

-> Docker-Machine

- > strumento per il provisioning e la gestione delle tue macchine virtuali predisposte per Docker
- propria interfaccia (CLI) "docker-machine"
- gestione delle proprie Macchine Virtuali

-> Docker

- > progetto open source che automatizza la distribuzione di applicazioni all'interno di container

-> Container

- > Imballaggio standardizzato per software e dipendenze , Applicazione isolata , Leggero

- condividono lo stesso Kernel
- non bisogna di librerie

vs Image

- template read-only , posso costruirle su altre immagini

ha un sottile strato scrivibile collegato all'immagine

-> Copy-on-write strategy

- quando avvio un container una piccola scrivibile viene aggiunta sopra agli altri livelli.

- > se un file viene modificato durante la creazione dell'immagine , un nuovo livello viene creato

⚠ Posso creare più container partendo dalla stessa immagine!

- > Lista delle immagini -> docker image ls -a
  - > Lista dei container -> docker ps -a
    - ↓
      - o
      - docker container ls -a
      - s / --size
- visualizza il totale dimensione dei dischi (container)

### -> Limitazione memoria

- cpus = <value>
  - cpu-period = <value>
  - cpu-quota = <value>
- 
- docker run -m 256m yourapp
  - o
  - memory-reservation

-> Docker mette a disposizione diverse immagini per snellire il processo di compilazione in diversi linguaggi

```
docker run -v "$(pwd)"/es-micro:/app -w /app
  --rm maven:3.5-jdk-8 mvn clean install
  -Dmaven.test.skip=true && java -jar
  ./app/target/micro-0.0.1-SNAPSHOT.jar
```

# Lezioni

## → Cloud computing

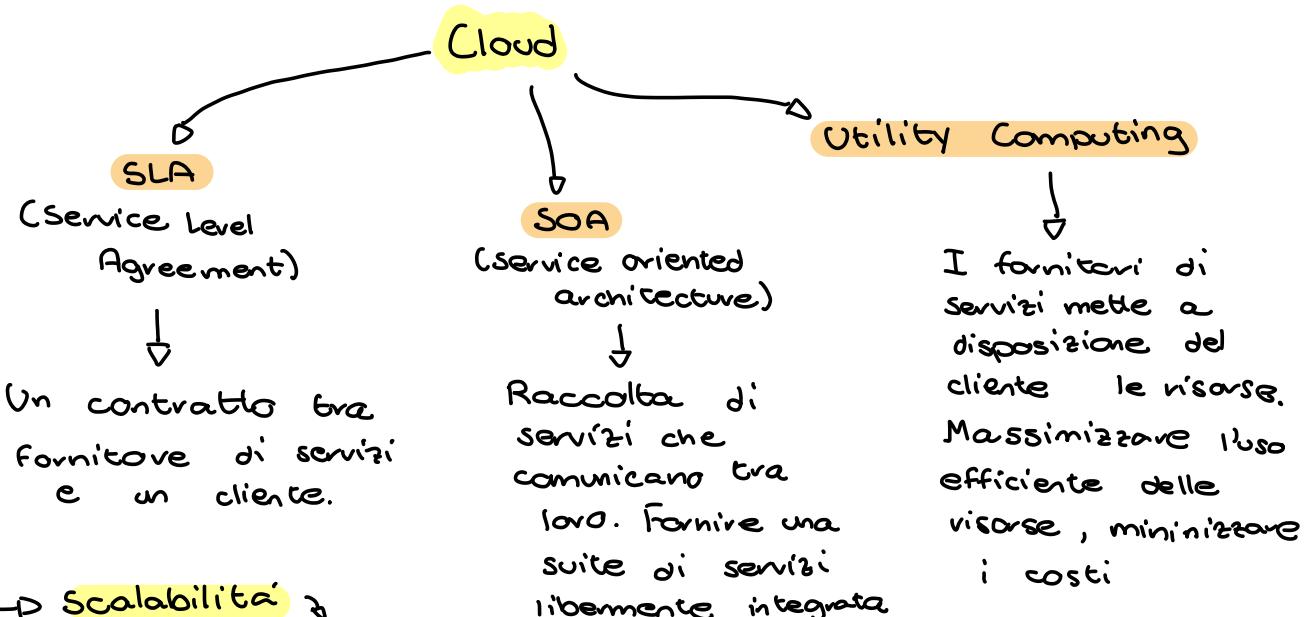
è uno stile di computing con dinamiche, scalabili e a volte virtualizzate risorse

è informatica basata su Internet, in cui risorse condivise vengono fornite su richiesta.

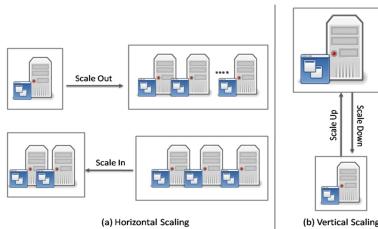
caratteristiche:

- self-service su richiesta
- ampio accesso alla rete
- pool di risorse
- rapida elasticità
- servizi misurati

**Parole Chiave:** distributed system, virtualized pc, dynamically provisioned , on service-level agreements

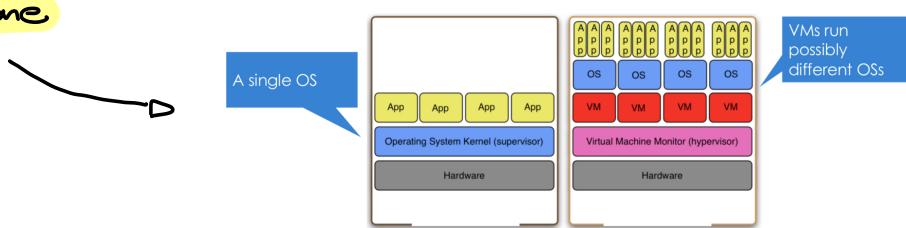


## → Scalabilità



(service request, service response)

→ **Virtualizzazione**

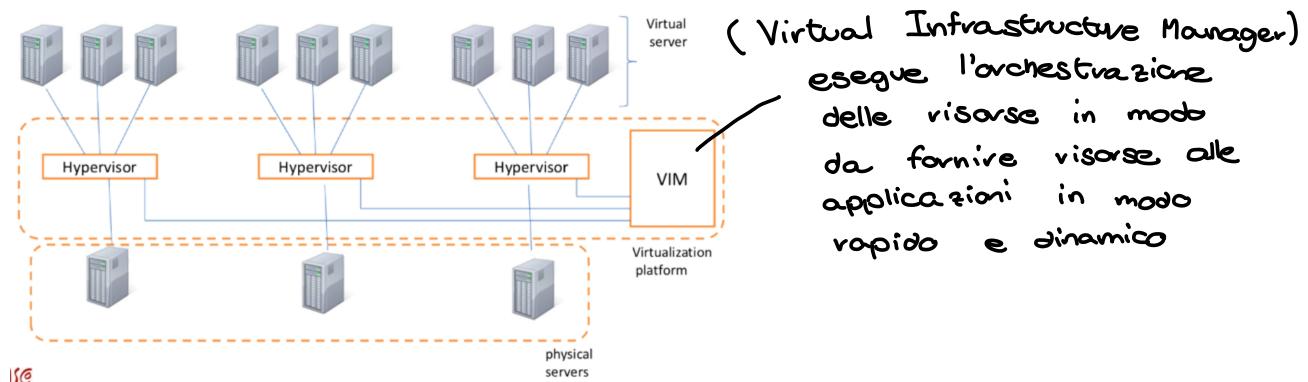


→ **Multi-tenant** → si riferisce a un'architettura dove una singola istanza del SW viene eseguita su un server.

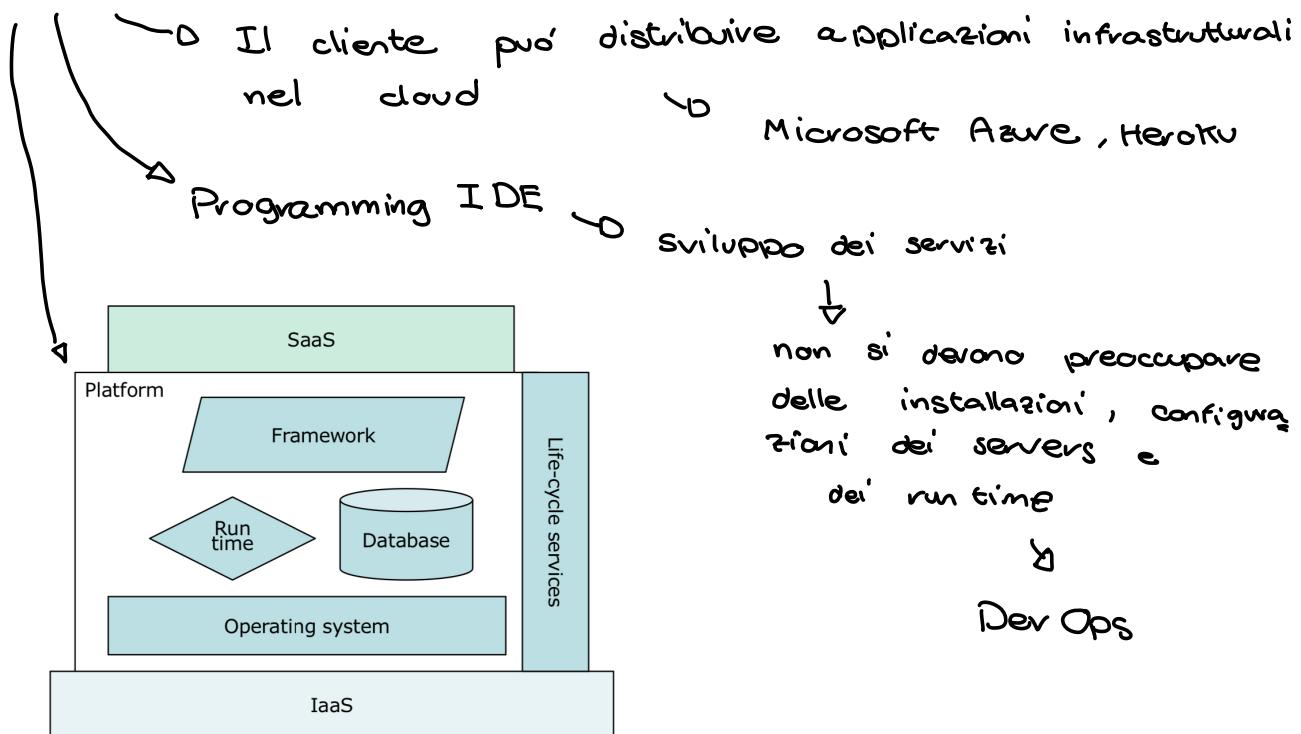
↓  
un'app software  
è progettata per  
partizionare virtualmente  
i suoi dati

**IaaS - Infrastructure as a Service**

Il cliente ha accesso alle risorse fondamentali e può sviluppare, deploy e run del software.  
ma non può controllare e modificare l'infrastruttura cloud.  
↓  
VM, Virtual Storage,  
Virtual Network



## PaaS → Platform as a Service



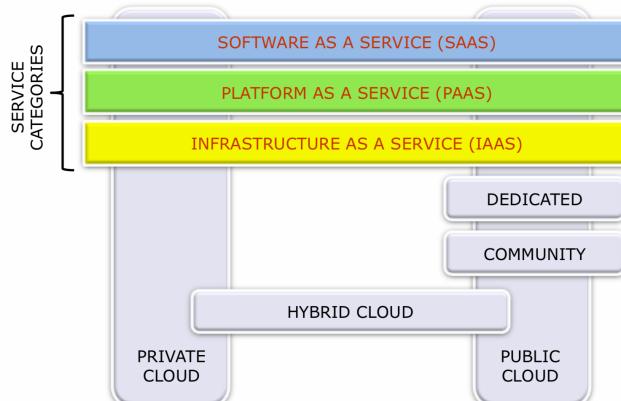
## SaaS → Software as a Service

→ Il cliente può utilizzare le applicazioni fornite dal provider. Le applicazioni sono accessibili dai vari dispositivi. (e-mail Web)

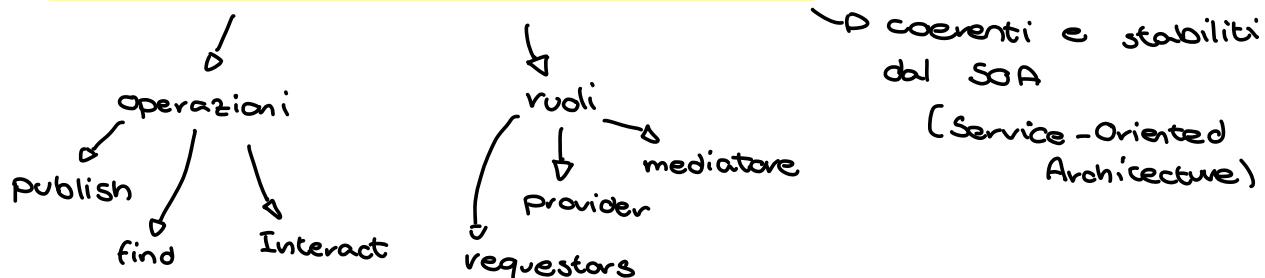
application finita

→ Deploy

app →



## -D Architettura dei Servizi Web basici



## -D REST → Representational State Transfer

stile architettonicale per sistemi distribuiti  
Risorse sono definite da URIs, e posso manipolare la loro rappresentazione. Messaggi autodescritti e senza stato.

URI & HTTP

-D Ogni richiesta deve avere tutte le informazioni necessarie per eseguire una richiesta. (Non ho memoria sul server)

-D REST consente la memorizzazione nella cache e il riutilizzo delle interazioni

## -D Microservizi

autonomi,  
piccoli e concentrati  
sul far bene  
una cosa.  
più facile manutenzione  
↓  
+ velocità consegna

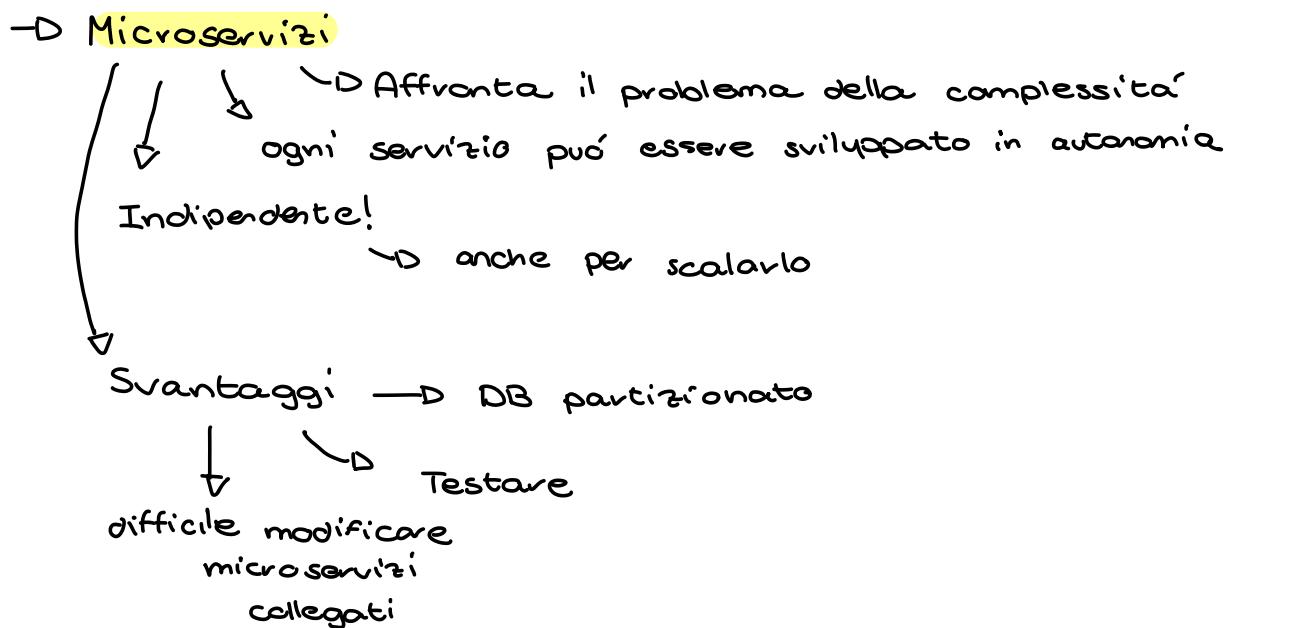
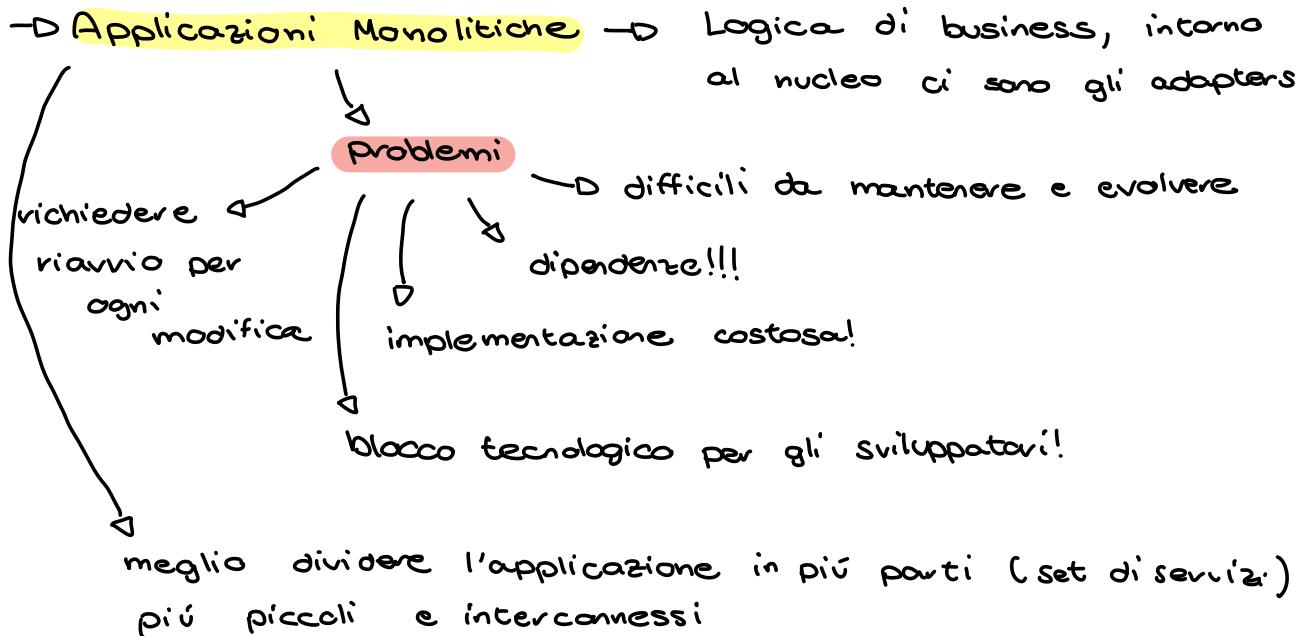
→ REST → Il web si basa su client e server per le risorse

HTTP è il protocollo utilizzato  
(metodi + controllo + media type)

→ vs Services

→ monolithic, grosse tutto in uno

- comunicazione



→ Distribuzione → è molto complessa

Un app di microservizi  
è tipicamente costituita  
da un grande numero di servizi.

→ un app monolitica viene semplicemente distribuita con bilanciamento

→ Un pattern è una soluzione riutilizzabile a un problema  
che si verifica in un contesto