UNIT-6

Abhishek Bhattacherjee

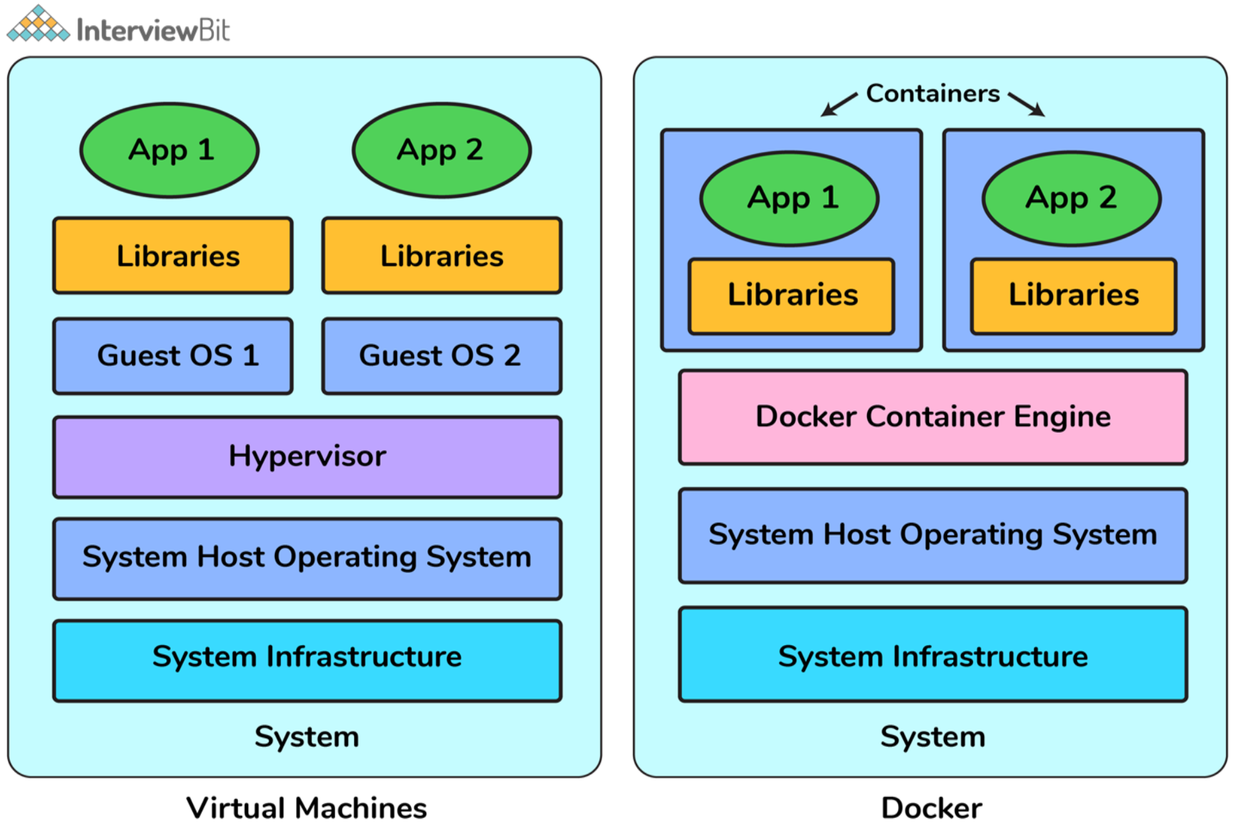
Contents

Containers vs. Virtual Machines

When compared to Virtual machines, the Docker platform moves up the abstraction of resources from the hardware level to the Operating System level. This allows for the realization of the various benefits of Containers e.g. application portability, infrastructure separation, and self-contained microservices.

Advantages

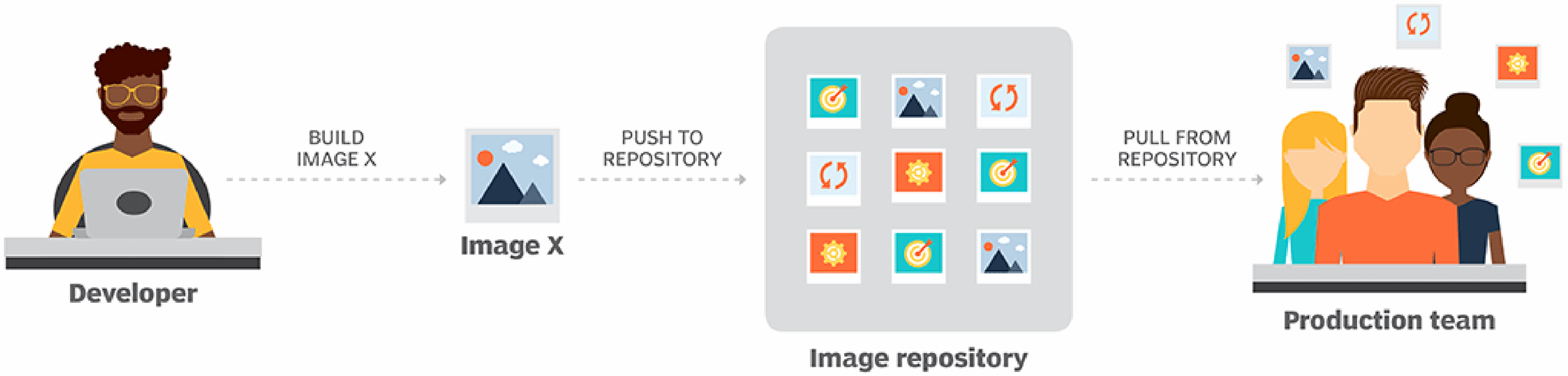
- **Resource Efficiency**:.
- **Portability**:
- **Continuous Deployment and Testing**:



Introduction to Containers

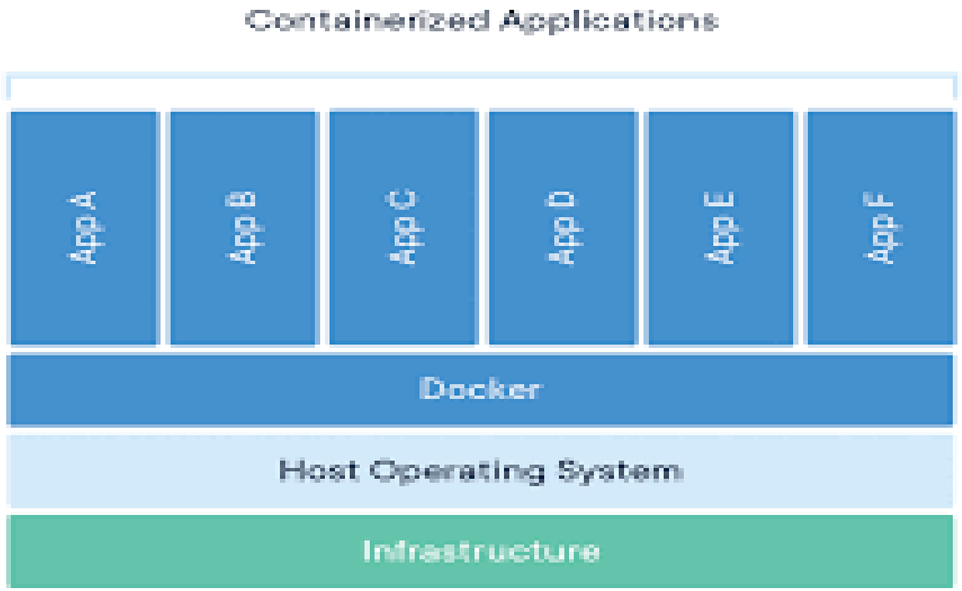
- A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.
- A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.
- Container images become containers at runtime and in the case of Docker containers - images become containers when they run on Docker Engine.

Container image system



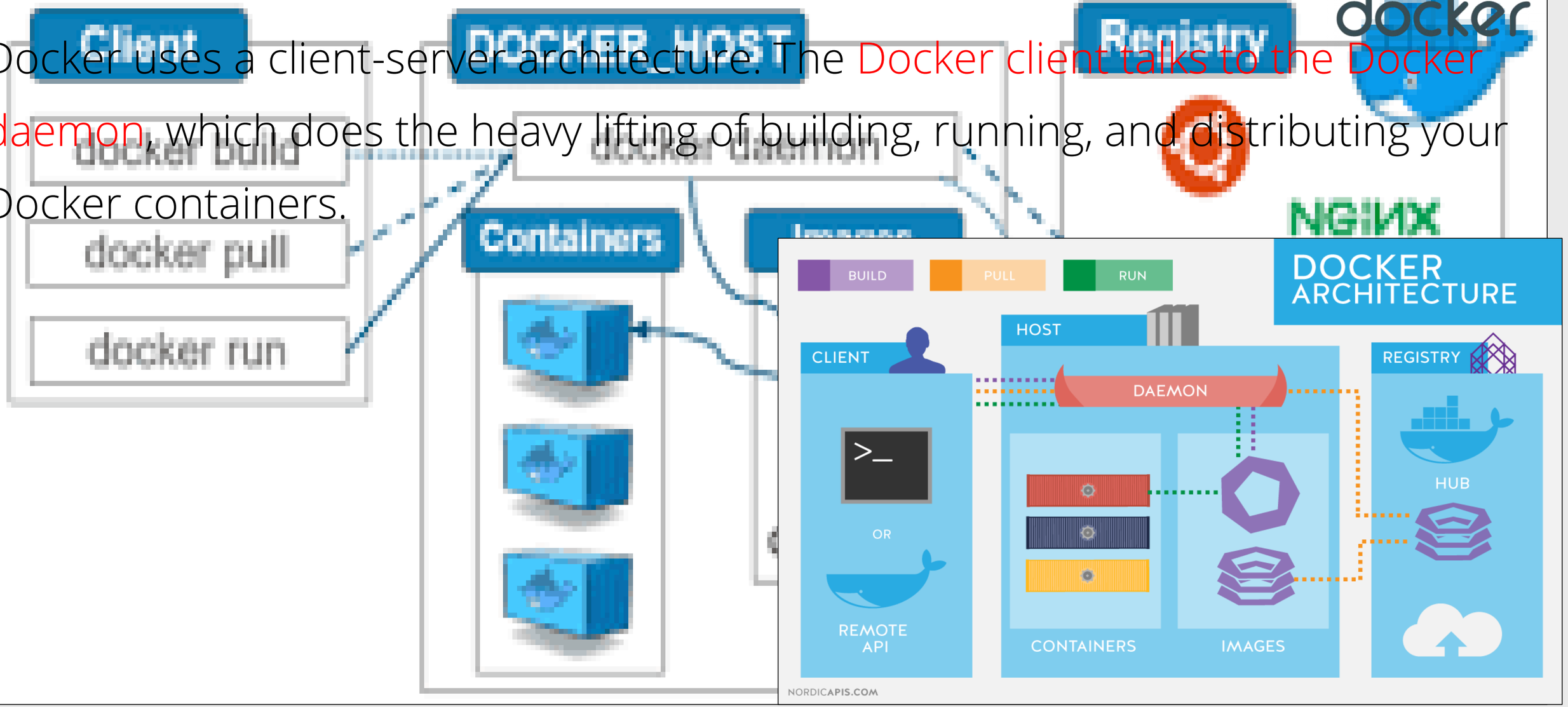
Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

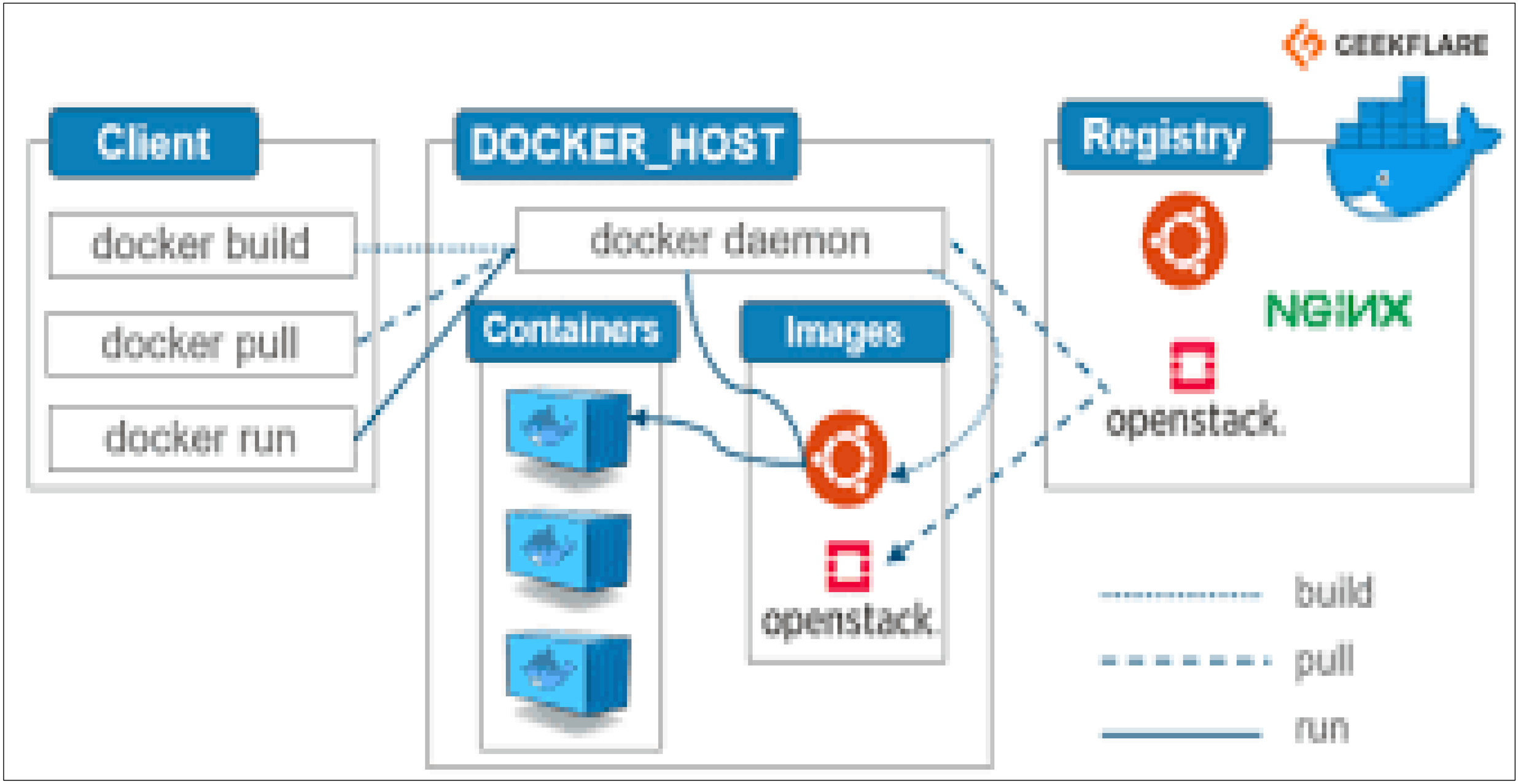
Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.



INTRODUCTION TO DOCKERS

Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers.




GEEKFLARE
Client
docker build
docker pull
docker run
DOCKER_HOST
docker daemon
Containers
Images
openstack.
Registry
NGINX
openstack.
build
pull
run

- **The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface.**
- **The Docker daemon**

The Docker daemon (dockerd) listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes.

- **The Docker client**

The Docker client (docker) is the primary way that many Docker users interact with Docker. When you use commands such as docker run, the client sends these commands to dockerd, which carries them out. The docker command uses the Docker API. The Docker client can communicate with more than one daemon.

- **Docker registries**
  - A Docker registry stores Docker images. Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default. When the docker pull or docker run commands, the required images are pulled from your configured registry.
- **Docker objects**
  - When you use Docker, you are creating and using images, containers, networks, volumes, plugins, and other objects. This section is a brief overview of some of those objects.
- **Images & containers**
  - 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.
  - 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.

- **Services**
  - Services allow you to scale containers across multiple Docker daemons, which all work together as a swarm with multiple managers and workers. Each member of a swarm is a Docker daemon, and the daemons all communicate using the Docker API.
- **Docker Engine**
  - Docker Engine is a client-server application with these major components:
- **A server which is a type of long-running program called a daemon process (the dockerd command).**
- **A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.**
- **A command line interface (CLI) client (the docker command).**

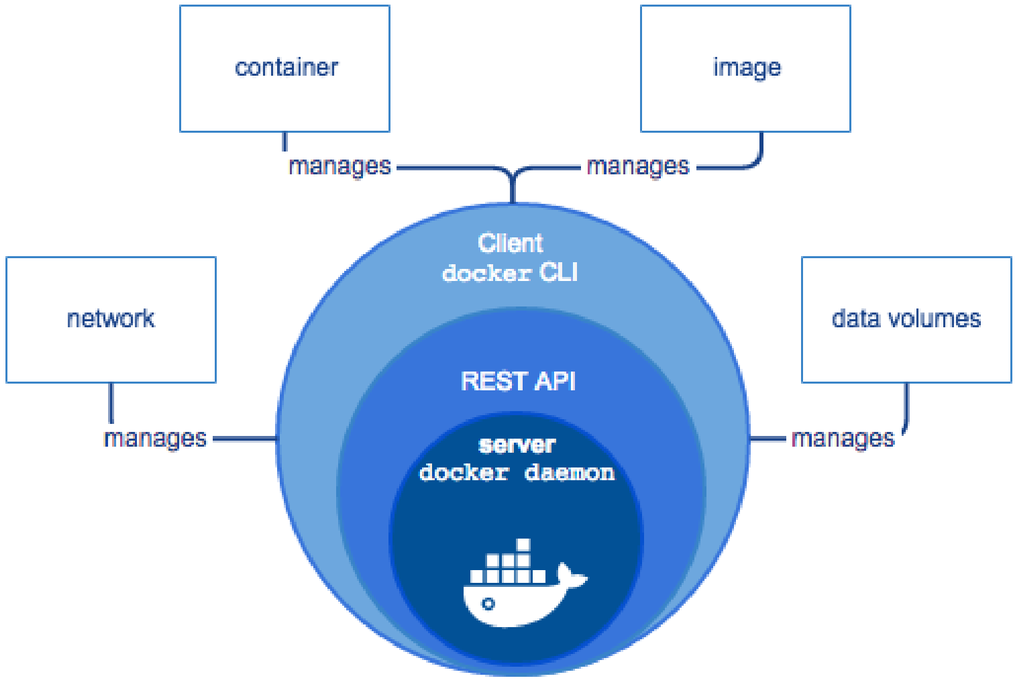
The Docker Engine

Docker Engine allows you to develop, assemble, ship, and run applications using the following components:

•**Docker Daemon**: A persistent background process that manages Docker images, containers, networks, and storage volumes. The Docker daemon constantly listens for Docker API requests and processes them.

•**Docker Engine REST API**: An API used by applications to interact with the Docker daemon; it can be accessed by an HTTP client.

•**Docker CLI**: A command line interface client for interacting with the Docker daemon. It greatly simplifies how you manage container instances and is one of the key reasons why developers love using Docker.



Monolithic Application

Recruitment Website

Job Applicants

Job Vacancies

Recruiters

Transition to Microservices

Recruitment Website

Recruiters

Job Applicants

Job Vacancies

Docker

Create containers for your application

Kubernetes

Launch your containerised application in K8s

Kubernetes

Kubernetes (K8s) is an open-source system for automating deployment, scaling, and management of containerized applications.

- Containerization helps package software to serve these goals, enabling applications to be released and updated in an easy and fast way without downtime.
- Kubernetes helps you make sure those containerized applications run where and when you want, and helps them find the resources and tools they need to work.
- Kubernetes is a production-ready, open source platform designed with Google's accumulated experience in container orchestration, combined with best-of-breed ideas from the community.
- A container orchestrator is essentially an administrator in charge of operating a fleet of containerized applications. If a container needs to be restarted or acquire more resources, the orchestrator takes care of it for you.

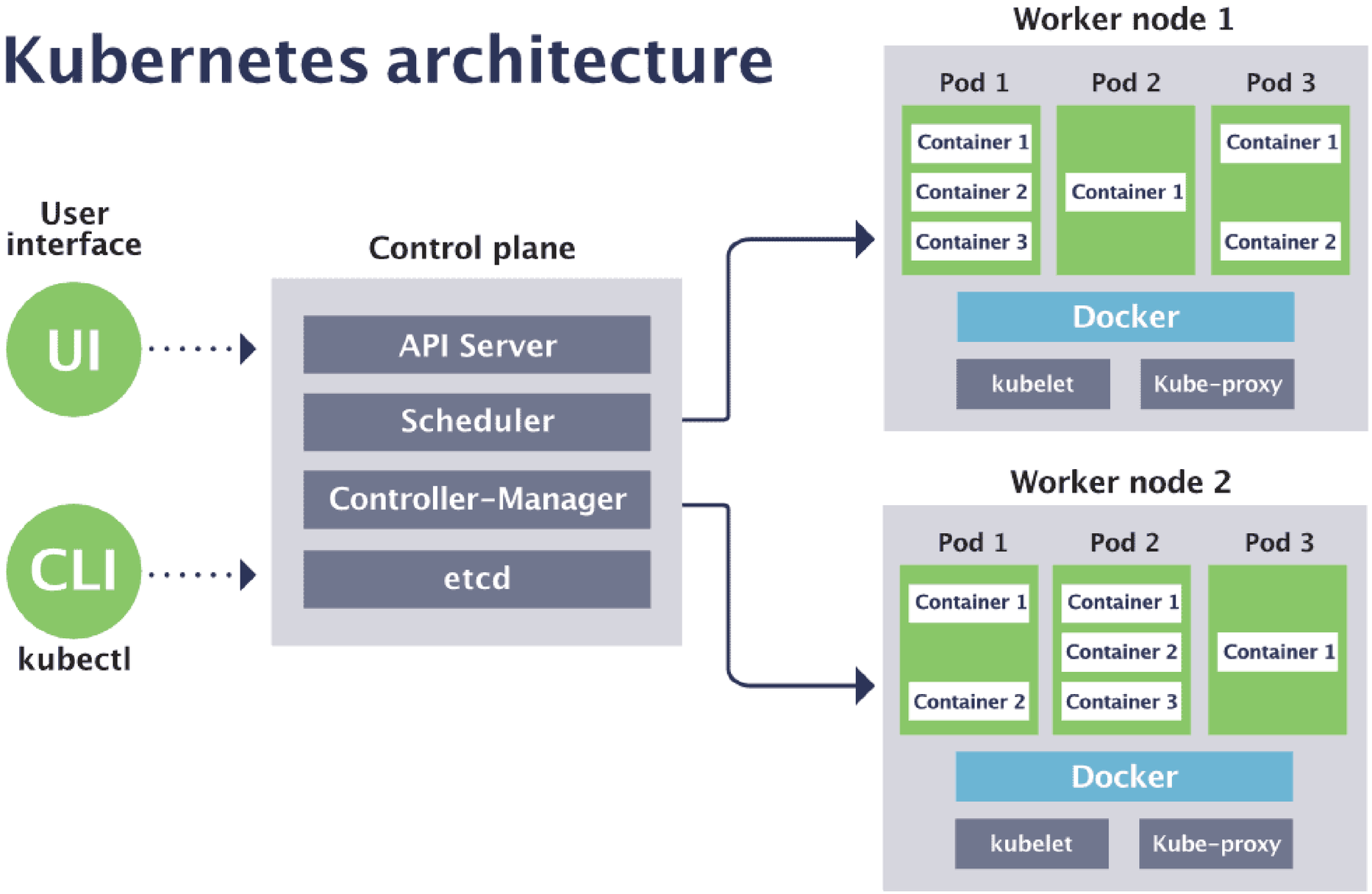
WHAT IS KUBERNETES USED FOR?

Kubernetes keeps track of your container applications that are deployed into the cloud. It restarts orphaned containers, shuts down containers when they’re not being used, and automatically provisions resources like memory, storage, and CPU when necessary.

How does Kubernetes work with Docker?

Actually, Kubernetes supports several base container engines, and Docker is just one of them. The two technologies work great together, since Docker containers are an efficient way to distribute packaged applications, and Kubernetes is designed to coordinate and schedule those applications.

Kubernetes architecture



Pods

- A Kubernetes pod is a group of containers, and is the smallest unit that Kubernetes administers.
- Pods have a single IP address that is applied to every container within the pod. Containers in a pod share the same resources such as memory and storage.
- It is common to have a pod with only a single container, when the application or service is a single process that needs to run.
- For example, if you were working on an image-processing service that created GIFs, one pod might have several containers working together to resize images.
- The primary container might be running the non-blocking microservice application taking in requests, and then one or more auxiliary (side-car) containers running batched background processes or cleaning up data artifacts in the storage volume as part of managing overall application performance.

Deployments

- Kubernetes deployments define the scale at which you want to run your application by letting you set the details of how you would like pods replicated on your Kubernetes nodes. Deployments describe the number of desired identical pod replicas to run and the preferred update strategy used when updating the deployment. Kubernetes will track pod health, and will remove or add pods as needed to bring your application deployment to the desired state.

Services

- A service is an abstraction over the pods, and essentially, the only interface the various application consumers interact with. As pods are replaced, their internal names and IPs might change. A service exposes a single machine name or IP address mapped to pods whose underlying names and numbers are unreliable. A service ensures that, to the outside network, everything appears to be unchanged.

Nodes

- A Kubernetes node manages and runs pods; it’s the machine (whether virtualized or physical) that performs the given work. Just as pods collect individual containers that operate together, a node collects entire pods that function together. When you’re operating at scale, you want to be able to hand work over to a node whose pods are free to take it.

Control plane

- The Kubernetes control plane is the main entry point for administrators and users to manage the various nodes. Operations are issued to it either through HTTP calls or connecting to the machine and running command-line scripts. As the name implies, it controls how Kubernetes interacts with your applications.

Cluster

- A cluster is all of the above components put together as a single unit.

API server

- The API server exposes a REST interface to the Kubernetes cluster. All operations against pods, services, and so forth, are executed programmatically by communicating with the endpoints provided by it.

Scheduler

- The scheduler is responsible for assigning work to the various nodes. It keeps watch over the resource capacity and ensures that a worker node’s performance is within an appropriate threshold.

Controller Manager

- The controller-manager is responsible for making sure that the shared state of the cluster is operating as expected. More accurately, the controller manager oversees various controllers which respond to events (e.g., if a node goes down).

Kubelet

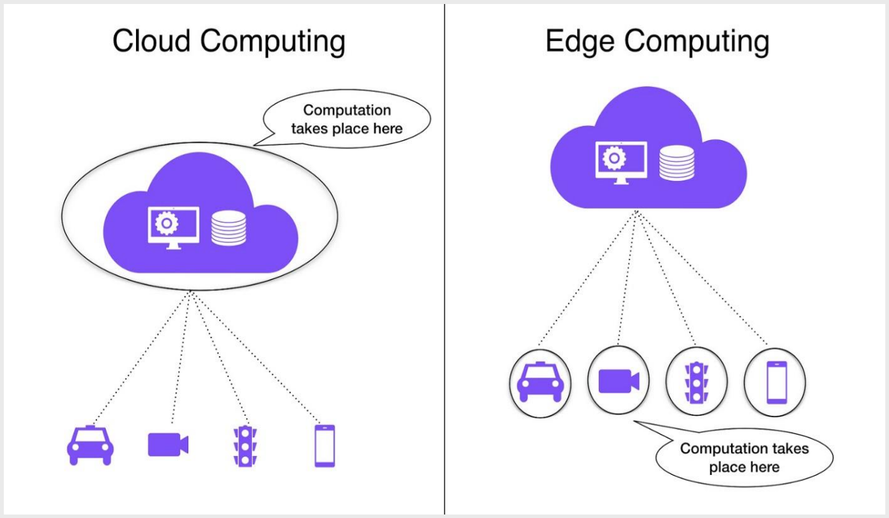
- A Kubelet tracks the state of a pod to ensure that all the containers are running. It provides a heartbeat message every few seconds to the control plane. If a replication controller does not receive that message, the node is marked as unhealthy.

Kube Proxy

- The Kube proxy routes traffic coming into a node from the service. It forwards requests for work to the correct containers.

Edge Computing

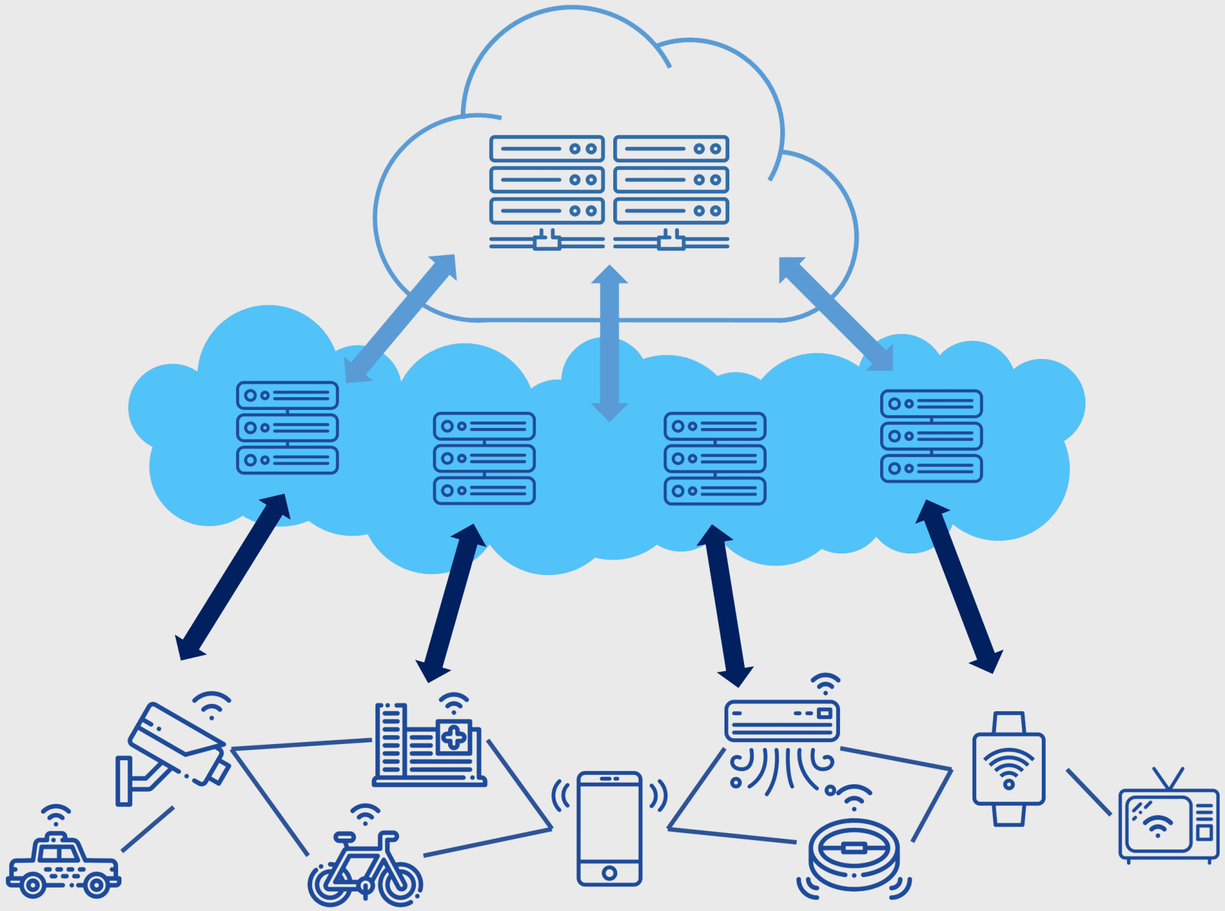
- A part of a distributed computing topology in which information processing is located close to the edge – where things and people produce or consume that information.”
- *Edge computing brings computation and data storage closer to the devices where it’s being gathered, rather than relying on a central location that can be thousands of miles away.* In addition, companies can save money by having the processing done locally, reducing the amount of data that needs to be processed in a centralized or cloud-based location.
- Edge computing was developed due to the exponential growth of IoT devices, which connect to the internet for either receiving information from the cloud or delivering data back to the cloud. And many IoT devices generate enormous amounts of data during the course of their operations.



Cloud

Edge nodes

Edge devices



Edge v/s Cloud

Edge and cloud computing have distinct features and most organizations will end up using both. Here are some considerations when looking at where to deploy different workloads.

Cloud Computing	Edge Computing
Non-time-sensitive data processing	Real-time data processing
Reliable internet connection	Remote locations with limited or no internet connectivity
Dynamic workloads	Large datasets that are too costly to send to the cloud
Data in cloud storage	Highly sensitive data and strict data laws

Fog Computing

The word "fog" refers to its cloud-like properties, but closer to the "ground",

- Fog computing can be perceived both in large cloud systems and big data structures, making reference to the growing difficulties in accessing information objectively.
- This results in a lack of quality of the obtained content. The effects of fog computing on cloud computing and big data systems may vary.
- However, a common aspect is a limitation in accurate content distribution, an issue that has been tackled with the creation of metrics that attempt to improve accuracy.
- Fog networking consists of a control plane and a data plane. For example, on the data plane, fog computing enables computing services to reside at the edge of the network as opposed to servers in a data-center.

Specialty	Cloud Computing	fog computing
Delay	Cloud computing has higher latency than fog computing	Fog computing has low latency
Capacity	Cloud computing does not provide any reduction in data while sending or converting data.	Fog computing reduces the amount of data sent to cloud computing.
Responsiveness	The response time of the system is low.	The response time of the system is high.
Security	Cloud computing has less Security compared to Fog Computing	Fog computing has high Security.
Speed	Access speed is high depending on the VM connectivity.	High even more compared to Cloud Computing.
Data Integration	Multiple data sources can be integrated.	Multiple Data sources and devices can be integrated.
Mobility	In cloud computing, mobility is Limited.	Mobility is supported in fog computing

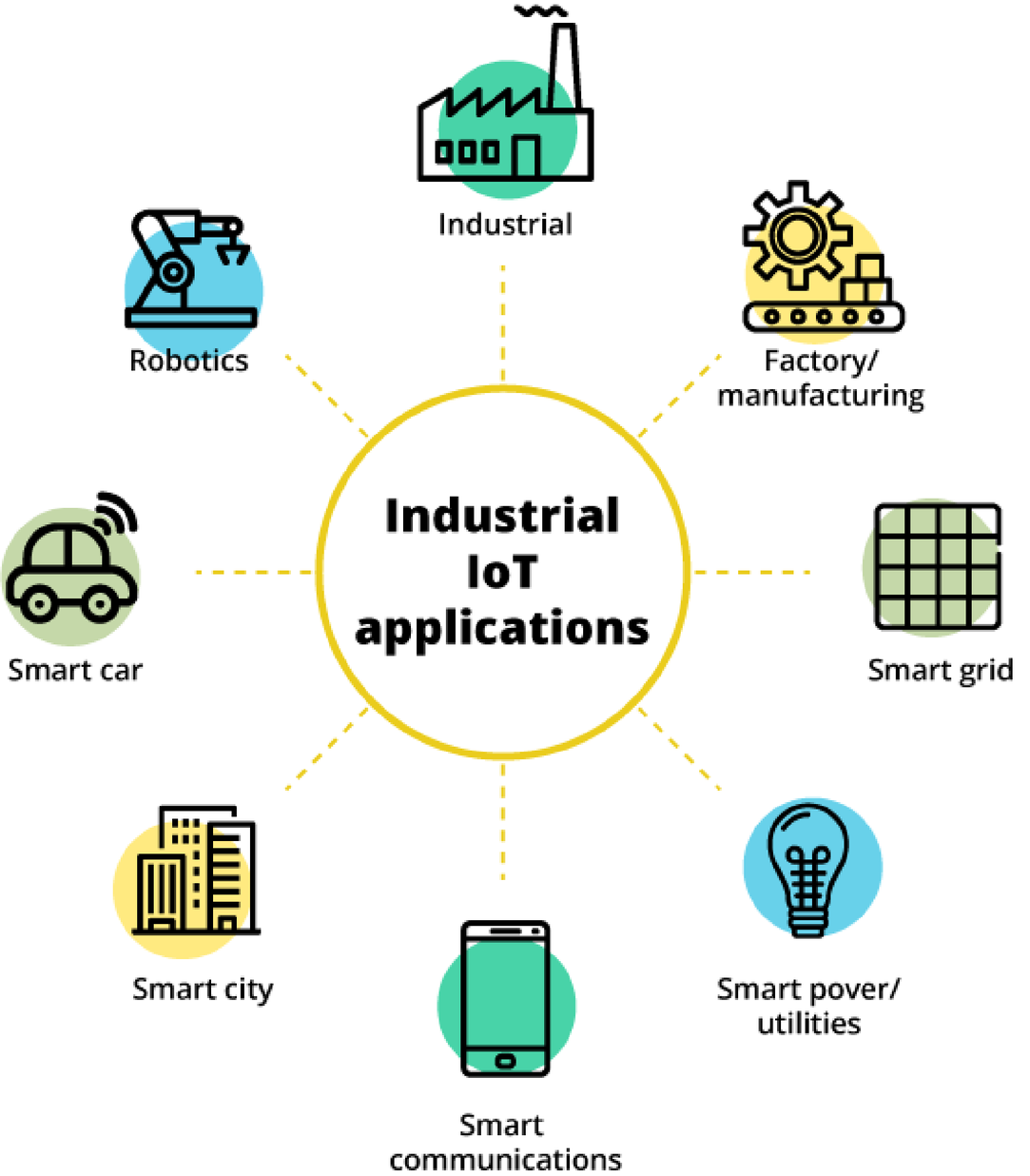
POINTS OF DIFFERENCE	Cloud Computing	Fog Computing	Edge Computing
LOCATION OF DATA PROCESSING	Central cloud server	Within an IoT gateway or fog nodes that are located in the LAN network.	Device itself
PURPOSE	Long term in-dept analysis	Quick analysis and real time response	Quick analysis and real time response
LATENCY	High	Low	Very Low
SECURITY	Less secure	High Security	High Security
GEOGRAPHICAL DISTRIBUTION	Centralized	Distributed	Distributed

IIOT

- The Industrial Internet of Things (IIoT) is a methodology, a practice, an implementation sweeping through businesses and industries worldwide. On a basic level, IIoT is a way of congregating /grouping data that was previously inaccessible and locked within inflexible data streams. This provides all stakeholders with a more complete and comprehensive view of operations.

- Imagine smart TVs and watches or security cameras - devices that were historically lacking in internet connection but now have that capability. This is IoT, the Internet of Things. IIoT is used to refer to industrial equipment and plant assets that are now integrated.

- Developments in technology are participle in the age of Industry 4.0, where real-time data is captured and made available within integrated digital ecosystems. Similarly, software is now increasingly platform independent , this means that plant floor information won't be originating from a single platform, but there will be a multitude of systems that needs to feed into a companies digital nervous system.


Industrial
Robotics
Factory/
manufacturing
Industrial
IoT
applications
Smart car
Smart grid
Smart city
Smart pover/
utilities
Smart
communications

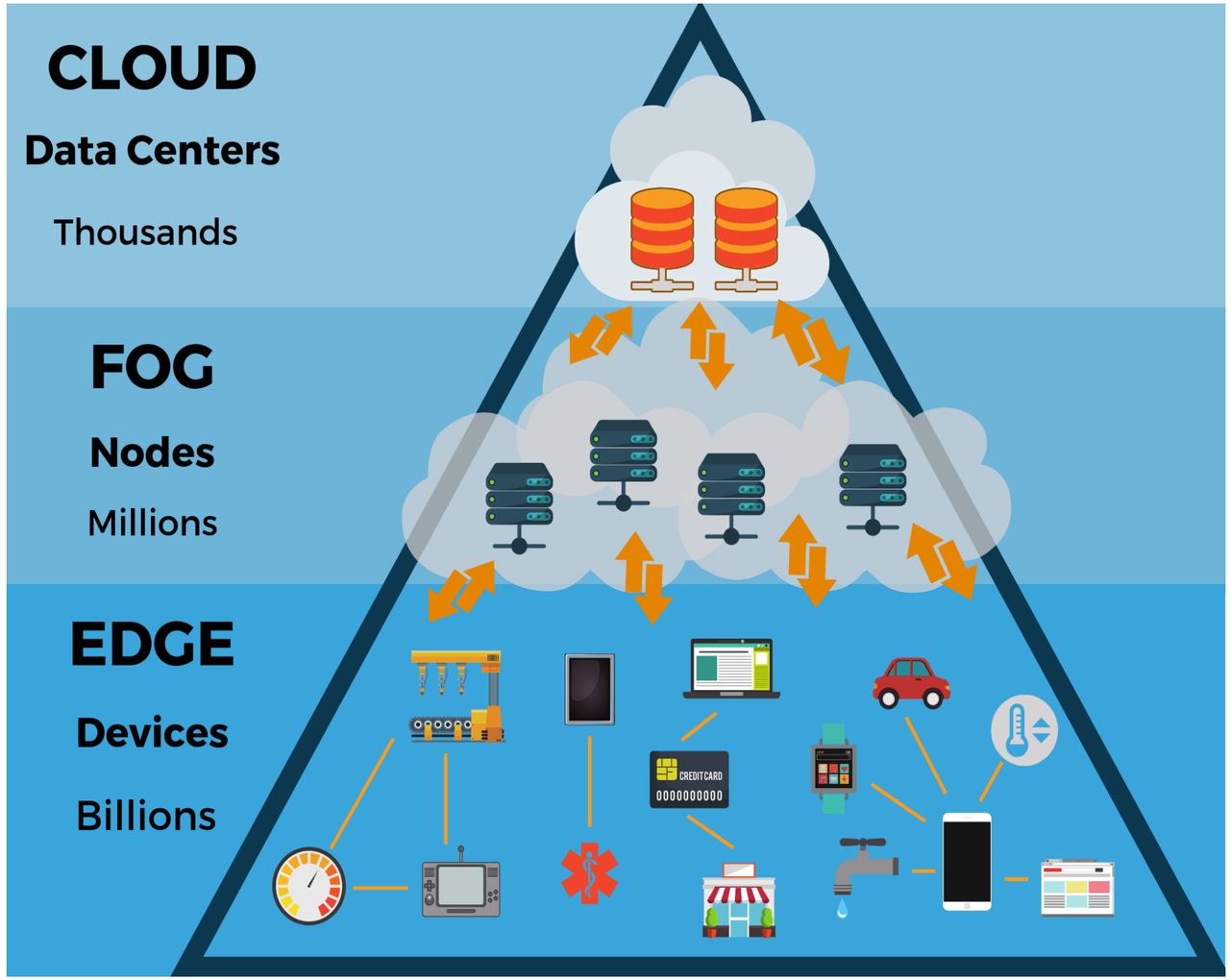
IIOT

- The concept behind cloud computing is a computer processor placed at remote location runs a specific application or software rather than running it on each user’s computer. User can interact with the system via a network on the cloud. Most of the times this network is Internet.

- Since the storage and processing capacity of the cloud is unlimited, it provides more economic, convenient and secure storage alternatives. The growing storage capacity can also be met as the cloud is expandable.

- In connected IIoT, data is expected to flow to and from the connected devices. These streams of data usually get larger with time and that is when Big Data comes in existence. Integration of Big Data into Industrial IoT can help to generate, access and gather the huge amount of data.

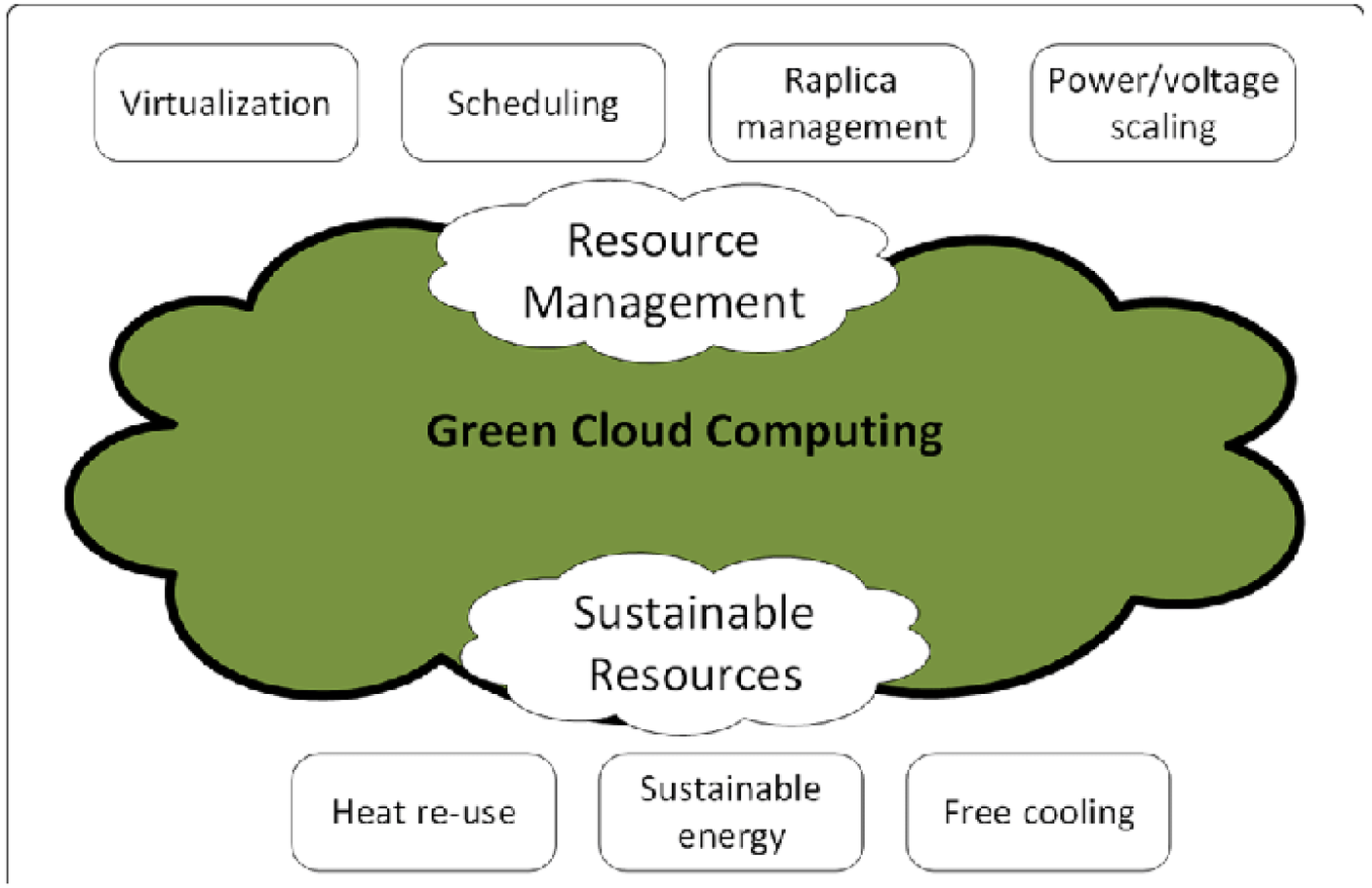
As the Internet of Things evolves, the rise of edge computing becomes inevitable. We need to start thinking how does edge computing fit our digital strategy and start building the implementation roadmap.



Green Cloud Computing

- With Green cloud computing, world is looking forward to higher energy efficient mechanism, managed security services, cloud security solutions at one place by offering equivalent and cloud management platform benefits with enormous environmental impact!
- Cloud computing, is an important facet for any IT operations of any organization, continuous attempts have been made to make it- much “greener”.
- “The green cloud”- certainly a superior marketing label, is employed by organizations for handling environmental considerations and concerns effectively. With contribution, towards the critical business operational goals and reduced costs across the servers, green cloud is most environment friendly initiative.


energy
ENERGY STAR


Virtualization
Scheduling
Raplica management
Power/voltage scaling
Resource Management
Green Cloud Computing
Sustainable Resources
Heat re-use
Sustainable energy
Free cooling

Factors for Green Computing

- Product longevity
  - Gartner maintains that the PC manufacturing process accounts for 70% of the natural resources used in the life cycle of a PC. More recently, Fujitsu released a Life Cycle Assessment (LCA) of a desktop that show that manufacturing and end of life accounts for the majority of this desktop's ecological footprint. Create a product which can support for a longer period of time.
- Data center design
  - Energy efficient data center design should address all of the energy use aspects included in a data center: from the IT equipment to the HVAC equipment to the actual location, configuration and construction of the building.

- Resource allocation
  - Algorithms can also be used to route data to data centers where electricity is less expensive. .Larger server centers are sometimes located where energy and land are inexpensive and readily available. Local availability of renewable energy, climate that allows outside air to be used for cooling, or locating them where the heat they produce may be used for other purposes could be factors in green siting decisions.
- Power management
  - The Advanced Configuration and Power Interface (ACPI), an open industry standard, allows an operating system to directly control the power-saving aspects of its underlying hardware. This allows a system to automatically turn off components such as monitors and hard drives after set periods of inactivity. In addition, a system may hibernate, when most components (including the CPU and the system RAM) are turned off. ACPI is a successor to an earlier Intel-Microsoft standard called Advanced Power Management

- Materials recycling
  - Recycling computing equipment can keep harmful materials such as lead, mercury, and hexavalent chromium out of landfills, and can also replace equipment that otherwise would need to be manufactured, saving further be given for recycling, and they typically sign a non-disclosure agreement.
- Algorithmic efficiency
  - The efficiency of algorithms has an impact on the amount of computer resources required for any given computing function and there are many efficiency trade-offs in writing programs. Algorithm changes, such as switching from a slow (e.g. linear) search algorithm to a fast (e.g. hashed or indexed) search algorithm can reduce resource usage for a given task from substantial to close to zero.