moku

www.moku.io

Il motore

Fabio Ros

In questa sezione

- Evoluzione del Docker Engine
- Componenti principali

Disclaimer

Sezione di approfondimento.

Puoi usare Docker anche senza conoscere nel dettaglio gli aspetti di questa sezione.

Docker Engine

Docker Engine

Software principale per il controllo di esecuzione e gestione dei container

Docker Engine

- Composto da componenti specializzate ed intercambiabili
- Reso il più aderente possibile agli standard definiti da OCI

Docker Engine

Componenti principali:

- Docker client
- Docker daemon
- containerd
- runc

Docker Engine

Prima c’era un demone monolitico: “**The Docker Daemon**”.

“**LXC**” gli forniva accesso alle principali primitive relative ai container presenti nel kernel linux.

Docker Engine

Problema con LXC:

- Dipendenza da LXC per funzionalità core

Soluzione:

- **libcontainer** (Docker 0.9+)
  - proprietario
  - indipendente dalla piattaforma in cui gira - cross platform

Docker Engine

Problema con Docker deamon:

- Monolitico => difficile da mantenere, correggere ed estendere

Soluzione:

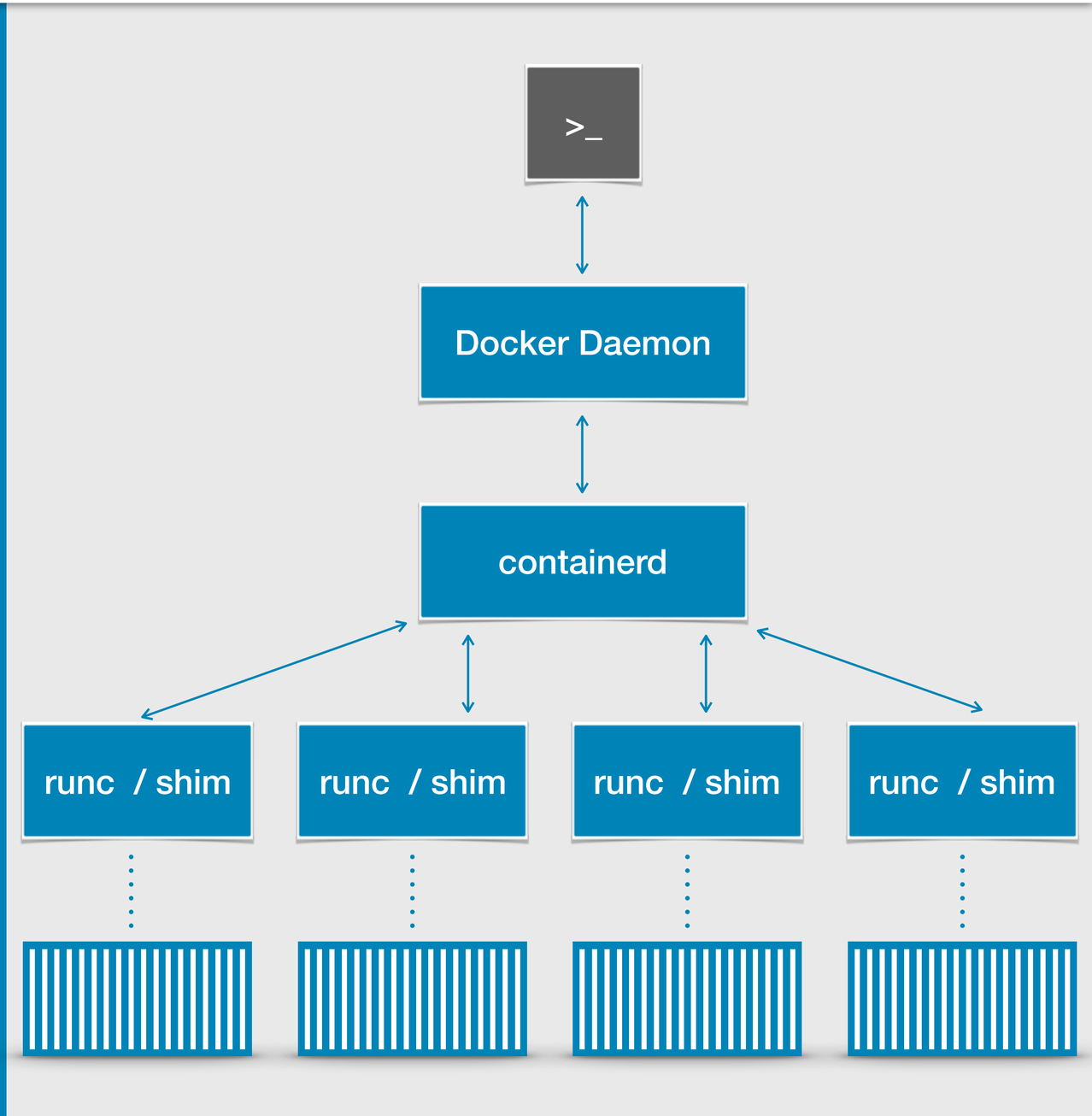
- Scomporlo in moduli più piccoli specializzati ed intercambiabili
- Attività ancora in corso
- Container runtime ed esecuzione dei container sono stati rimossi dal demone e gestiti dai moduli **runc** e **containerd**

Componenti principali

Componenti principali

Architettura ad alto livello del motore di Docker

(Docker Engine)



Componenti Principali

runc

Componenti Principali > runc

- Ha il compito di creare container.
- Standalone
- Implementato sulla base delle specifiche OCI per i container-runtime
- Interfaccia per libcontainer

Componenti Principali

containerd

Componenti Principali > containerd

- Gestisce la logica di esecuzione dei container
- Si occupa del ciclo di vita dei container nei vari aspetti, ma non della loro creazione, per quello c’è runc.
- Gestisce routine del tipo: start, stop, rm, ..
- Oltre ai container gestisce anche alcune funzionalità inerenti le immagini
- Sviluppato da Docker Inc. e donato a Cloud Native Computing Foundation (CNCF).

Componenti Principali

shim

Componenti Principali > shim

Una volta creato un container, il processo del runc termina.

A questo punto il processo shim diventa il padre del container ed assume responsabilità quali:

- Gestione degli stream di I/O standard (STDIN e STDOUT)
- Notifica degli eventi d’interesse (es terminazione) al demone