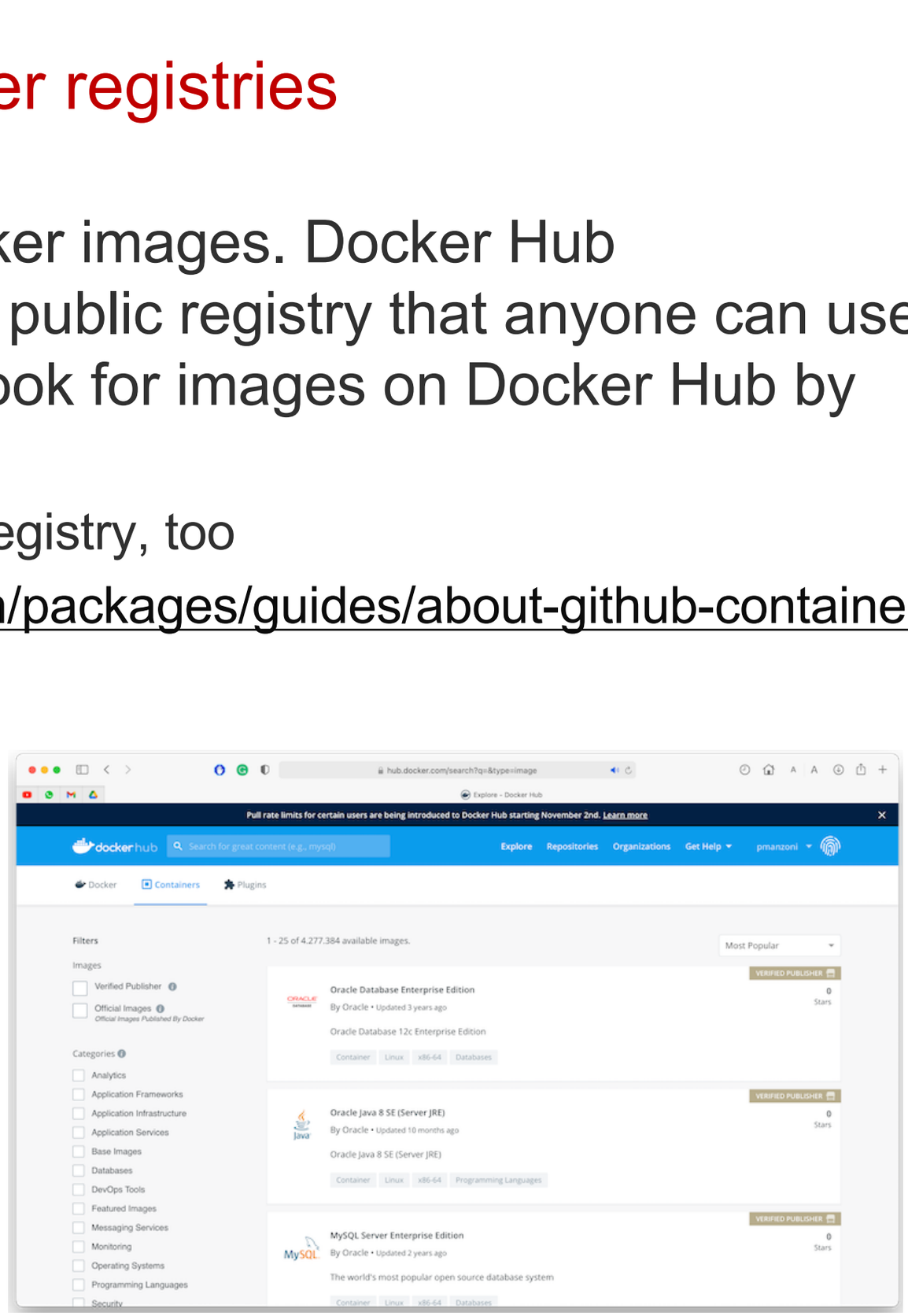
Docker architecture



GRC GRUPO DE REDES DE COMPUTADORES

Docker registries

- A Docker registry stores Docker images. Docker Hub (https://hub.docker.com/) is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default.
  - You can run your own private registry, too
  - E.g., https://docs.github.com/en/packages/guides/about-github-container-registry



Images

- An image is a read-only template with instructions for creating a Docker container. Often, an image is based on another image, with some additional customization.
  - For example, you may build an image which is based on the ubuntu image, but installs the Apache web server and your application, as well as the configuration details needed to make your application run.
- You might create your own images or you might only use those created by others and published in a registry.
- To build your own image, you create a Dockerfile with a simple syntax for defining the steps needed to create the image and run it.
  - Each instruction in a Dockerfile creates a **layer** in the image.
  - When you change the Dockerfile and rebuild the image, only those layers which have changed are rebuilt.

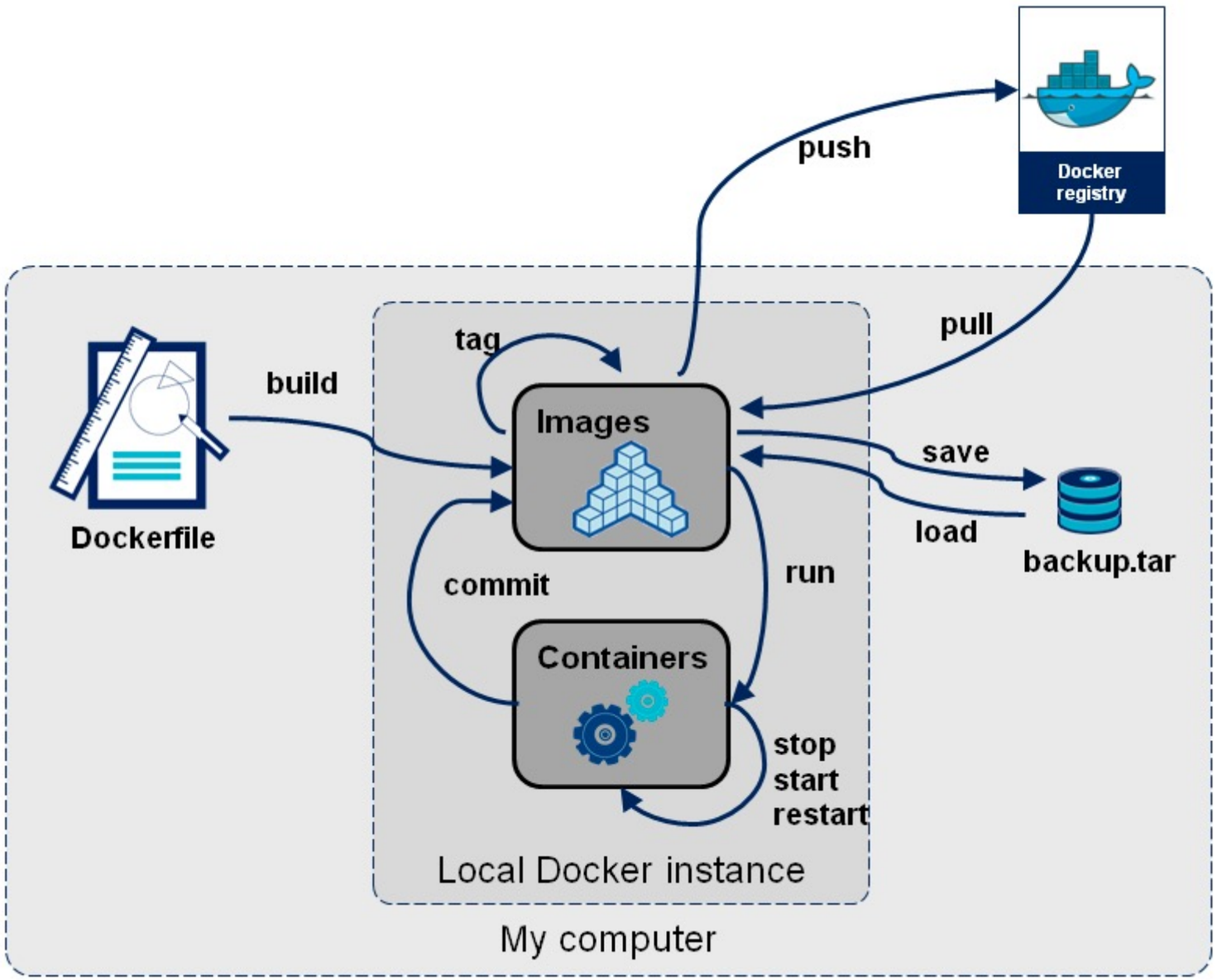
Dockerfile example

```
FROM alpine:3.5

RUN apk add --update py2-pip

RUN pip install paho-mqtt

COPY sisub.py /home/

CMD ["python", "/home/sisub.py"]
```


Containers

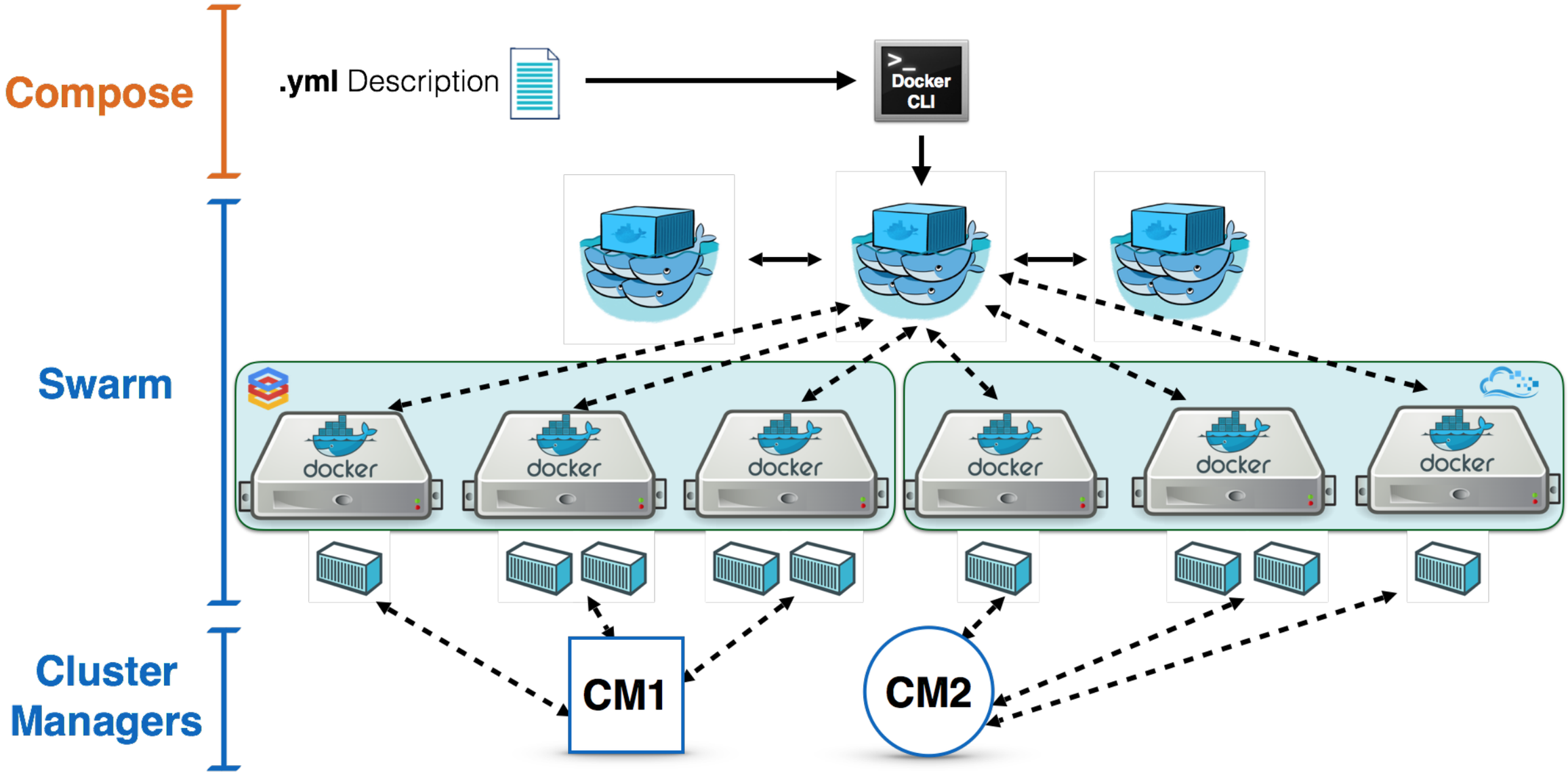
- A container is **a runnable instance of an image**. You can create, start, stop, move, or delete a container using the Docker API or CLI. You can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.
- By default, **a container is relatively well isolated from other containers and its host machine**. You can control how isolated a container's network, storage, or other underlying subsystems are from other containers or from the host machine.
- A container is defined by its image as well as any configuration options you provide to it when you create or start it. **When a container is removed, any changes to its state that are not stored in persistent storage disappear.**

Manage the lifecycle of your containers



GRC GRUPO DE REDES DE COMPUTADORES

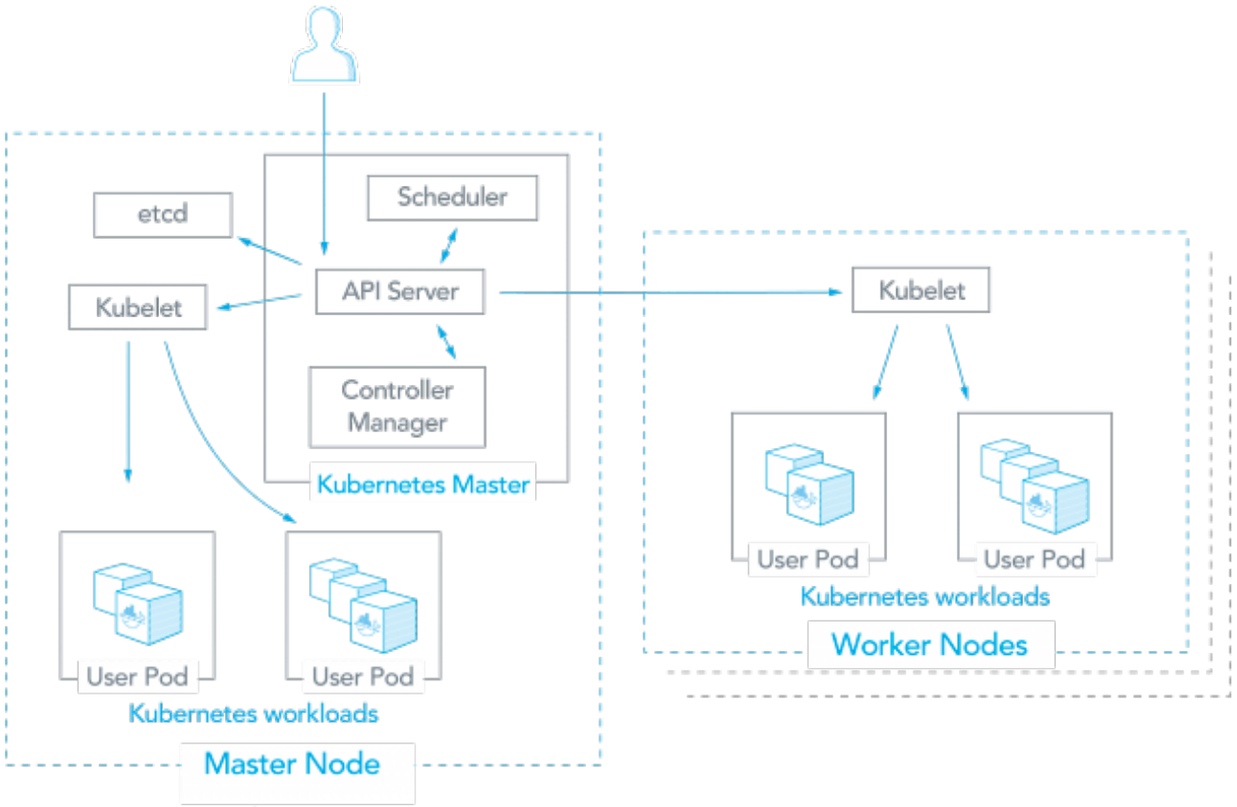
Docker Swarm



GRC GRUPO DE REDES DE COMPUTADORES

Kubernetes

- Developed by Google and introduced in the year 2014.
- An IT management tool that has been specifically designed to simplify the scalability of workloads using containers.
- It has the ability to automate deployment, scaling, and operating application containers.



API Server: management hub for Kubernetes
Scheduler: places a workload on the appropriate Node
Controller Manager: scales workloads up/down
etcd: stores configuration data which can be accessed by API Server

Kubelet: Receives pod specifications from API Server, updates Nodes
Master Node: places workloads on Nodes
Worker Nodes: receives requests from Master Nodes and dispatches them
User Pod: a group of containers with shared resources