CGI

Experience the commitment®

From Zero to Docker

Mario Dagot – Software Architect
Hugo Calado – Software Architect
Jorge Dias – Software Architect

About the Speakers

Mário Dagot is a software architect at CGI Portugal with over 15 years of experience in the IT industry.

Has worked as a software and system engineer, backend and web developer. Most recently has been part of a DevOps team focusing on improving processes and breaking down silos between development and operation teams.

His passions are, hanging out with family and friends, squash, gym, photography and all that is IT related.

About the Speakers

Jorge Dias is an IT Consultant with more than 15 years of experience, with a technical background, acting with architect functions, as technical and functional analyst and programmer.

In last 4 years has worked in a nearshore cooperation with Finland CGI, what give him the opportunity to improve his experience on the area of the Continuous Integration and Systems Management.

His passion is sharing some time with family and friends. IT enthusiastic that always try to improve and expand his knowledge.

About the Speakers

Hugo Calado is an IT Consultant with more than 15 years of experience.

In last years have been involved in the development of CGI Sm@rtering product. Have joined CGI Global Delivery Center a year ago where he has been working in Mobilog product along with CGI Finnish team.

His passion is spending time with family and friends, travel, mountain biking, swimming and relax at the beach.

What is a container

A container is a software package that contains everything it needs to run.
This includes the executable program as well as **system tools**, **libraries**, and **settings**.

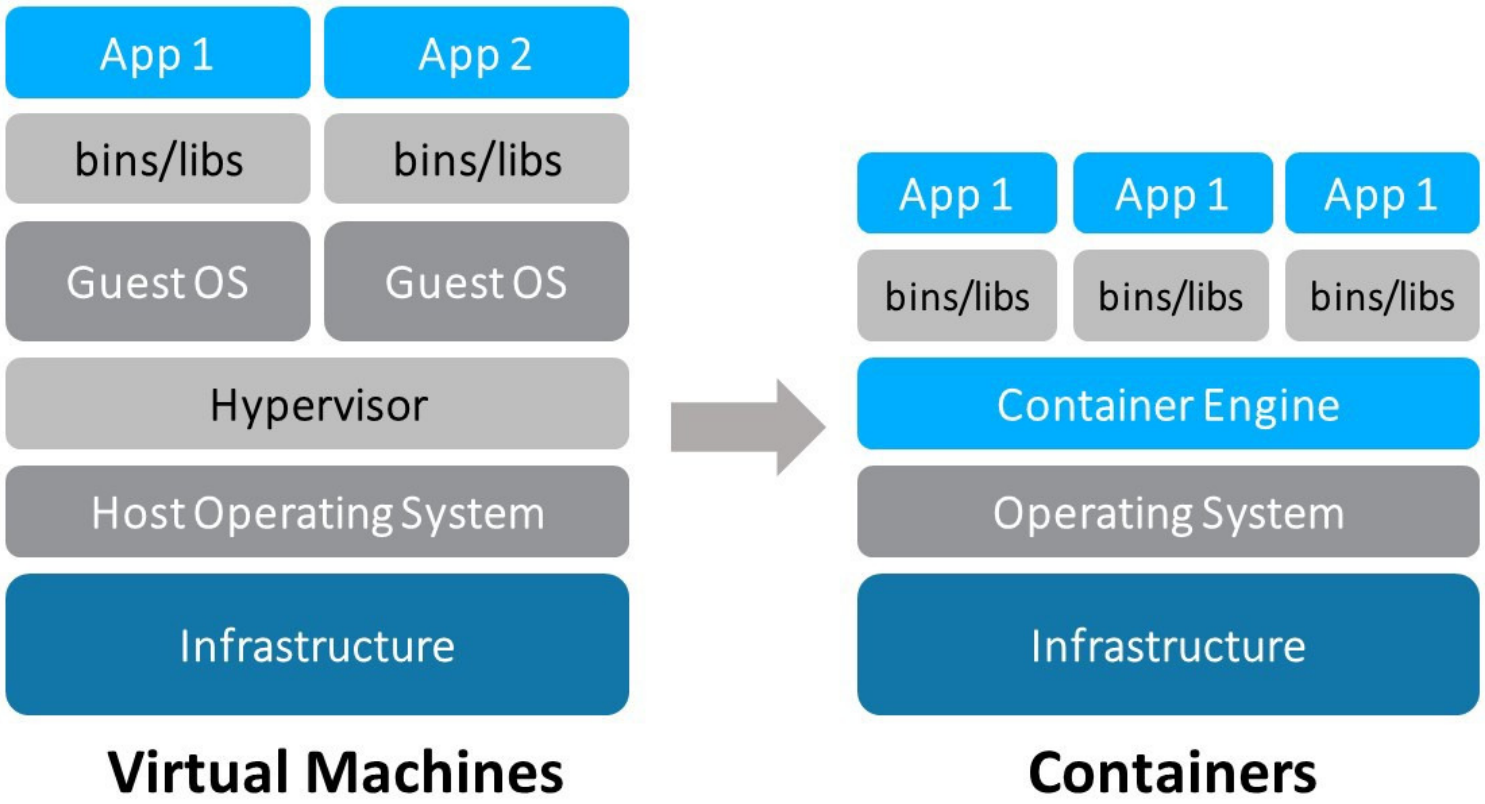
Strong Points:

- Agile environment
- Enhanced productivity
- Version control
- Computing environment portability
- Standardization
- Secure

Weak Points:

- Increased complexity
- Private data inside container images
- Persistent storage
- Native Linux support

Comparing Containers and Virtual Machines



What is Docker

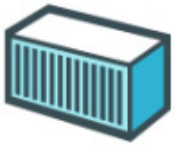
Docker is a computer program that performs operating-system-level virtualization, also known as "containerization“.

Docker allows to package an application into a standardized unit for software development: – The Docker Container.

Docker promise: Build, Ship, Run!

Build

Develop an app using Docker containers with any language and any toolchain.



Ship

Ship the “Dockerized” app and dependencies anywhere - to QA, teammates, or the cloud - without breaking anything.

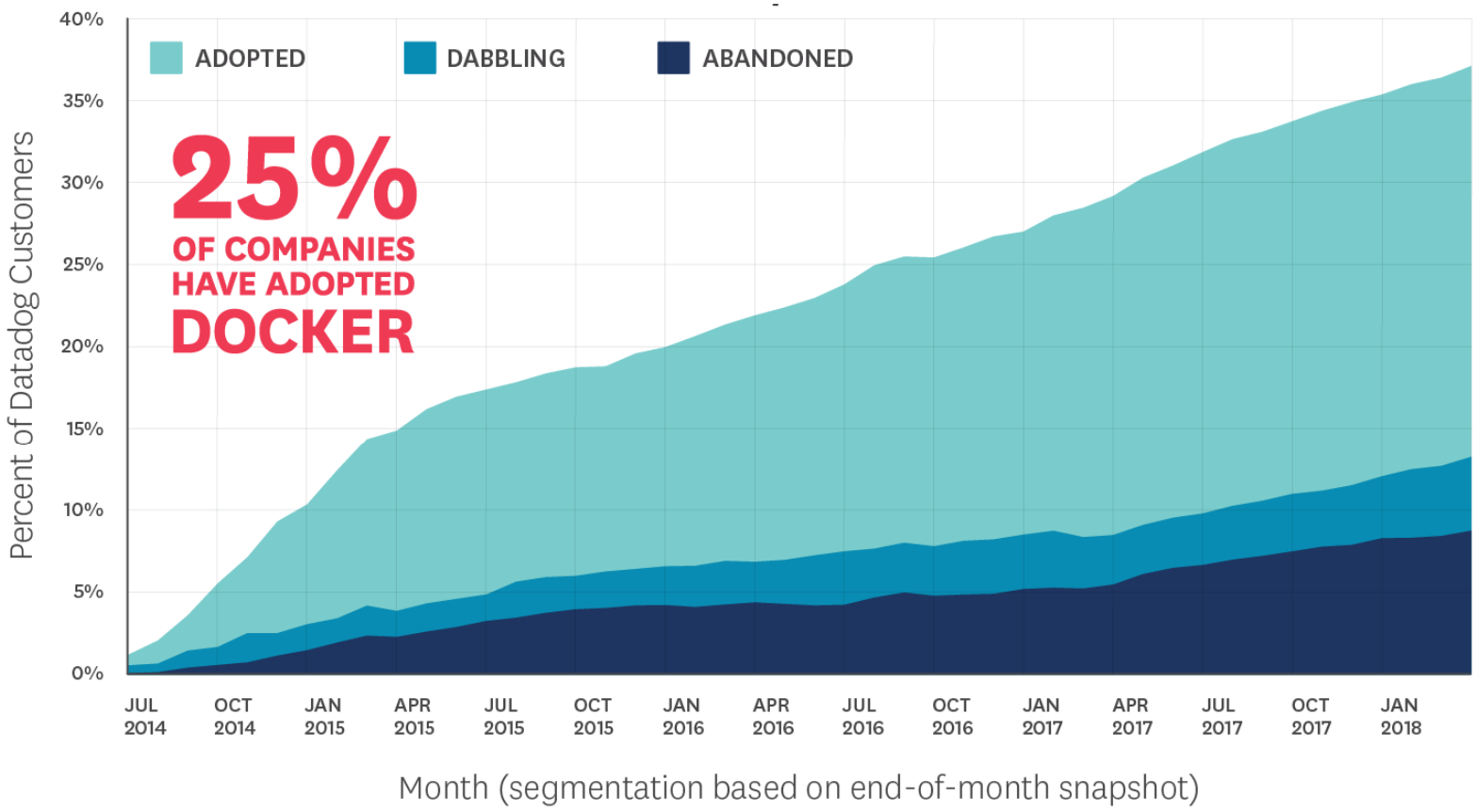
Run

Scale to 1000s of nodes, move between data centers and clouds, update with zero downtime and more.

Docker Facts

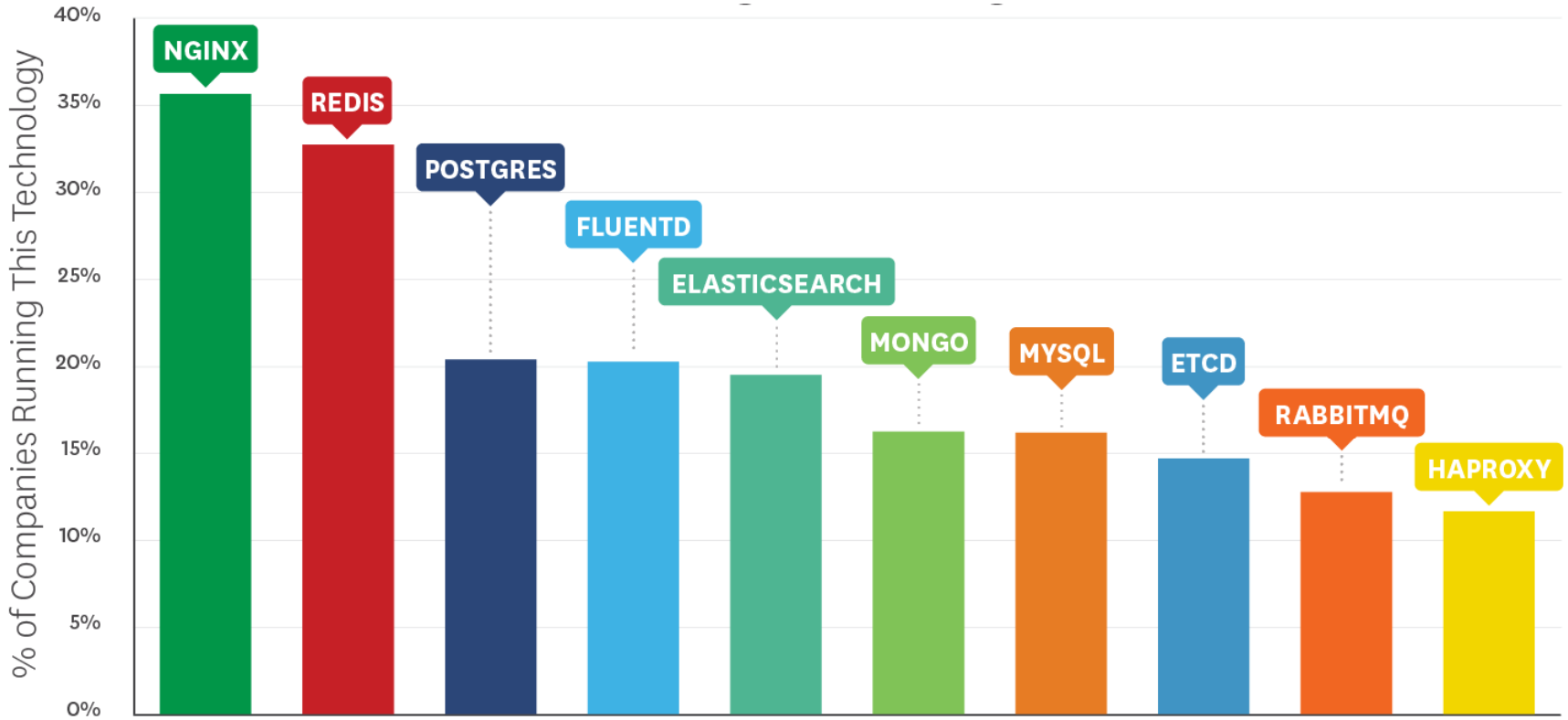
Original author(s)	Solomon Hykes
Developer(s)	Docker, Inc.
Initial release	March 13, 2013; 5 years ago
Stable release	18.06.1-ce[1] / August 22, 2018; 42 days ago
Repository	github.com/docker/docker-ce
Written in	Go[2]
Operating system	Linux,[a] Windows, macOS
Platform	x86-64, ARM
Type	Operating-system-level virtualization
License	•**Binaries:** Freemiumsoftware as a service[5] •**Source code:** Apache License 2.0
Website	docker.com

Docker Adoption Behavior



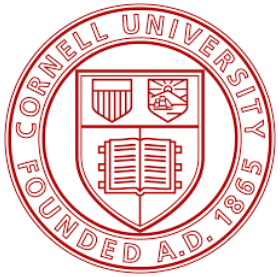
Source: Datadog

Top Technologies Running on Docker



Source: Datadog

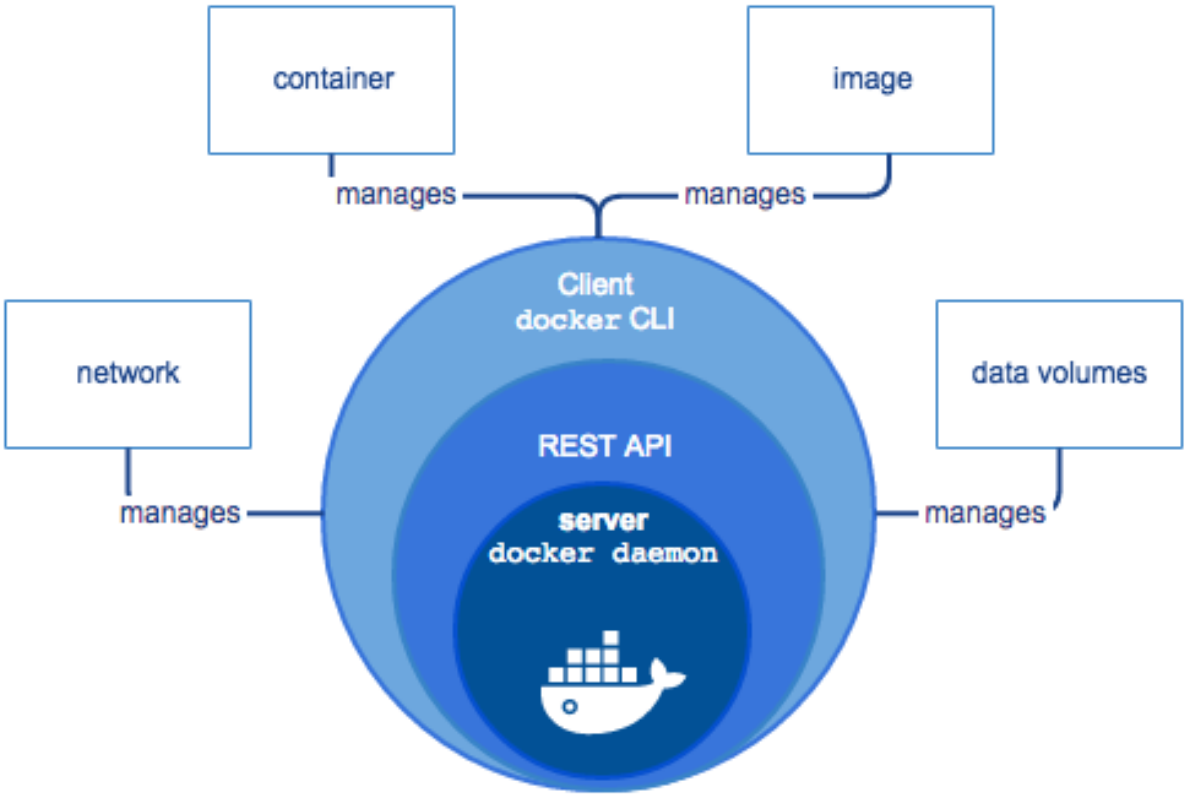
Who is using Docker?



Docker Architecture

Docker Engine is a client-server application with three major components:

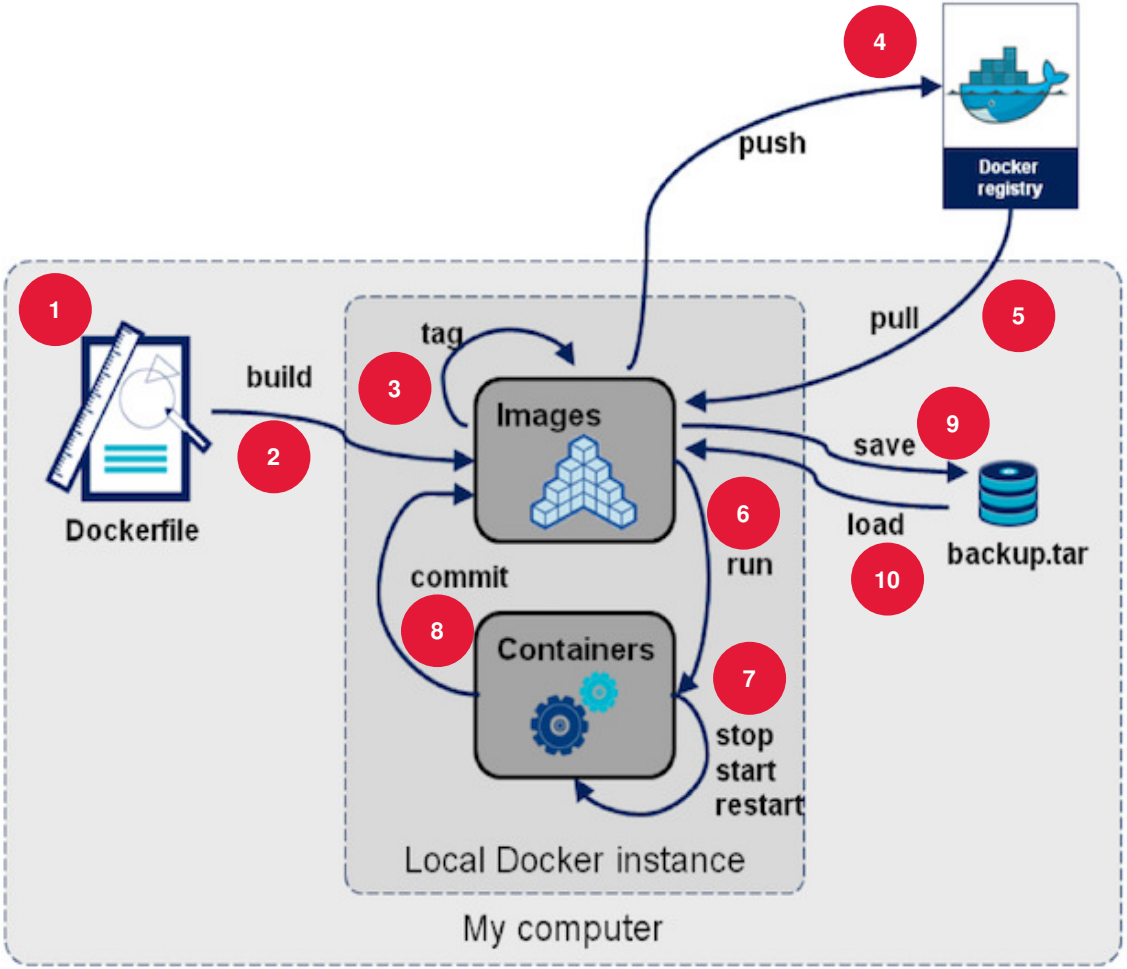
- A server which is a type of long-running program called a daemon process (the **dockerd** command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface (CLI) client (the **docker** command).



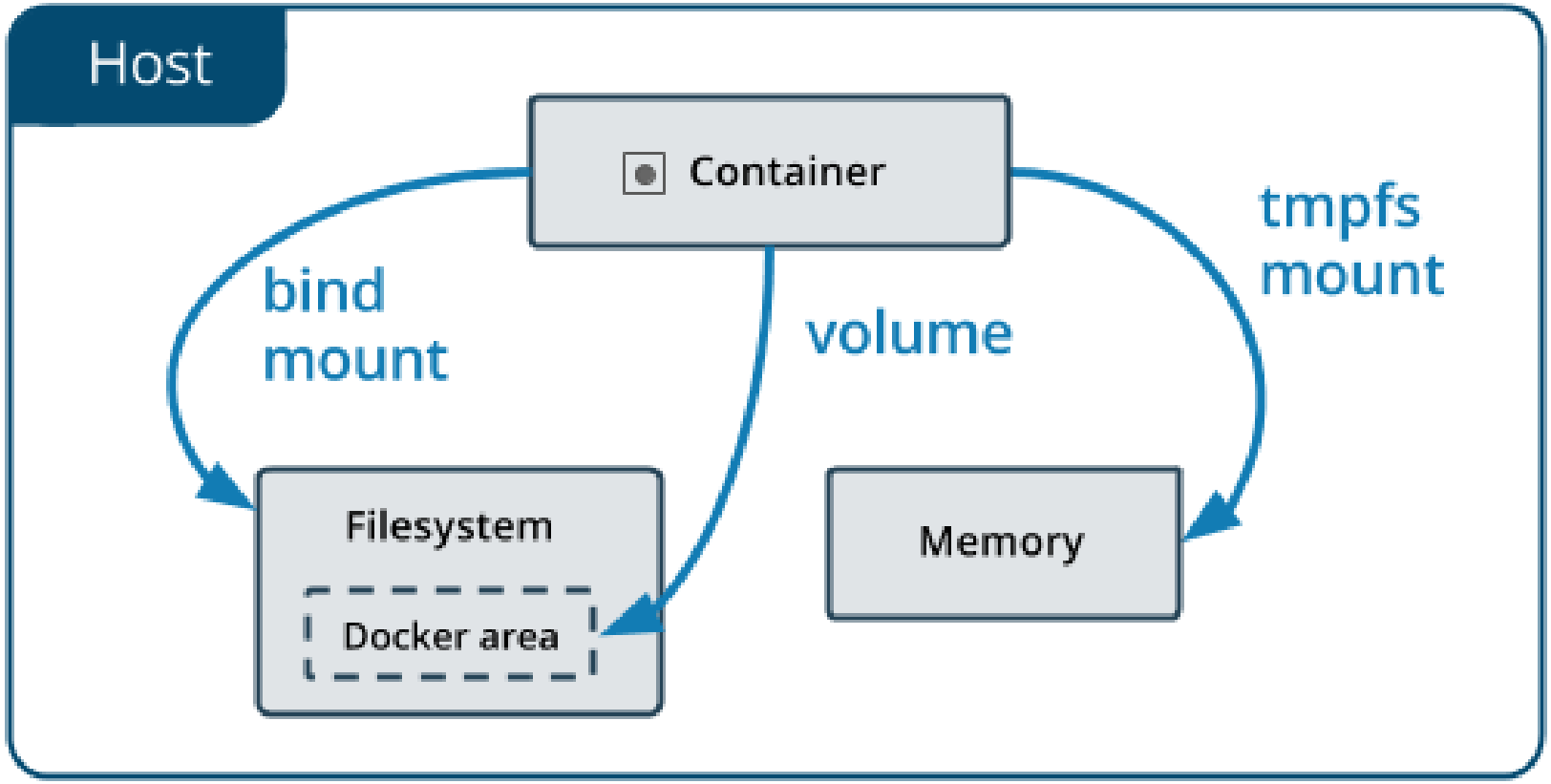
Docker Concepts

Image	A read-only template with instructions for creating a Docker container
Container	A runnable instance of an image.
Volume	Volumes are the preferred mechanism for persisting data generated by and used by Docker containers.
Service	In a distributed application, different pieces of the app are called "Service". A service only runs one image, but it codifies the way that image runs - what ports it should use, how many replicas of the container should run so the service has the capacity it needs, and so on.
Network	One of the reasons Docker containers and services are so powerful is that you can connect them together, or connect them to non-Docker workloads
Stack	A group of interrelated services that share dependencies, and can be orchestrated and scaled together.
Swarm	A group of machines that are running Docker and joined into a cluster.
Registry	The Registry is a stateless, highly scalable server side application that stores and lets you distribute Docker images. The Registry is open-source, under the permissive Apache license.

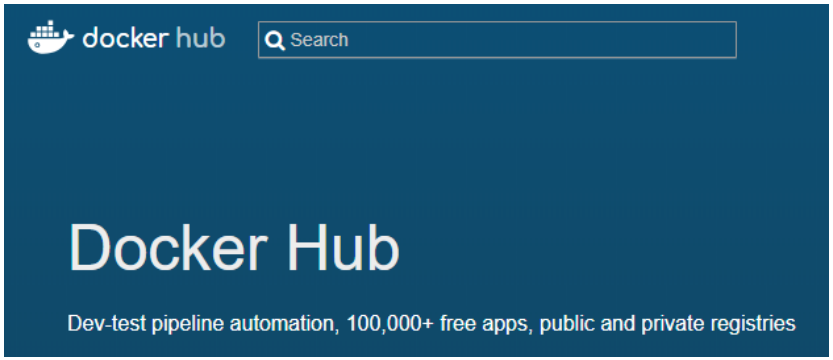
Docker workflow

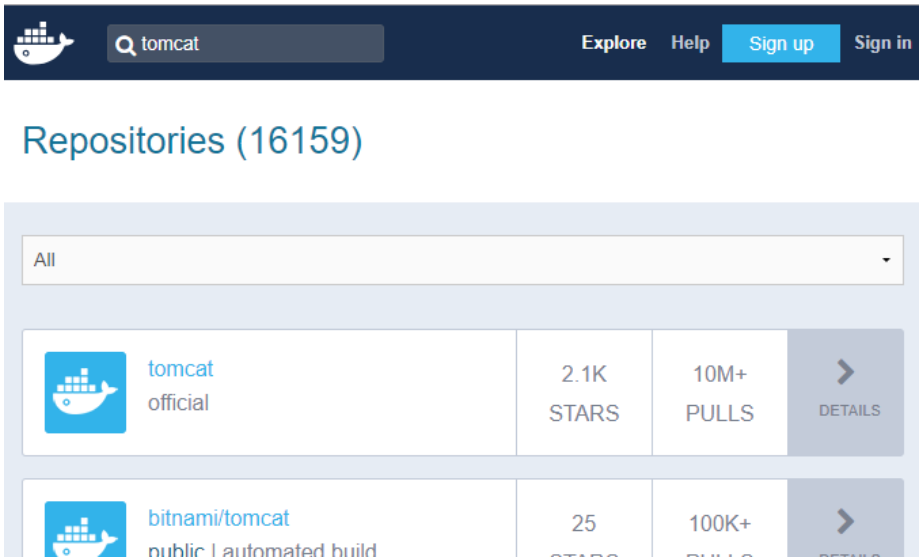


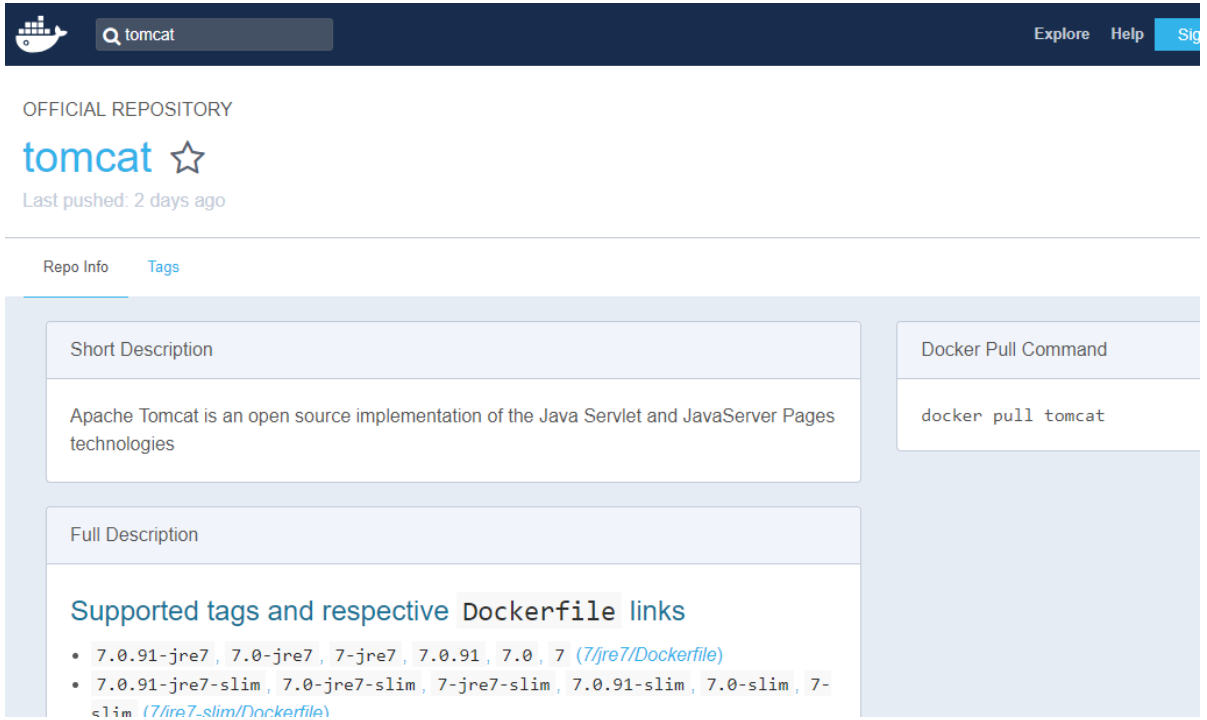
Manage data in Docker



Docker Registry – Docker Hub







Usefull Commands

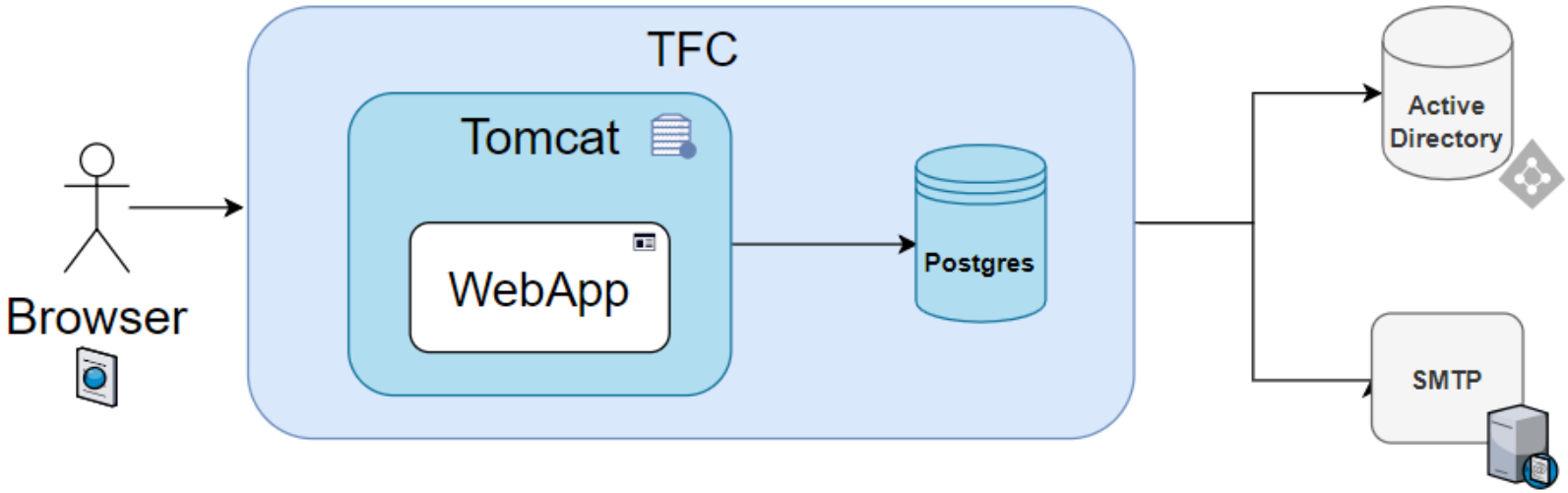
Command	Description
docker pull	Pull an image or a repository from a registry
docker run	Run a command in a new container
docker ps	List containers
docker container	Manage containers
docker images	List images
docker start	Start one or more stopped containers
docker stop	Stop one or more running containers
docker inspect	Return low-level information on Docker objects
docker rm	Remove one or more containers
docker rmi	Remove one or more images
docker logs	Fetch the logs of a container
docker build	Build an image from a Dockerfile
docker tag	Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE

But, it's just the beginning of the journey

https://www.cncf.io/

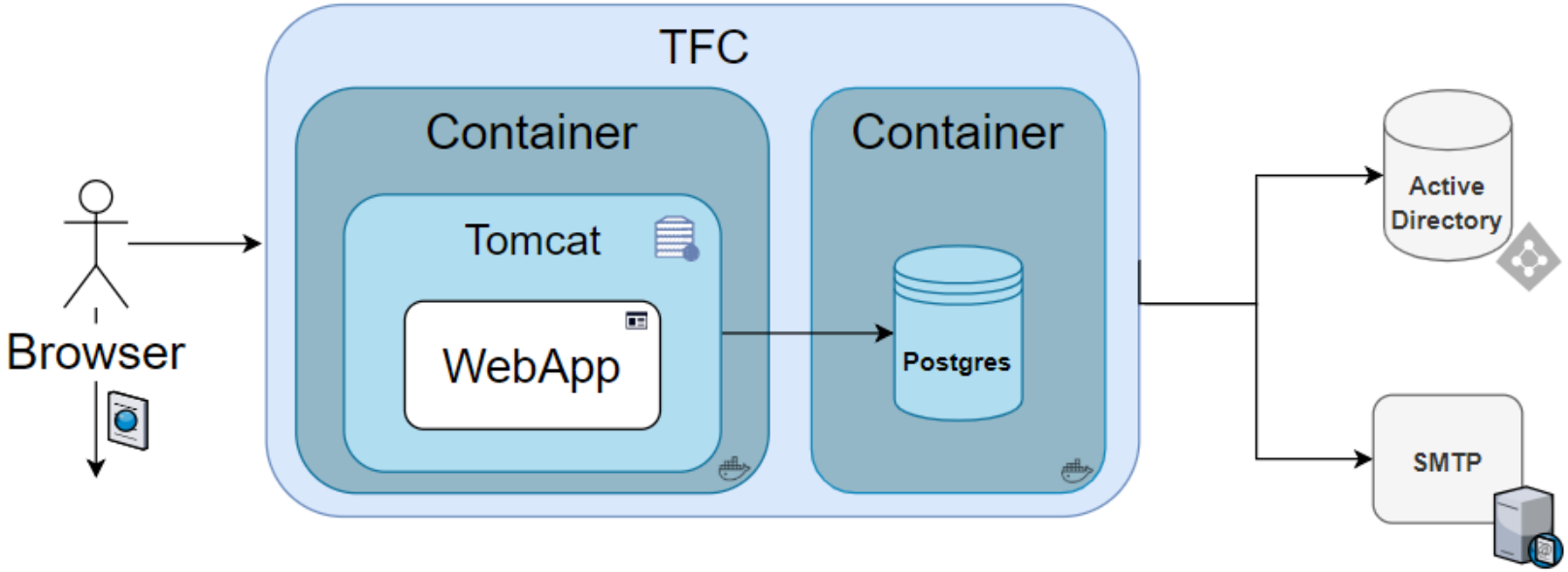
Aplicação de Gestão de TFC‘s

- What is it? – Aplicação de Gestão de Trabalhos de Final de Curso
- How it works

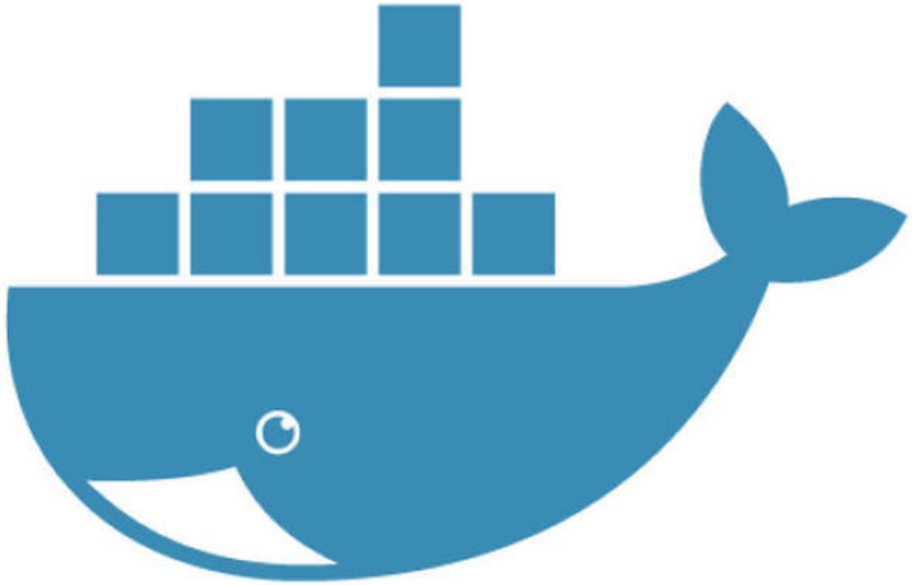


Aplicação de Gestão de TFC's

- Proposed architecture
- Advantages
- Services/Components envisioned



Hands On with Docker



References and …

From Zero to Docker on GitHub (Presentation and Hands-on material)	github.com/mariodagot/from-zero-to-docker

Docker Get Started	https://docs.docker.com/get-started/
Docker Registry	https://docs.docker.com/registry/
Docker file reference	https://docs.docker.com/engine/reference/builder/
Docker Compose	https://docs.docker.com/compose/overview/
Docker compose file reference	https://docs.docker.com/compose/compose-file/
Docker Swarm	https://docs.docker.com/engine/swarm/
Docker stack file reference	https://docs.docker.com/docker-cloud/apps/stack-yaml-reference/
Container Technologies Overview	https://dzone.com/articles/container-technologies-overview
	https://en.wikipedia.org/wiki/Hype_cycle https://www.slideshare.net/spnewman/confusion-in-the-land-of-the-serverless
Docker adoption	https://www.datadoghq.com/docker-adoption/