UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Tecnologie Cloud & Mobile

Tecnologie Cloud e Mobile

Cloud Systems and Infrastructure

mauro.pelucchi@gmail.com
Mauro Pelucchi

21059 - Tecnologie cloud e mobile

Cosa vedremo?

- Cloud computing
- AWS
- Micro-services
- Server-less architecture

Cloud computing

Computing paradigms

Mainframe computing
1 computer / molti utenti

"The network is the computer"
(John Gage, Sun Microsystems, 1984)

Client-server computing
molti computer / molti utenti

"The cloud is the computer"

Cloud computing

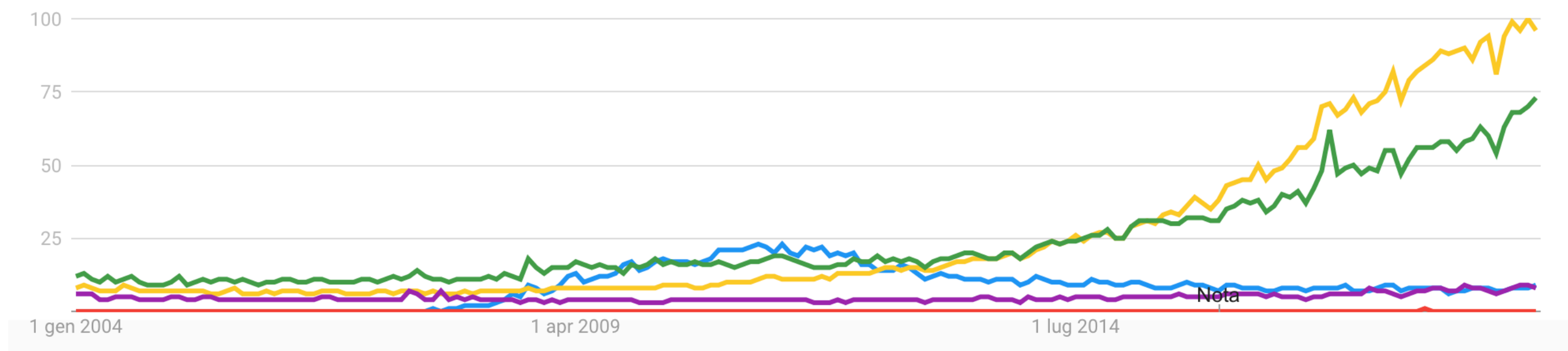
cloud computing
Termine di ricerca

big data cloud
Termine di ricerca

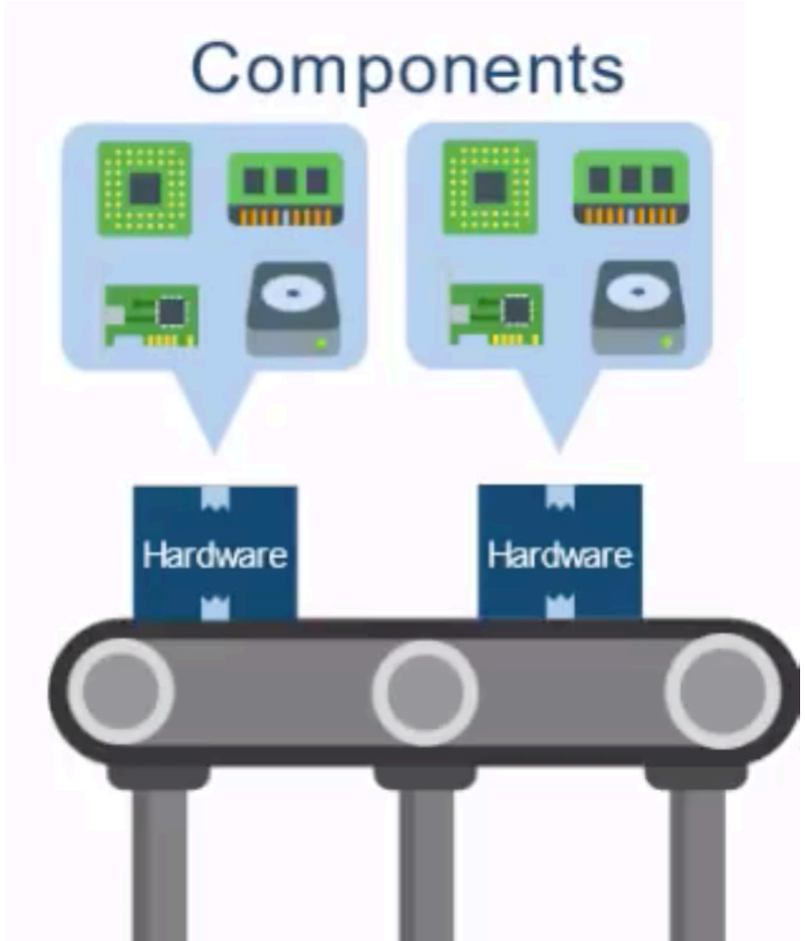
aws
Termine di ricerca

azure
Termine di ricerca

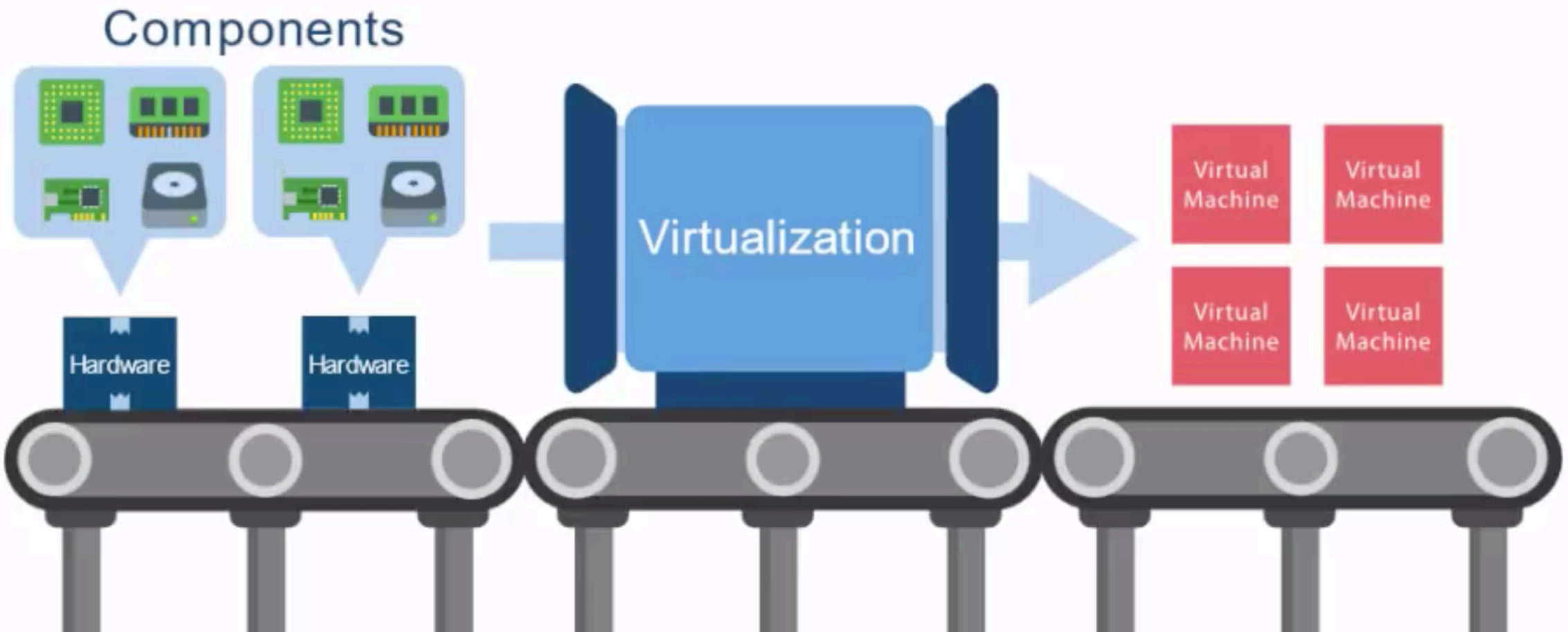
gpc
Termine di ricerca



Primo passo…
la virtualizzazione….



Primo passo…
la virtualizzazione….

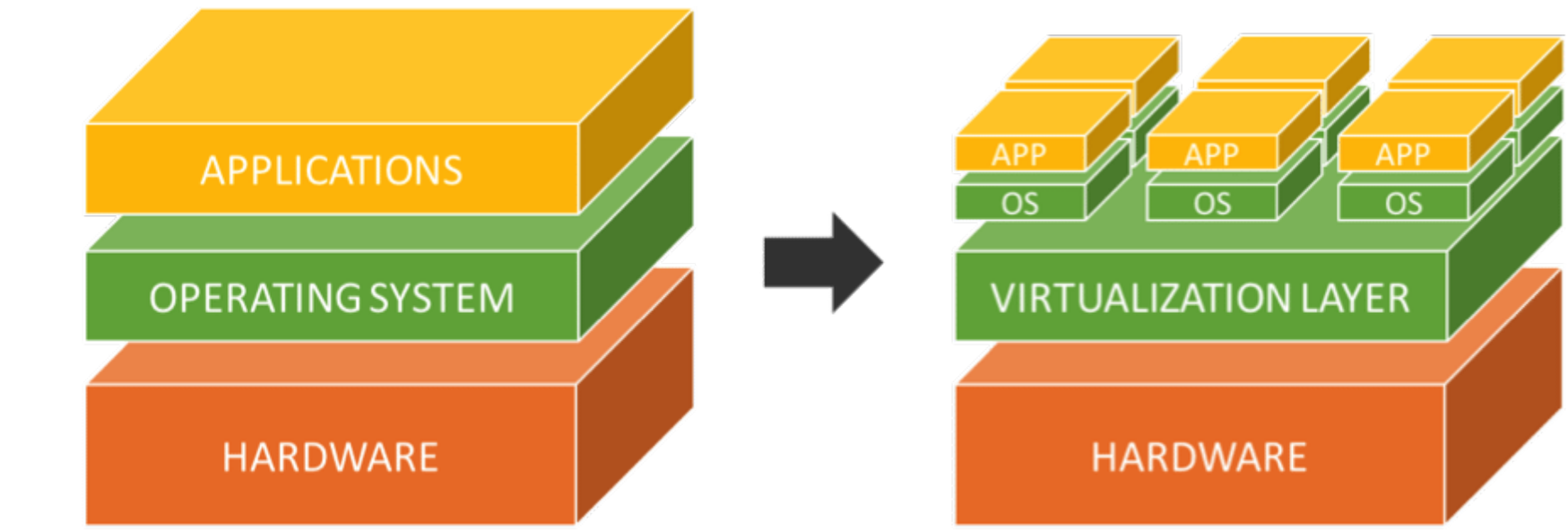


Virtualizzazione

Attraverso la virtualizzazione vado a scomporre un grosso server fisico in tanti server virtuali.
Ciascun server virtuale è dedicato ad un componente del mio sistema.

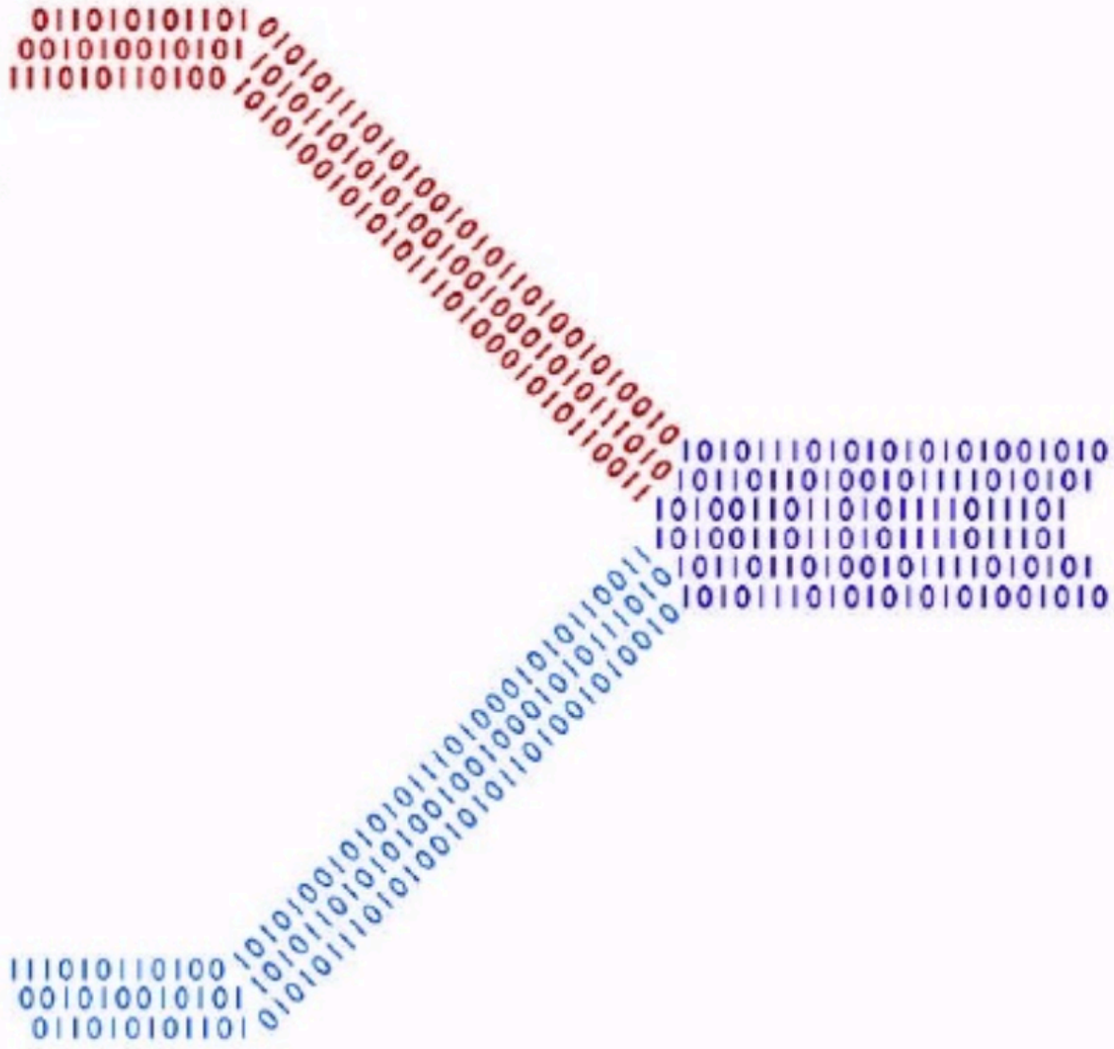
In altre parole, vado ad astrarre un servizio IT.

Poche grosse macchine fisiche ospitano tante piccole macchine virtuali indipendenti (diverso sistema operativo, diverse applicazioni, ….).



A Business Model

- Pay as you go
- Virtualized resources
- On-demand
- Multi-tenancy
- Scalability, etc.



An Architectural Platform and a Business Model

A Delivery Platform

Enable users to access IP services and on-demand applications

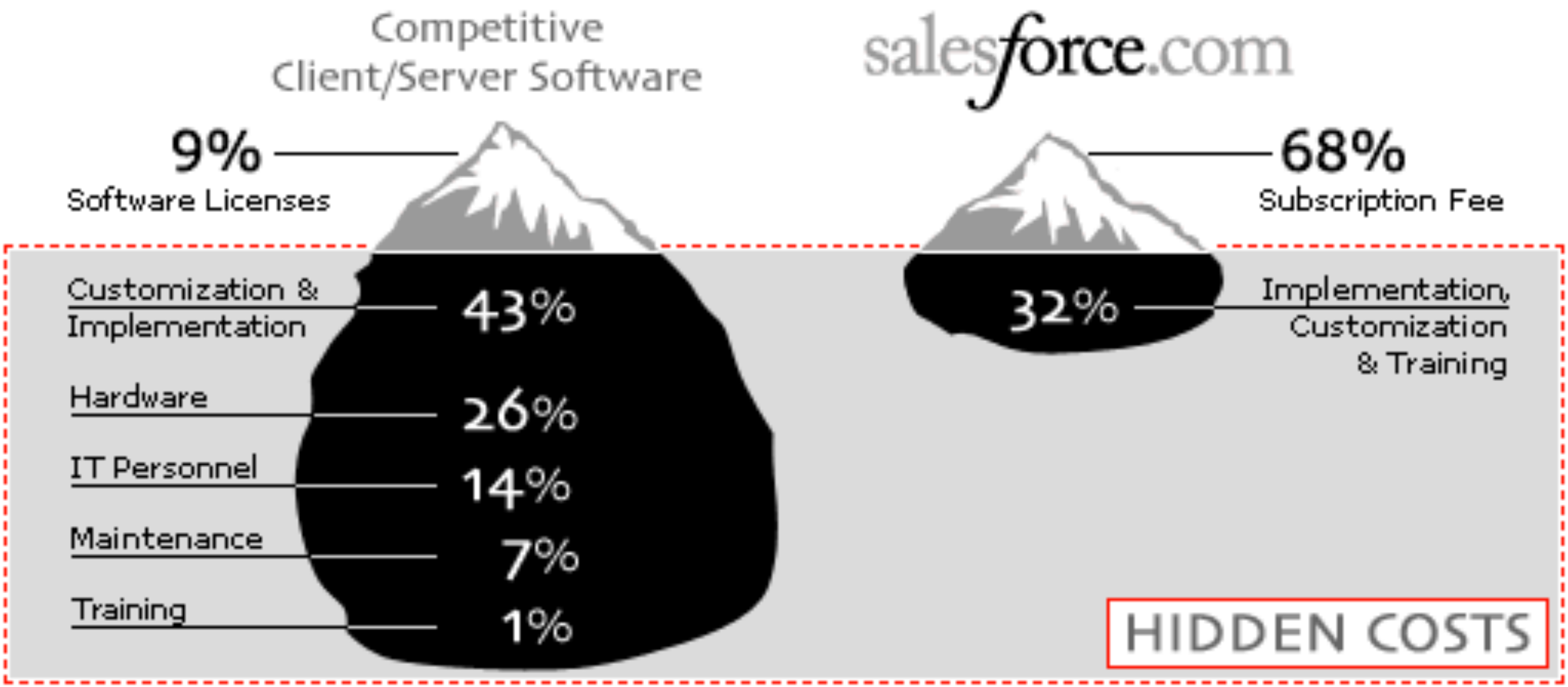
Yankee Group, 2010

A style of computing in which dynamically scalable and often virtualized resources are provided as a service over the Internet.

[Wikipedia]

Perché?

Avoid the hidden costs of traditional CRM software



Cloud computing

Il cloud computing fornisce tramite **WEB** risorse di elaborazione che spaziano dei server e lo storage fino ad intere applicazioni di livello enterprise (email, security, backup ecc.)

Il Cloud è in grado di effettuare provisioning di risorse **IT 'as a service'.**

5 Caratteristiche fondamentali

- Servizi su richiesta
- Accesso di rete
- Pooling di risorse
- Elasticità rapida
- Misura dei servizi

Benefici e sfide cloud computing

Pay per Use	Risorse infinite	Sicurezza (Compliance)	Service Assurance
Scalabilità	Servizi gestiti	Trasparenza	Integrazione con IT
Complessità gestita	Innovazione continua	Portabilità	Licenze
Riduce il time to market	Updated	Confronto dei costi	

Cloud computing
Vantaggi

Increase
Productivity
of my business
processes

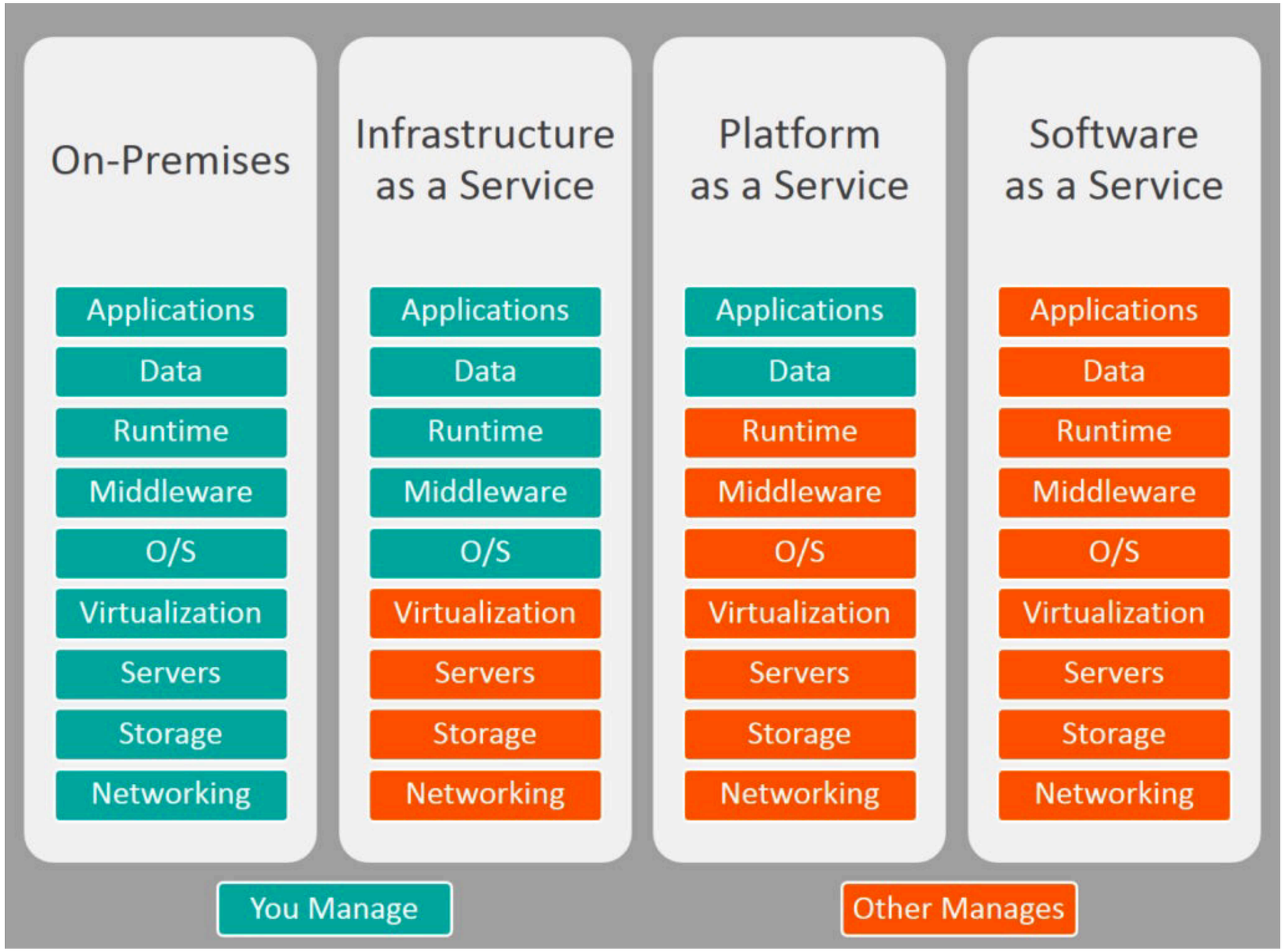
Lower Cost
Both CAPEX
and OPEX

No Risk
Keep all
my assets safe

No Pain
KISS principle
(Keep it simple
stupid)

Yankee Group, 2010

Modello di servizio



Modelli di servizio

SaaS

Software as a Service

Il cliente utilizza applicazioni su una infrastruttura accessibile da vari dispositivi client attraverso un'interfaccia (API, interfaccia web, client dedicato)

PaaS

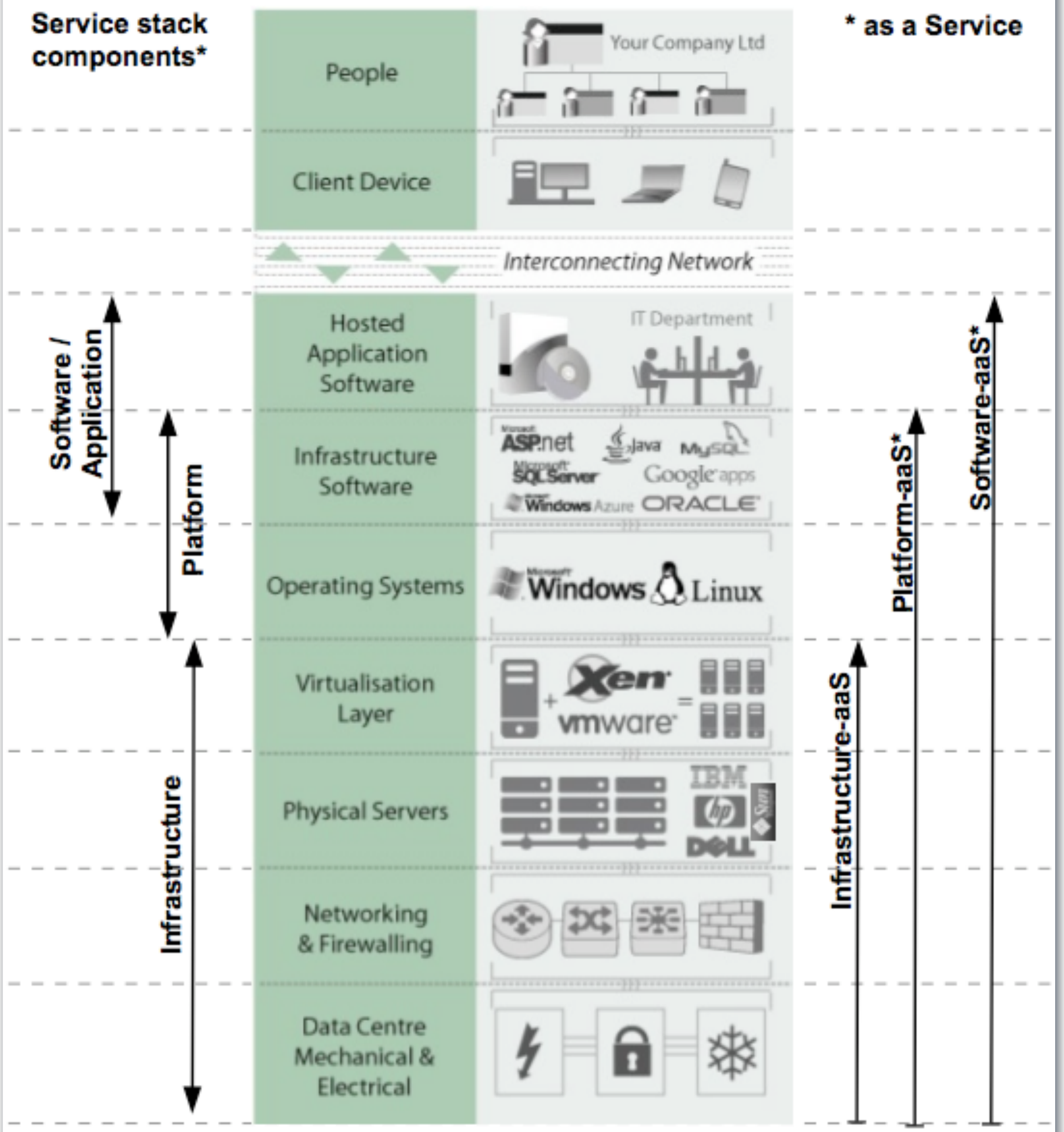
Platform as a Service

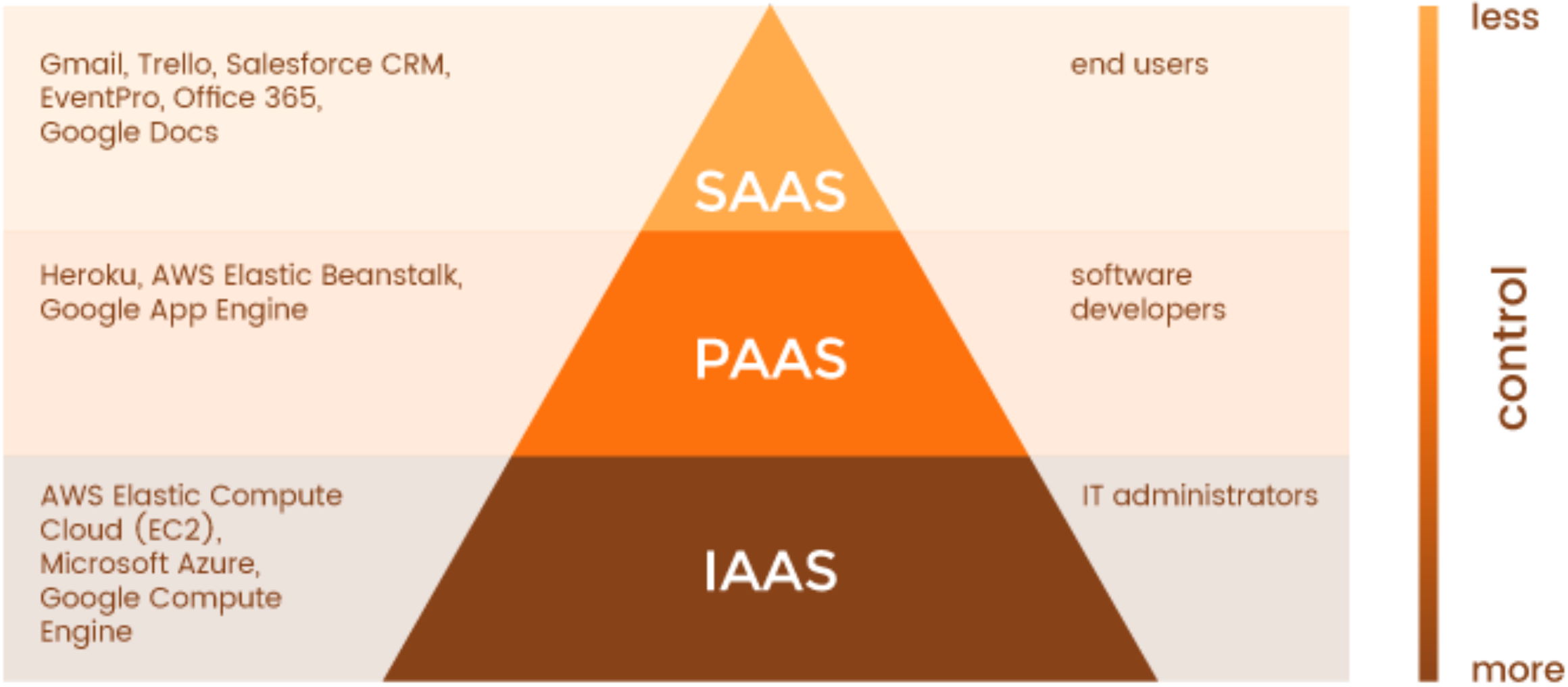
Permette di sviluppare e distribuire applicazioni create utilizzando linguaggi di programmazione supportati dal fornitore

IaaS

Infrastructure as a Service

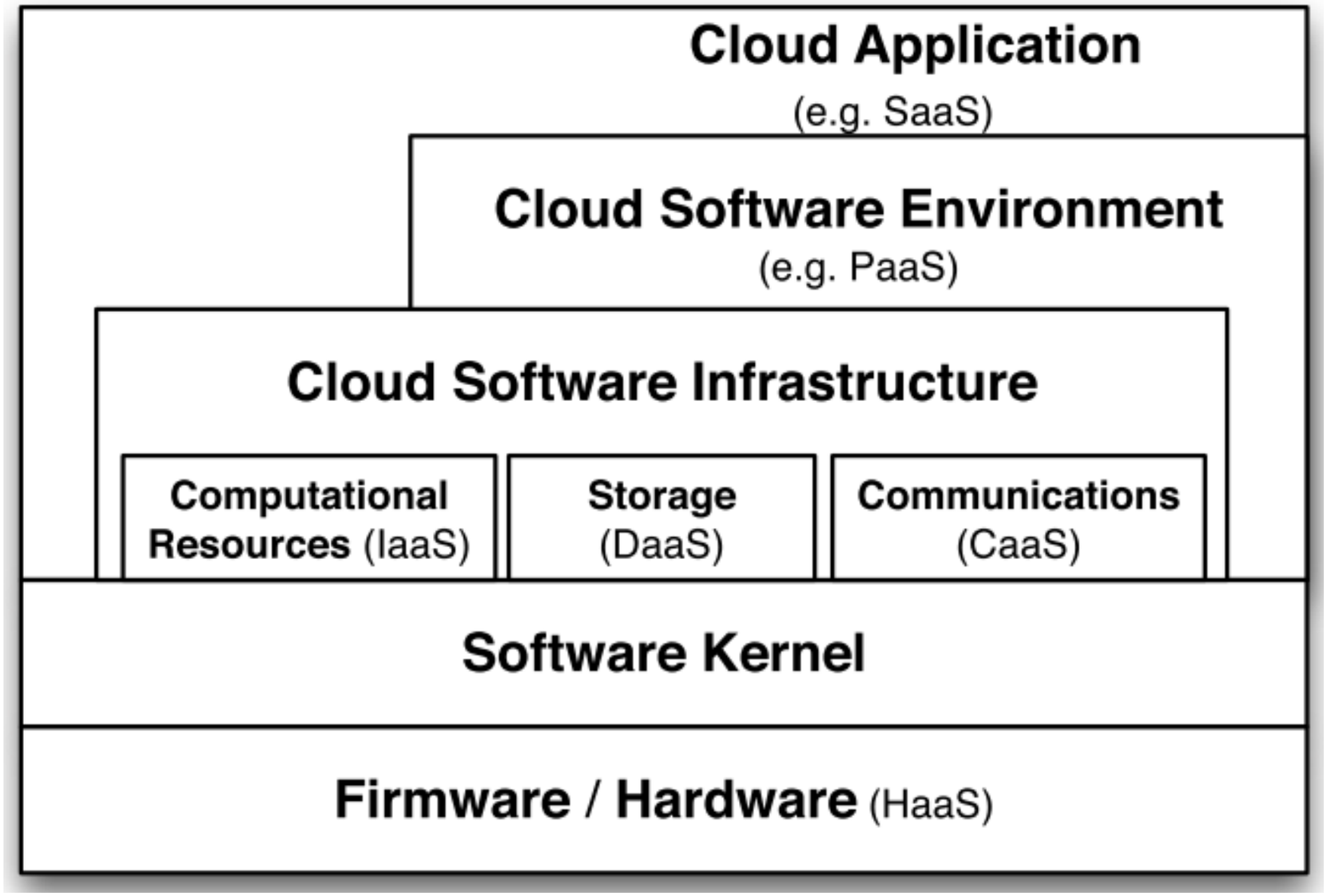
Noleggio capacità di CPU, storage, network e altre risorse come i sistemi operativi e le applicazioni


Service stack components*
* as a Service
People
Your Company Ltd
Client Device
Interconnecting Network
Hosted Application Software
IT Department
Infrastructure Software
ASP.net
Java
MySQL
SQLServer
Google apps
Windows Azure
ORACLE
Operating Systems
Windows
Linux
Virtualisation Layer
Xen
vmware
Physical Servers
IBM
HP
SUN
DELL
Networking & Firewalling
Data Centre Mechanical & Electrical
Software / Application
Platform
Infrastructure
Software-aaS*
Platform-aaS*
Infrastructure-aaS


Gmail, Trello, Salesforce CRM, EventPro, Office 365, Google Docs
end users
SAAS
Heroku, AWS Elastic Beanstalk, Google App Engine
software developers
PAAS
AWS Elastic Compute Cloud (EC2), Microsoft Azure, Google Compute Engine
IT administrators
IAAS
less
control
more

AaaS	Architecture as a Service
BaaS	Business as a Service
CaaS	Computing as a Service
CRMaaS	CRM as a Service
DaaS	Data as a Service
DBaaS	Database as a Service
EaaS	Ethernet as a Service
FaaS	Frameworks as a Service
GaaS	Globalization or Governance as a Service
HaaS	Hardware as a Service
IaaS	Infrastructure or Integration as a Service
IDaaS	Identity as a Service
ITaaS	IT as a Service
LaaS	Lending as a Service
MaaS	Mashups as a Service
OaaS	Organization or Operations as a Service
SaaS	Software as a Service
StaaS	Storage as a Service
PaaS	Platform as a Service
TaaS	Technology or Testing as a Service
VaaS	Voice as a Service

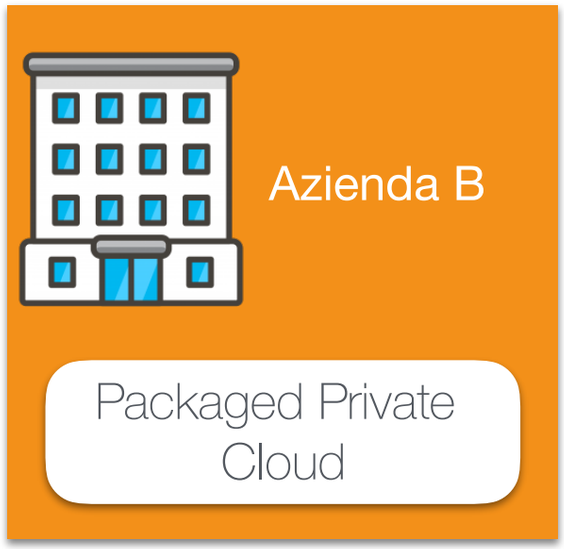
Una vista organica



Modelli di deployment

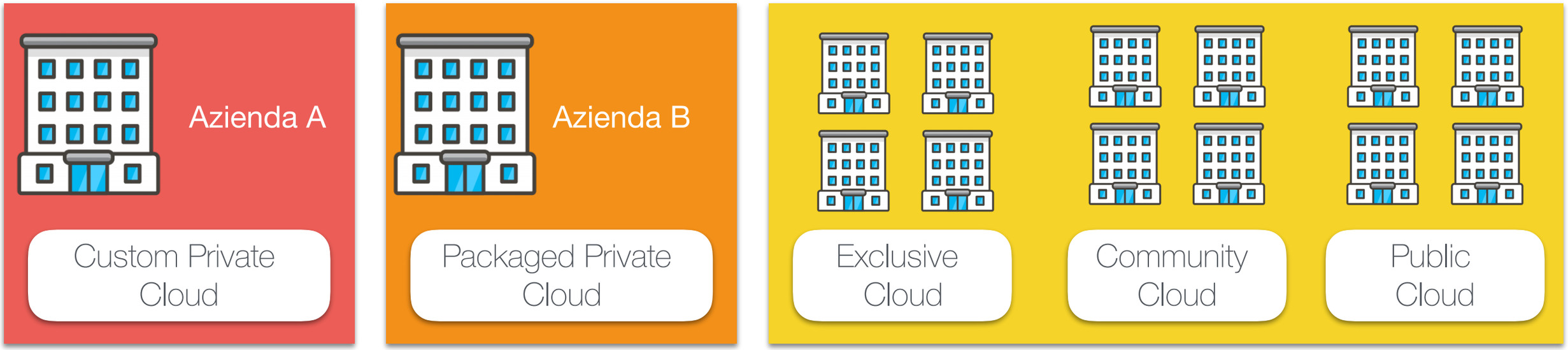
Privato

Pubblico



Privato

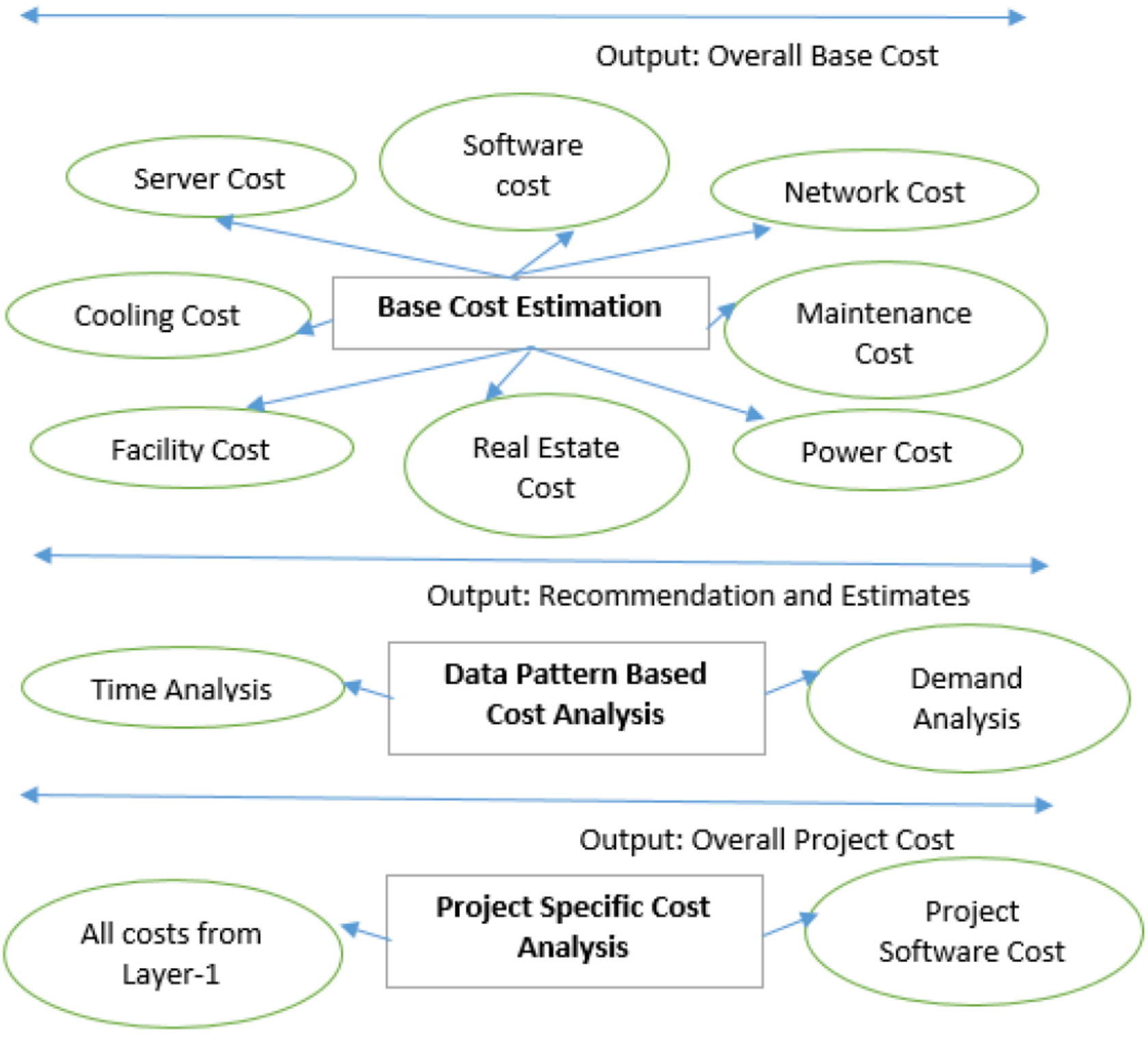
Pubblico



Flessibilità
Sicurezza personalizzata

Costi ridotti
Nessuna manutenzione
Scalabilità quasi illimitata
Affidabilità elevata
Semplicità

Quali sono i costi da tenere in considerazione per un servizio in cloud?



Nanath, Krishnadas, and Radhakrishna Pillai. "A model for cost-benefit analysis of cloud computing." *Journal of International Technology and Information Management* 22.3 (2013): 6.

Cloud pricing model

- Market-driven: variety of pricing models for each of IaaS, PaaS, SaaS service models
- Quanto costa 1 ora di computazione su un server “ready” per i Big Data?
- Quanto costa trasferire 1GB di dati?
- Quanto costa al mese 1GB di spazio occupato?

	vCPU	ECU	Memoria (GiB)	Storage dell'istanza (GB)	Utilizzo di Linux/UNIX
m5.large	2	8	8 GiB	Solo EBS	0,107 USD all'ora
m5.xlarge	4	16	16 GiB	Solo EBS	0,214 USD all'ora
m5.2xlarge	8	31	32 GiB	Solo EBS	0,428 USD all'ora
m5.4xlarge	16	60	64 GiB	Solo EBS	0,856 USD all'ora
m5.12xlarge	48	173	192 GiB	Solo EBS	2,568 USD all'ora
m5.24xlarge	96	345	384 GiB	Solo EBS	5,136 USD all'ora

Google Cloud Platform

Tipo di macchina	CPU virtuali	Memoria	Prezzo (USD)	Prezzo prerilasciabile (USD)
n1-standard-1	1	3.75GB	$0.0612	$0.01230
n1-standard-2	2	7.5GB	$0.1224	$0.02460
n1-standard-4	4	15GB	$0.2448	$0.04920
n1-standard-8	8	30GB	$0.4896	$0.09840
n1-standard-16	16	60GB	$0.9792	$0.19680
n1-standard-32	32	120GB	$1.9584	$0.39360
n1-standard-64	64	240GB	$3.9168	$0.78720

Modello di costo (semplice)

	Ore di utilizzo al giorno	Tipo di macchina	Totale (all'anno)
Database Server (NoSql)	24	m5.2xlarge	3500
ETL (Spark)	10	m5.4xlarge	2900
Presentation	10	m5.4xlarge	2900
			9300

Solo costi EC2!!!

Per risparmiare???

Per risparmiare???

Sfrutto i vantaggi del cloud….

Istanze SPOT

Le istanze **Spot** ti permettono di sfruttare le capacità inutilizzate all'interno del cloud. Le istanze Spot sono disponibili con prezzi **scontati fino al 90%** inferiori rispetto ai prezzi delle istanze on demand.

Quando si utilizzano le Istanze Spot, è necessario essere **preparati alle interruzioni**. Il fornitore può interrompere le Istanze Spot quando il prezzo Spot supera il prezzo massimo, se la domanda di Istanze Spot aumenta o se l'offerta di Istanze Spot diminuisce.

	vCPU	ECU	Memoria (GiB)	Storage dell'istanza (GB)	Utilizzo di Linux/UNIX	Spot
m5.large	2	8	8 GiB	Solo EBS	0,107 USD all'ora	$0.0365 all'ora
m5.xlarge	4	16	16 GiB	Solo EBS	0,214 USD all'ora	$0.0721 all'ora
m5.2xlarge	8	31	32 GiB	Solo EBS	0,428 USD all'ora	$0.1444 all'ora
m5.4xlarge	16	60	64 GiB	Solo EBS	0,856 USD all'ora	$0.2916 all'ora
m5.12xlarge	48	173	192 GiB	Solo EBS	2,568 USD all'ora	$0.8657 all'ora
m5.24xlarge	96	345	384 GiB	Solo EBS	5,136 USD all'ora	$1.7314 all'ora

Istanze riservate

Le **istanze riservate** ti danno la possibilità di effettuare un pagamento unico per ogni istanza che desideri prenotare e ricevere uno sconto significativo sulla tariffa di utilizzo oraria per quell'istanza.

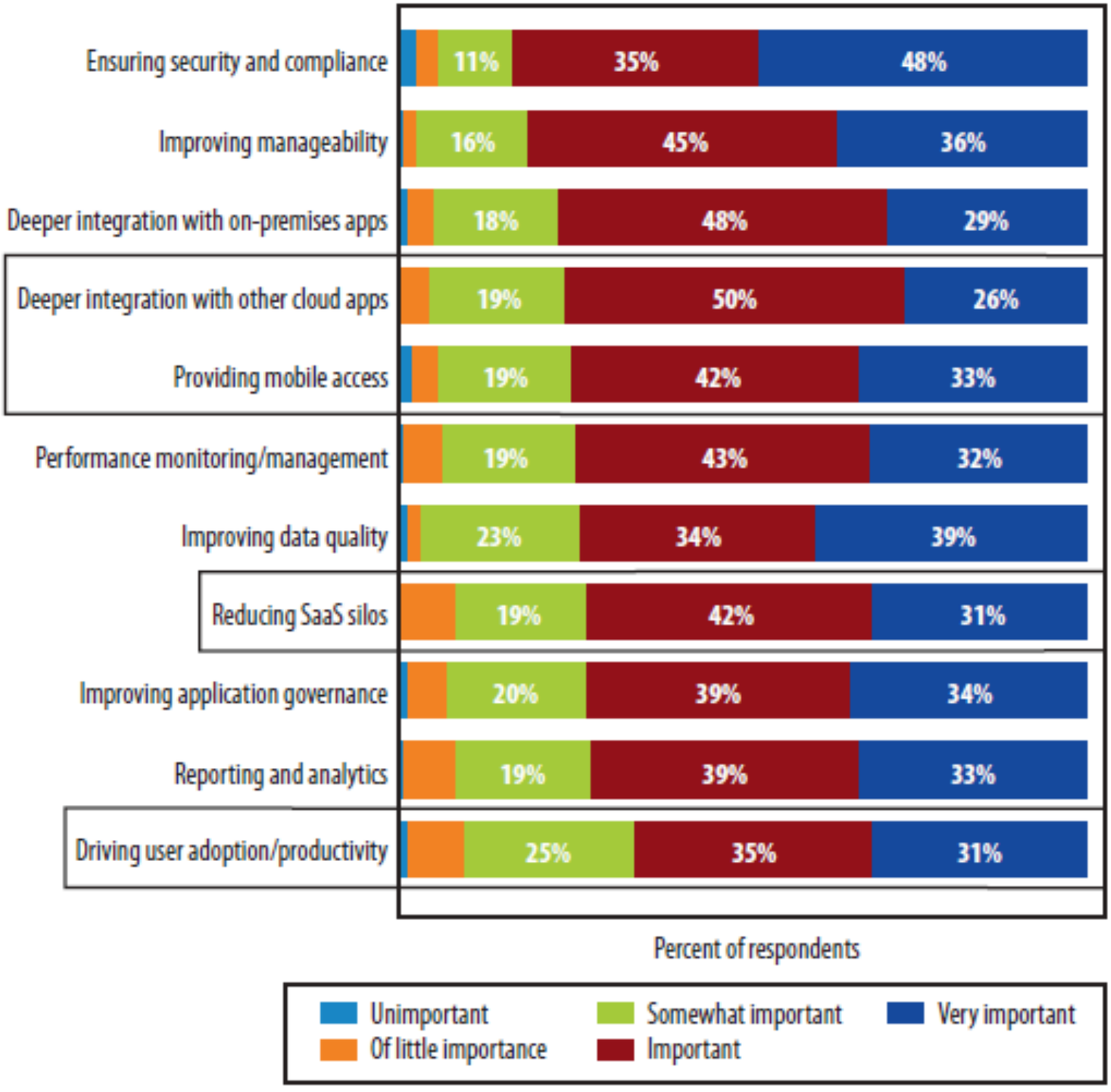
PERIODO DI 1 ANNO STANDARD					
Opzione di pagamento	**Pagamento anticipato**	**Mensile***	**Tariffa oraria****	**Risparmio rispetto a tariffe on demand**	**Tariffa oraria on demand**
Nessun pagamento anticipato	0,00 USD	219,73 USD	0,301 USD	30%	0,428 USD
Pagamento anticipato parziale	1.254,00 USD	104,39 USD	0,286 USD	33%	
Pagamento anticipato intero costo	2.457,00 USD	0,00 USD	0,28 USD	34%	

Recap
Concetti e parole chiave

- Risorse IT on demand
- Migliori benefici in un contesto affidabile
- Insieme di risorse informatiche virtualizzate
- Assegnamento rapido di risorse a run-time
- Sistemi ad alta scalabilità architetturale

- On demand
- Affidabilità
- Virtualizzazione
- Provisioning
- Scalabilità

Cloud emerging challenges



AWS

Amazon Web Services (AWS) è una piattaforma di servizi cloud sicura, in grado di offrire potenza di elaborazione, storage di database, distribuzione di contenuti e altre funzionalità per aiutare le imprese a ricalibrare le proprie risorse e crescere in modo organico.

Gartner Magic Quadrant for Cloud Infrastructure as a Service 2018

Gartner®

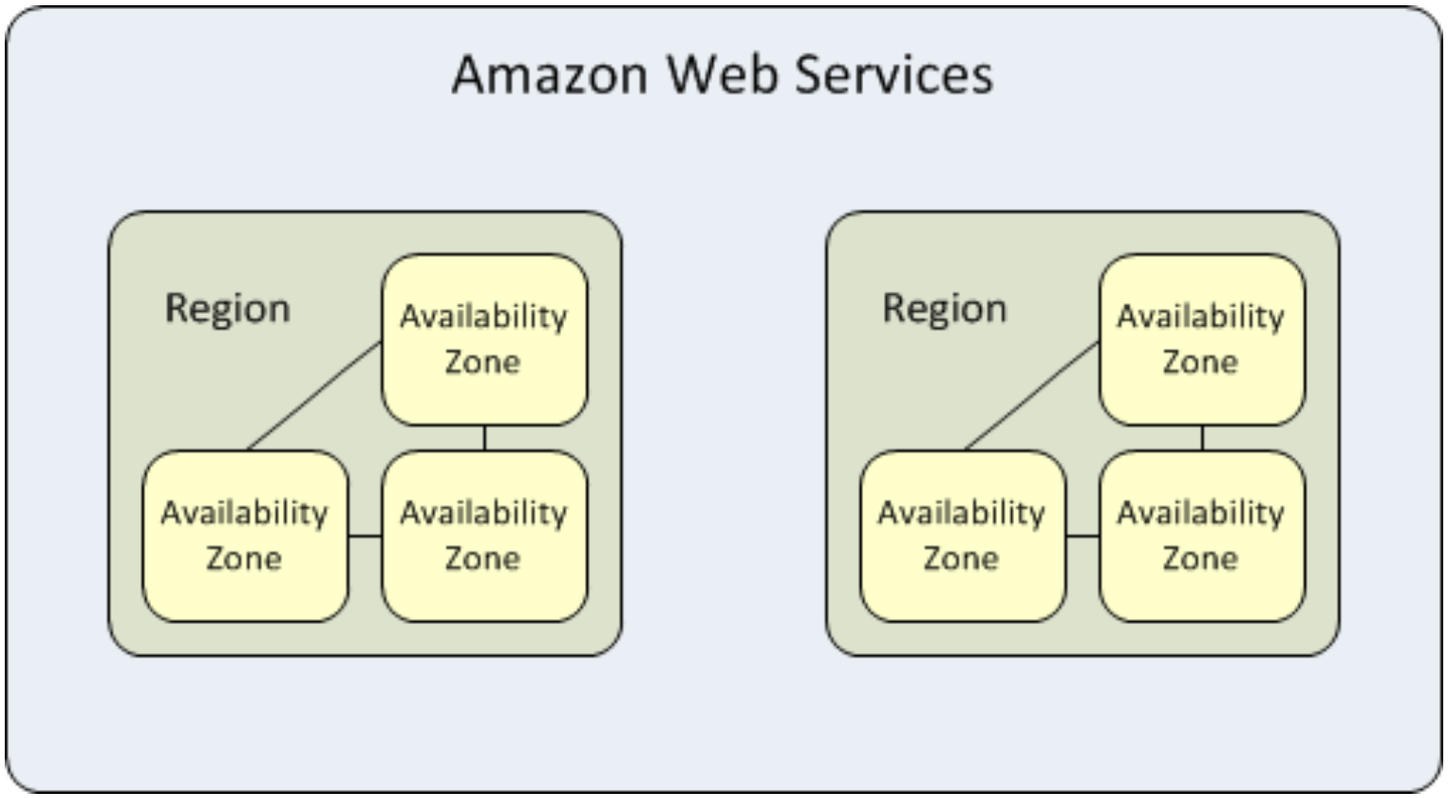
Region

AWS provides customers with the flexibility to place instances and store data within multiple geographic regions called Region. Each region is an independent collection of AWS resources in a defined geography.


Regioni
Disponibile a breve

Availability Zones

Each Region has multiple, isolated locations known as Availability Zones. Amazon EC2 provides you the ability to place resources, such as instances, and data in multiple locations.



AWS Servizi


Enterprise Applications
Virtual Desktops
Sharing & Collaboration
Platform Services
Analytics
Hadoop
Real-time Streaming Data
Data warehouse
Data Pipelines
App Services
Queuing & Notifications
Workflow
App streaming
Transcoding
Email
Search
Deployment & Management
One-click web app deployment
Dev/ops resource management
Resource Templates
Mobile Services
Identity
Sync
Mobile Analytics
Push Notifications
Administration & Security
Identity Management
Access Control
Usage Auditing
Key Storage
Monitoring and Logs
Core Services
Compute (VMs, Auto-scaling and Load Balancing)
Storage (Object, Block and Archival)
CDN
Databases (Relational, NoSQL, Caching)
Networking (VPC, DX, DNS)
Infrastructure
Regions
Availability Zones
Points of Presence

Quicksight

Rekognition

ML

SageMaker

Redshift

EC2

EMR

Athena

Lambda

S3

S3

Storage

Alta scalabilità

Servizio

Web Store, No file system

Accesso via API

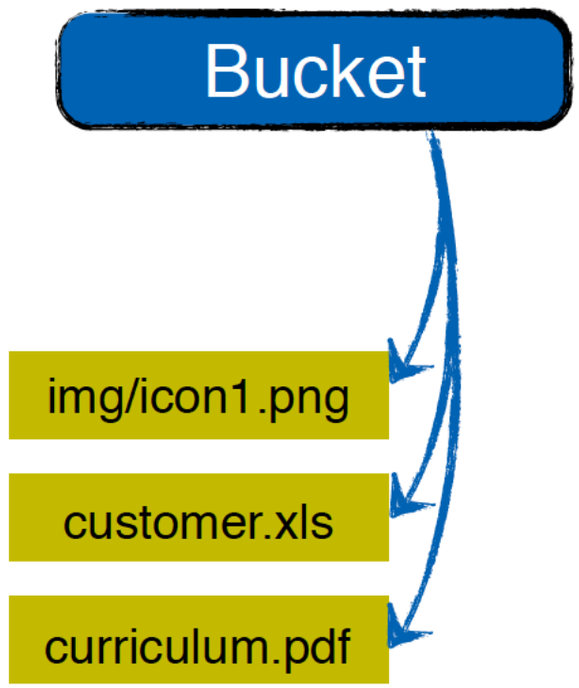
Veloce

Alta disponibilità

Economico

S3

- Object Store
- Nativamente supportato da Spark, Hive, …
- E’ un servizio (non mi servono cluster)
- Oggetti (file illimitati)
- Basso costo
- Sicurezza



AWS Elastic Compute Cloud (EC2)

- Virtual Machines
- AWS has everything from the tiny to gigantic
  - T2.Nano: 1 VCPU 512 MB Ram
  - X1.32xlarge: 128 VCPU 2000 GB Ram
    - GPUS! Useful for deep learning
- Priced per-second
- Options for On-Demand and “Spot Instances”
  - Spot instance: Auction for unused EC2 capacity; generally much cheaper than On-Demand

Amazon
Elastic Map Reduce

- Hosted Hadoop, Spark, HBase, Presto, Hive clusters
- Performs all necessary cluster scaling / provisioning automatically

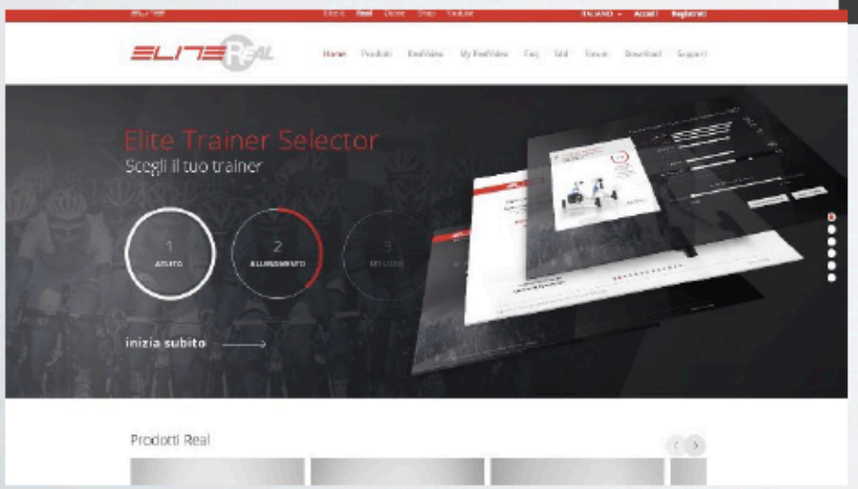
Recording
video &
Subscription

Training
⟷
Personal
data

AWS

MongoDB

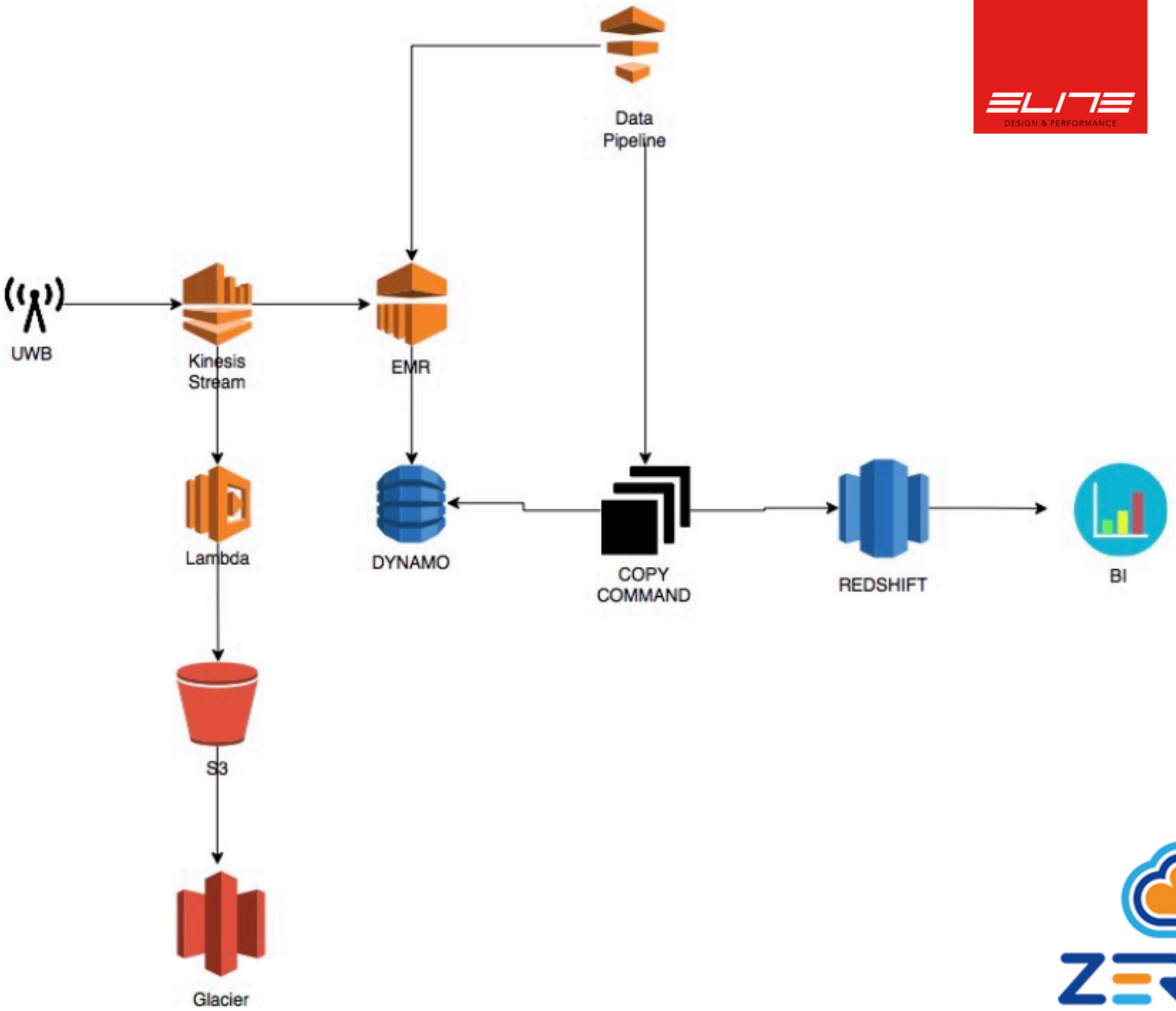
Real-Time data
Analysis

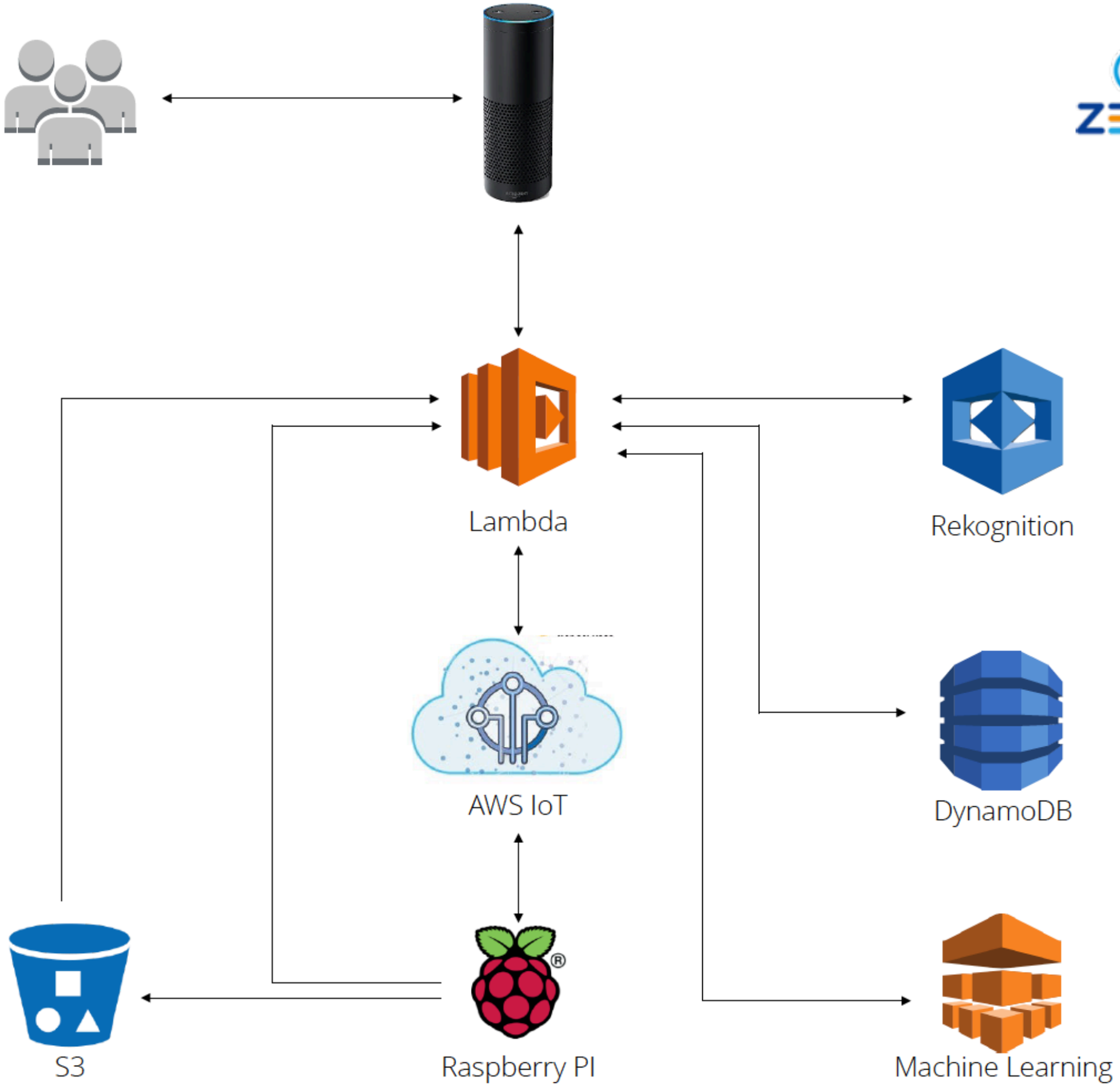


Configurator

ZERO12

VPC, EC2, S3, CDN, CodeDeploy, Autoscaling


Data
Pipeline
UWB
Kinesis
Stream
EMR
Lambda
DYNAMO
COPY
COMMAND
REDSHIFT
BI
S3
Glacier


ZERO12
Lambda
Rekognition
AWS IoT
DynamoDB
S3
Raspberry PI
Machine Learning

Input

Unstructured Data

Automation & management

System and process monitoring

Data Ingestion

Data Processing

Modelling, Machine Learning, AI

Data visualization

Data storage & archiving

Output

Dashboard and interactive report

Machine to machine

Web App

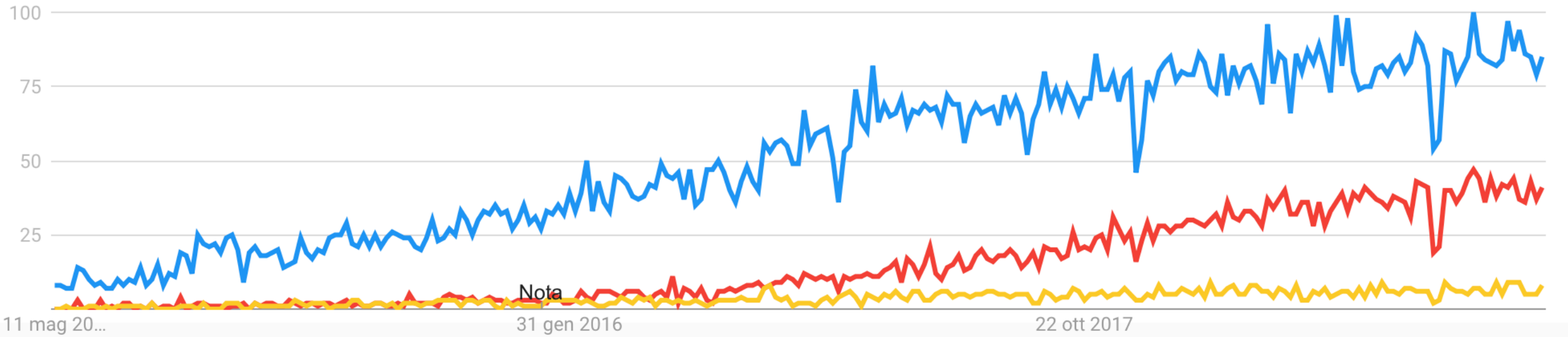
Tabulaex
Analytics for new business
A BURNING GLASS COMPANY

Micro-services

docker
Termine di ricerca

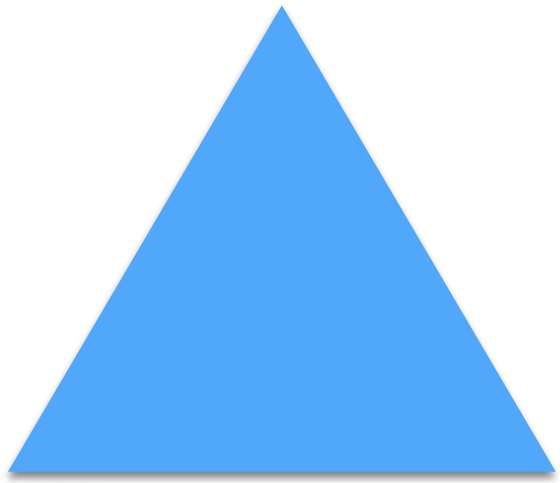
kubernetes
Termine di ricerca

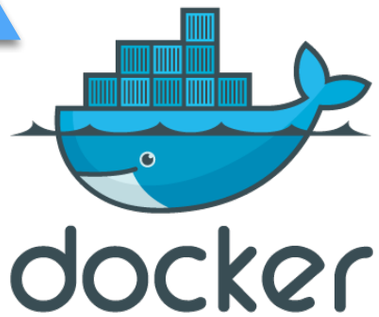
microservices
Termine di ricerca



Contesto

Microservices





Contesto

Manutenibilità

Monitoring

Scalability

Updates

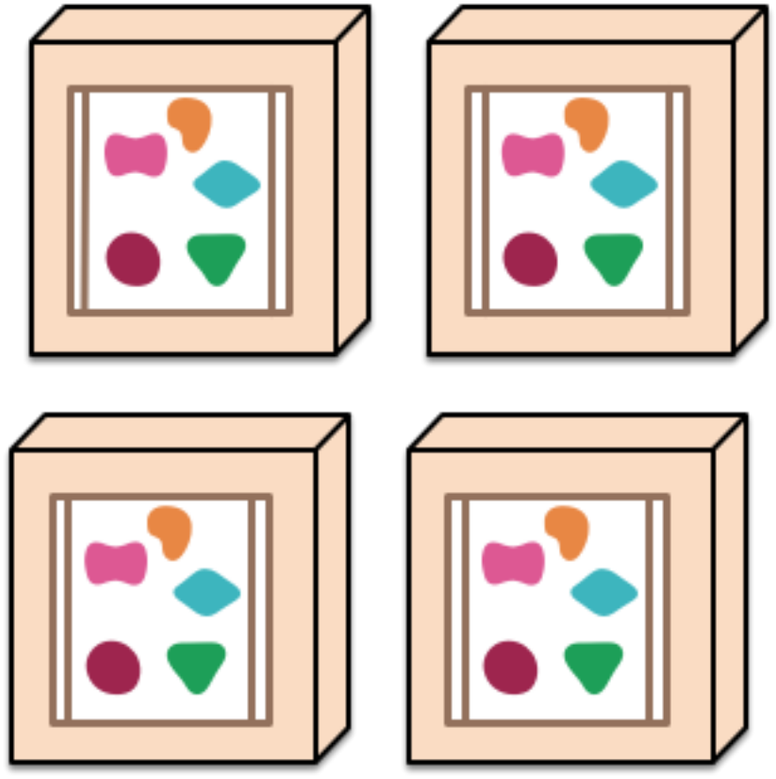
Onboarding

Micro-sevices

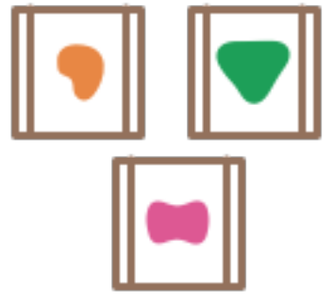
James Lewis and Martin Fowler

A monolithic application puts all its functionality into a single process...

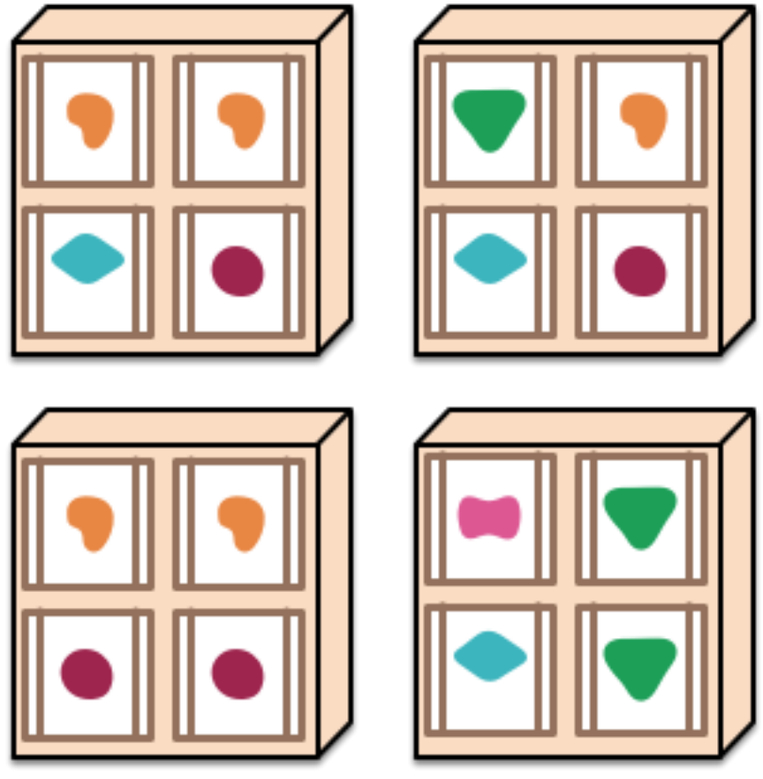
... and scales by replicating the monolith on multiple servers



A microservices architecture puts each element of functionality into a separate service...



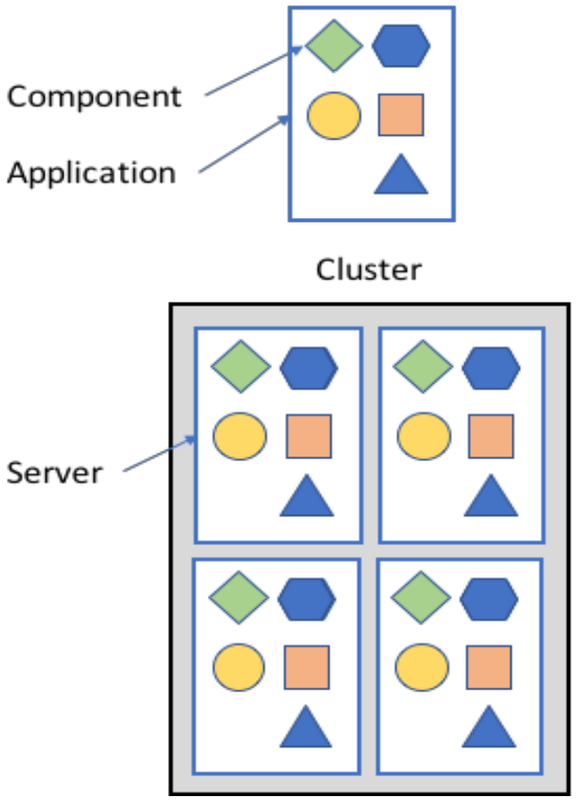
... and scales by distributing these services across servers, replicating as needed.



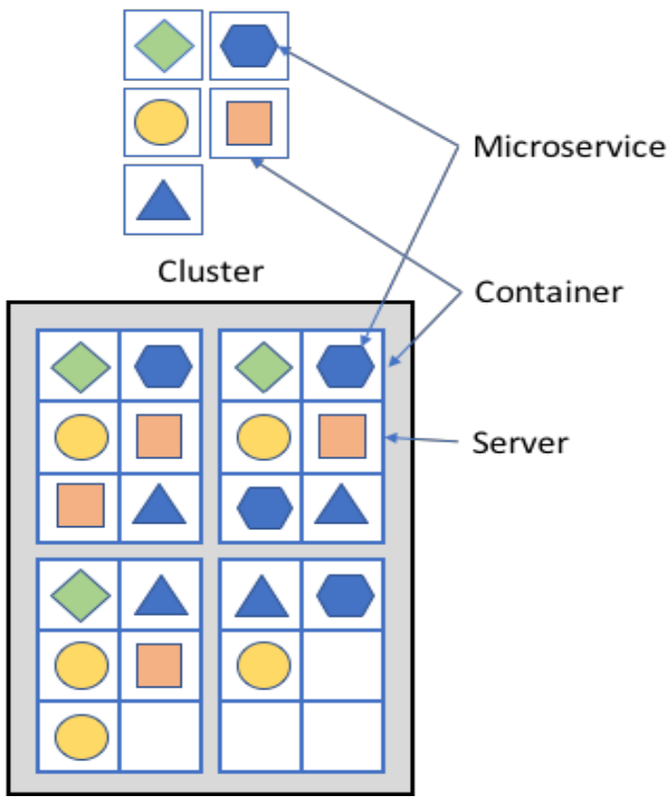
https://martinfowler.com/articles/microservices.html

Functional system decomposition into manageable and independently deployable components

Traditional Monolithic Architecture



Microservice Architecture



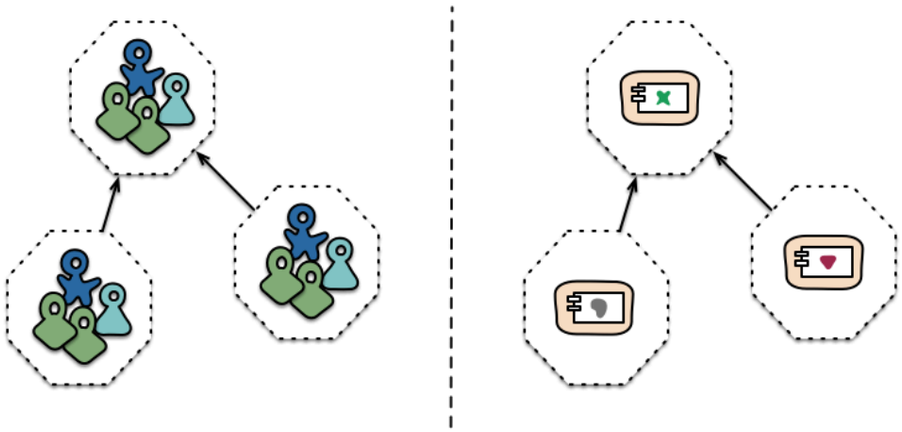
"Monoliths and microservices are not a simple binary choice.

Both are fuzzy definitions that mean many systems would lie in a blurred boundary area among the two"

Caratteristiche

Componentization
via services

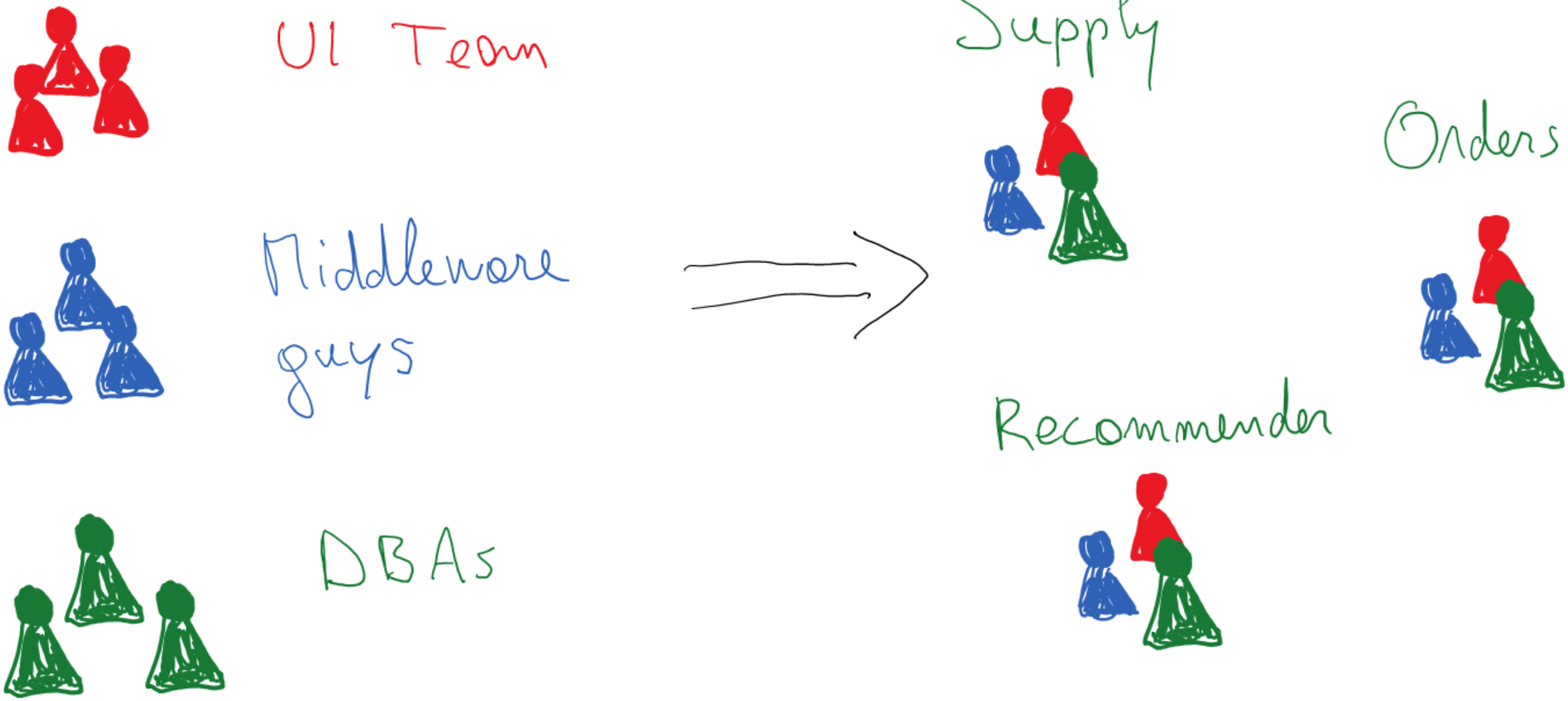
Products,
not projects



Organized around
business capabilities

Decentralized

Organize around business services



Source: Martin Fowler

Monolithic app

Scritta in un linguaggio (C++)

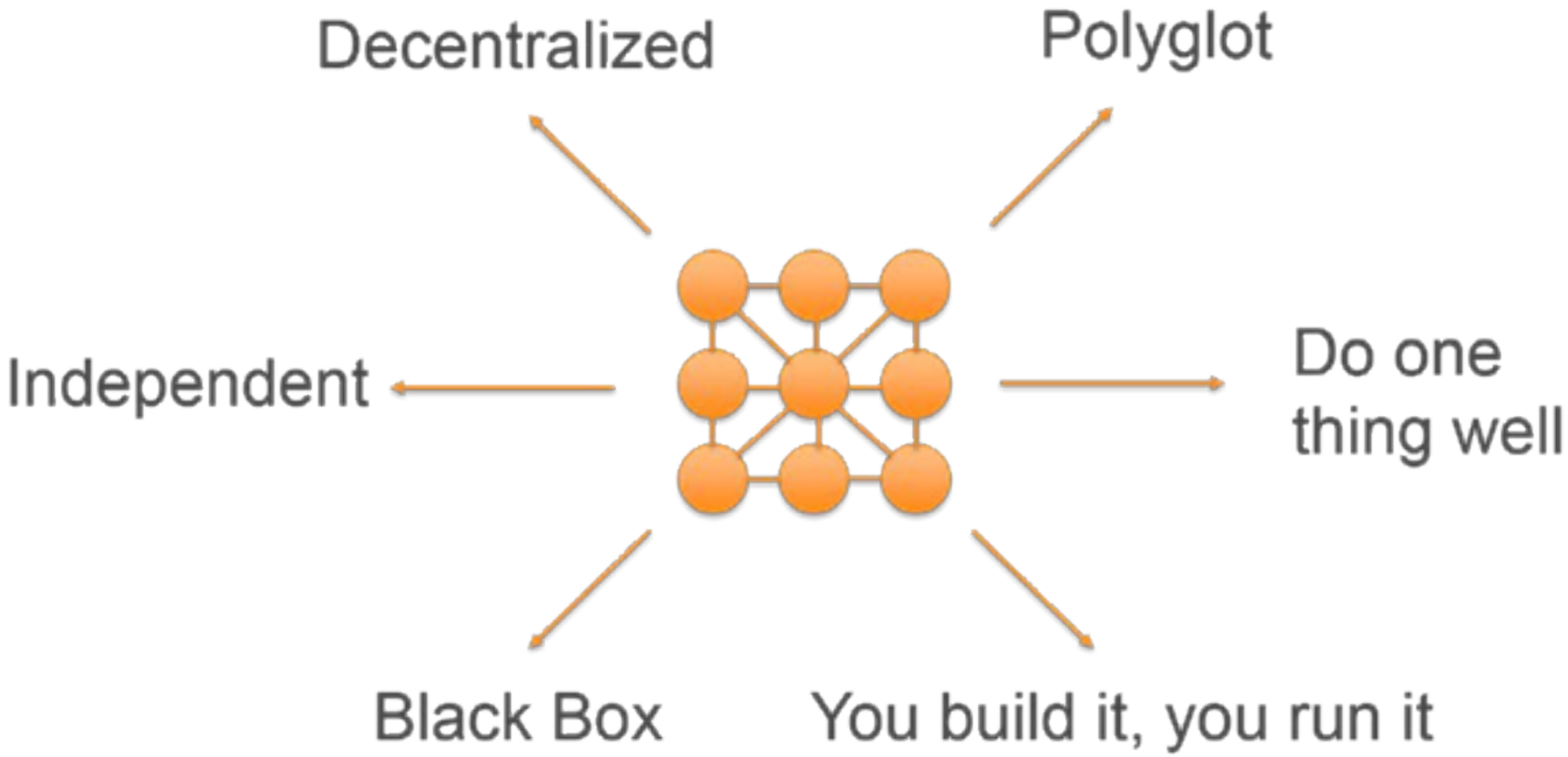
Tiered architecture (MVC pattern)

Java

Polyglot microservices

Mix di linguaggi

Recap



Source: AWS

Docker

Docker tries to solve the "run anywhere" problem using (easily) Linux containers.

Perchè?

Development environment

Your script / program

python3	Lib1 v2	Lib2 v2

Production environment

Your script / program

python3	Lib1 v2	Lib2 v2

My script / program

python2	Lib1 v1	Lib2 v1

Development environment

Your script / program

Production environment

Your script / program

My script / program

Conflict!!!

Development environment

Your script / program

python3 | Lib1 v2 | Lib2 v2

Production environment

Your script / program

python3 | Lib1 v2 | Lib2 v2

Docker container

My script / program

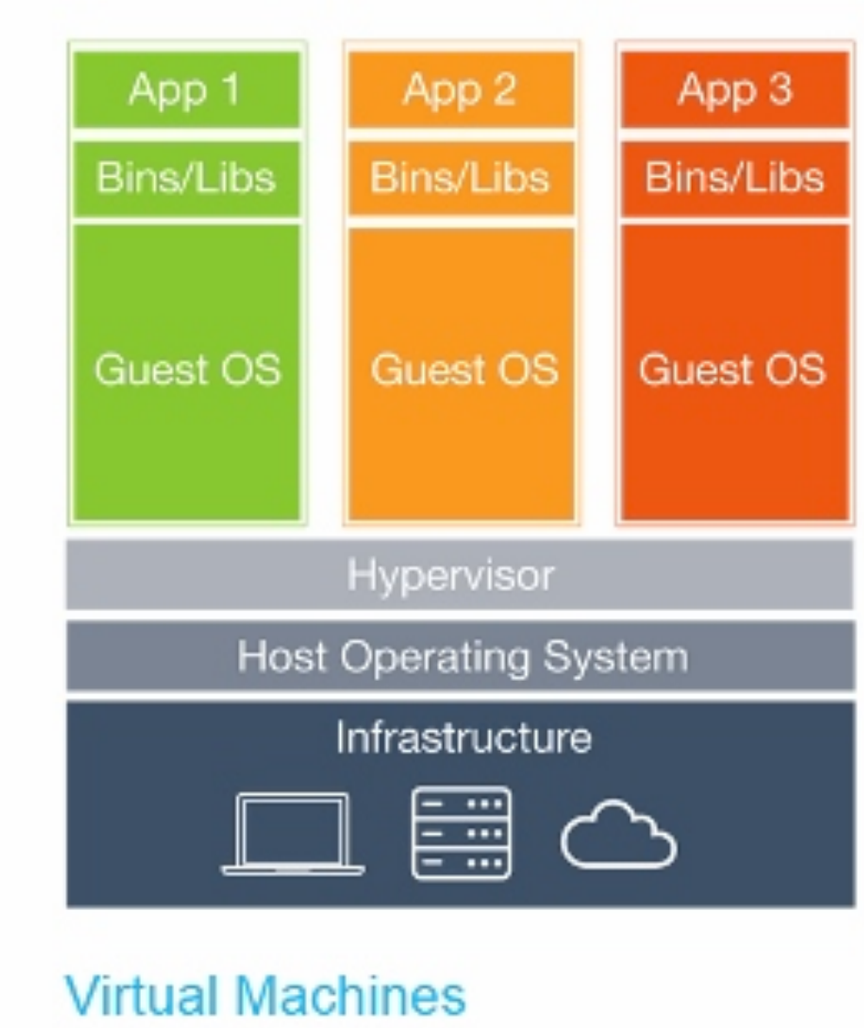
python2 | Lib1 v1 | Lib2 v1

Docker container

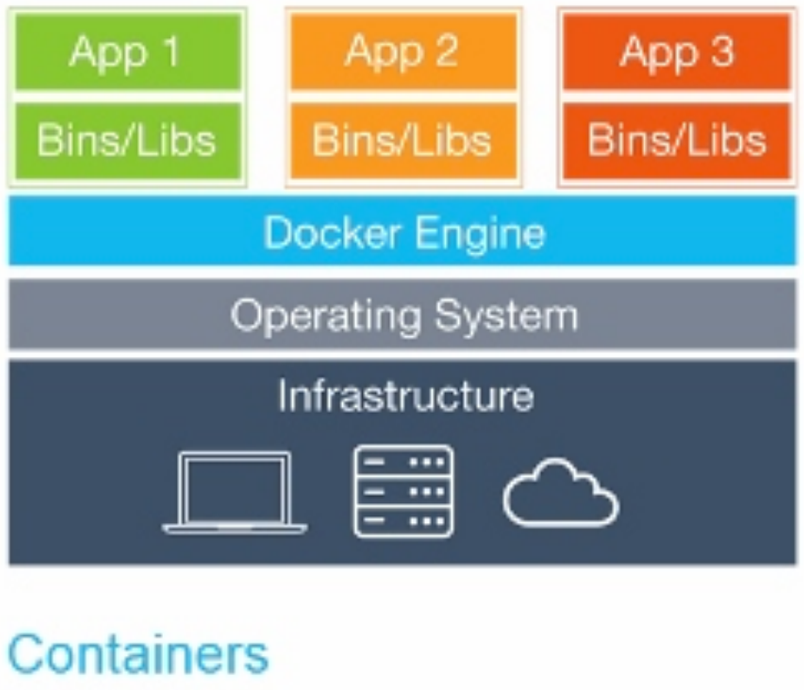
Docker Engine

Efficienza

Perché è diverso da una virtual machine?
(VMware, Xen, …)



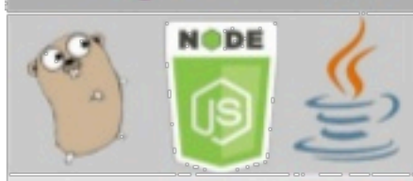
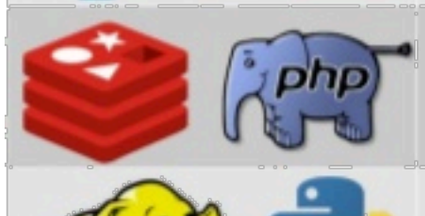
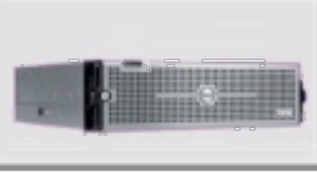
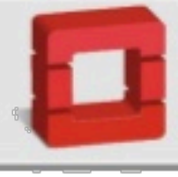
Virtual Machines



Containers

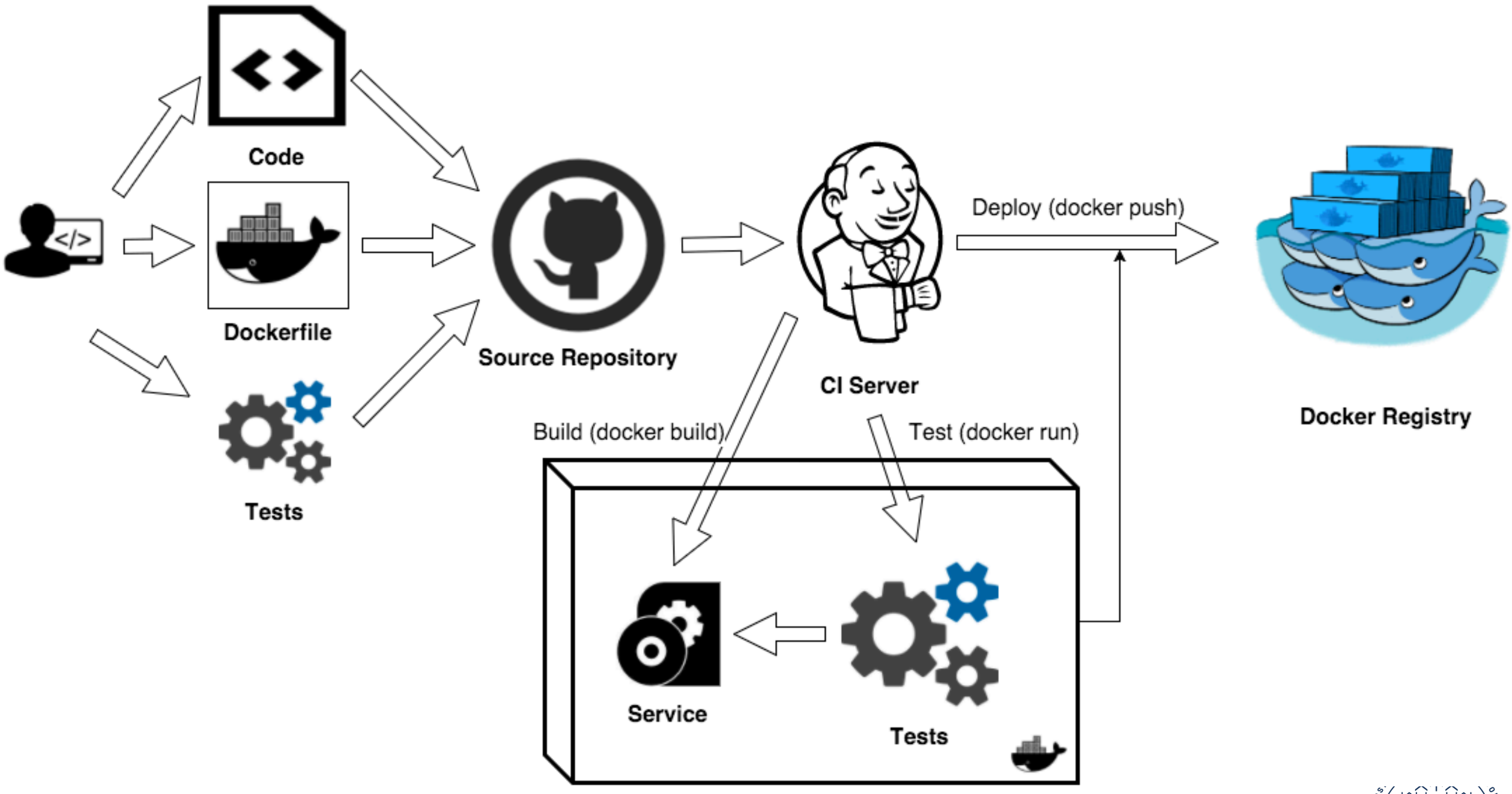
https://blog.jayway.com/2015/03/21/a-not-very-short-introduction-to-docker/

Portabilità

	?	?	?	?	?	?
	?	?	?	?	?	?
	?	?	?	?	?	?
	?	?	?	?	?	?
	?	?	?	?	?	?
	?	?	?	?	?	?
						

The matrix from hell

Flessibilità



Build, ship and run any app, anyware

Sicurezza

Sicurezza

- Don't give root
- If the application needs root, give looks-like-root
- If that's not sufficient, give root, but build another wall

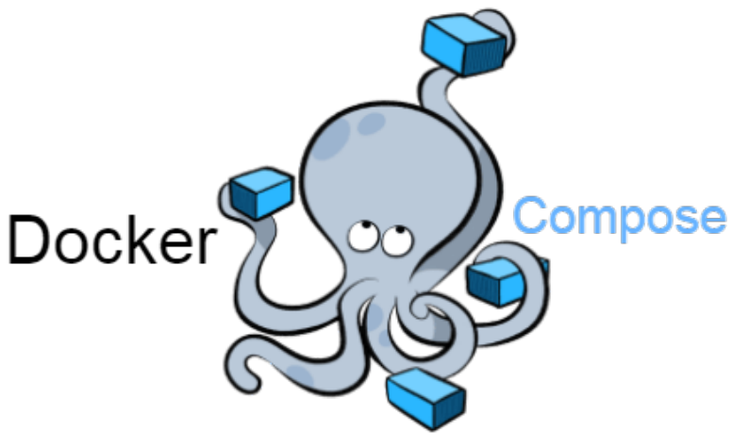
Docker
Immagini e container

An image is an inert, immutable, file that's essentially a snapshot of a container.

Images are created with the build command, and they'll produce a container when started with run. Images are stored in a Docker registry

https://stackoverflow.com/questions/23735149/what-is-the-difference-between-a-docker-image-and-a-container

Immagini



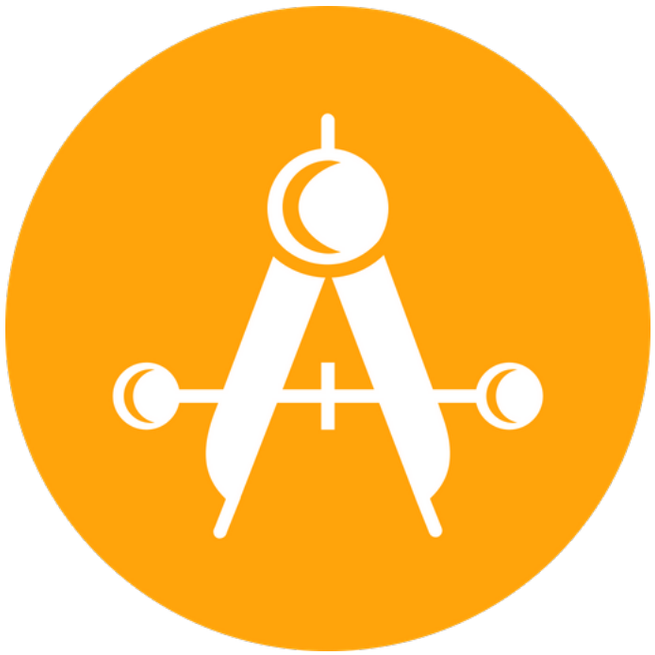
Immagini private (Dockerfile)

Docker Hub

Registri privati

https://github.com/docker/distribution

Come definire un’immagine e avviarla?

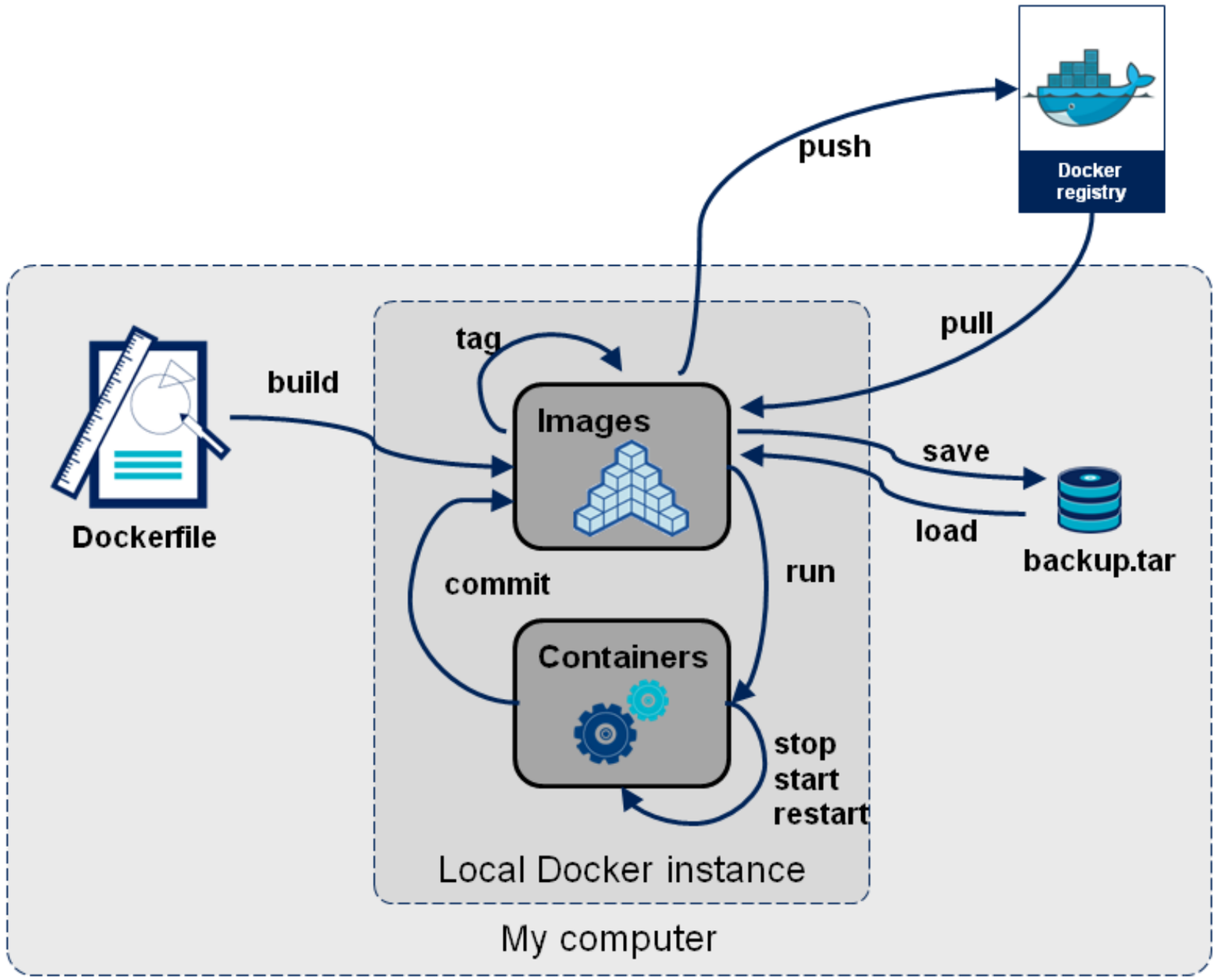


Scrivere un dockerfile

Fare il built dell’immagine

Avviare il container

Come definire un'immagine e avviarla?



http://blog.octo.com/en/docker-registry-first-steps/

Container programming
Dockerfile

```
# Version: 0.0.1
        FROM sequenceiq/hadoop-docker:2.7.1
        LABEL maintainer="alessandrov87 (https://github.com/AlessandroVaccarino)"
        # Download Pig
        RUN curl http://apache.mirror.anlx.net/pig/latest/pig-0.17.0.tar.gz | tar -zx -C /usr/local
        ENV PATH /usr/local/pig-0.16.0/bin:$PATH
        ENV PATH /usr/local/hadoop/bin:$PATH
        # Download and Init Hive
        RUN curl http://apache.mirror.anlx.net/hive/stable/apache-hive-1.2.2-bin.tar.gz | tar -zx -C /usr/local
        ENV PATH /usr/local/apache-hive-1.2.1-bin/bin:$PATH
        # Downlaod Zookeeper (for HBase)
        RUN curl http://apache.mirror.anlx.net/zookeeper/stable/zookeeper-3.4.12.tar.gz | tar -zx -C /usr/local
        ENV PATH /usr/local/zookeeper-3.4.12/bin:$PATH
        # Download HBase
        RUN curl http://apache.mirror.anlx.net/hbase/stable/hbase-1.2.6-bin.tar.gz | tar -zx -C /usr/local
        ENV PATH /usr/local/hbase-1.2.6/bin:$PATH
        # Configure HBase
        RUN rm /usr/local/hbase-1.2.6/conf/hbase-site.xml
        COPY hbase-site.xml /usr/local/hbase-1.2.6/conf/
        # Configure boostrap file
        RUN rm /etc/bootstrap.sh
        COPY bootstrap.sh /etc/
        RUN chown root:root /etc/bootstrap.sh
        RUN chmod 700 /etc/bootstrap.sh
        # Expose Ports
        EXPOSE 10020
        EXPOSE 50070
        EXPOSE 50030
        EXPOSE 8088
        EXPOSE 16000
        EXPOSE 16010
        EXPOSE 16020
        EXPOSE 16030
        EXPOSE 16100
```

Dockerfile

FROM	Sets the base image for subsequent instructions
ENV	Set environment variable in container
RUN	Execute command in the image and commit results
ADD	Add file from host/URL to container
VOLUME	Specify directory that lives outside union fs
EXPOSE	Specify ports to open between containers
CMD	Default command when executing container

Docker
Gestione

docker ps	Mostra l’elenco dei container attivi
docker stop ubuntu	Ferma un container
docker rm ubuntu	Rimuove un container
docker run ubuntu …	Avvia un container

Docker
Hello World

```
sudo apt install docker.io

sudo docker run ubuntu:14.04 /bin/echo 'Hello world'
```

```
master@master-pc:~$ sudo docker run ubuntu:14.04 /bin/echo 'Hello world'
Unable to find image 'ubuntu:14.04' locally
14.04: Pulling from library/ubuntu
d6fdcbe24ed5: Downloading [=============================================>     ]  61.77MB/67.19MB
1115217119d4: Download complete
8180a23d2c44: Download complete
ceab6b3c2a5e: Download complete
```

Docker
Hello World

```
docker run ubuntu:14.04 /bin/echo 'Hello world'
Unable to find image 'ubuntu:14.04' locally
Pulling repository ubuntu
6b4e8a7373fe: Download complete
511136ea3c5a: Download complete
b18d0a2076a1: Download complete
67b66f26d423: Download complete
25c4824a5268: Download complete
8b1c48305638: Download complete
c900195dcbf3: Download complete
Hello world
```

Recap

- Docker permette di costruire sistemi modulari e portabili
- Possiamo distribuirlo ovunque: on premises o in cloud
- Dobbiamo orchestrare i container: Kubernetes
- Iniziate ad usare Docker e Kubernetes!!!
  - Google ha introdotto i container più di 10 anni fa

Server-less architecture

Serverless computing
(also known as *functions as a service*)
is a new cloud computing abstraction that makes it easier to write robust, large-scale web services.

In serverless computing, programmers write what are called serverless functions, which are programs that respond to external events.

When **demand for the serverless function spikes**, the platform automatically allocates additional hardware and manages load-balancing

When **demand falls**, the platform silently deallocates idle resources; and when the platform detects a failure, it transparently retries affected requests.

Serverless
Function as a service

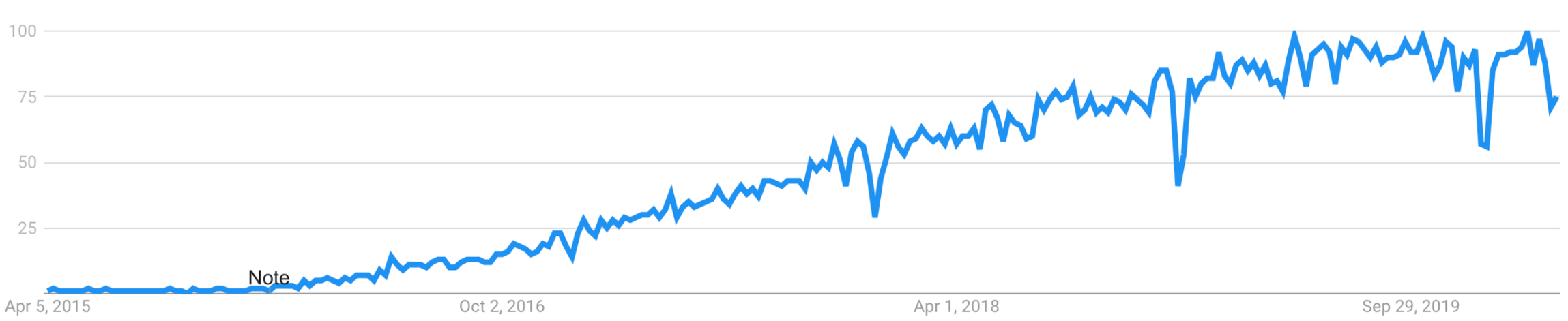
- Allows for granularly billing by cloud service providers.
- An event-driven programming model and architecture.
- Handles most, if not all, operational concerns.
- Removes control over resources.
- Often short-lived.
- Auto-scaling and provisioning.
- Implicit high availability.
- Implicit fault tolerance.
- No servers

serverless
Search term

+ Compare

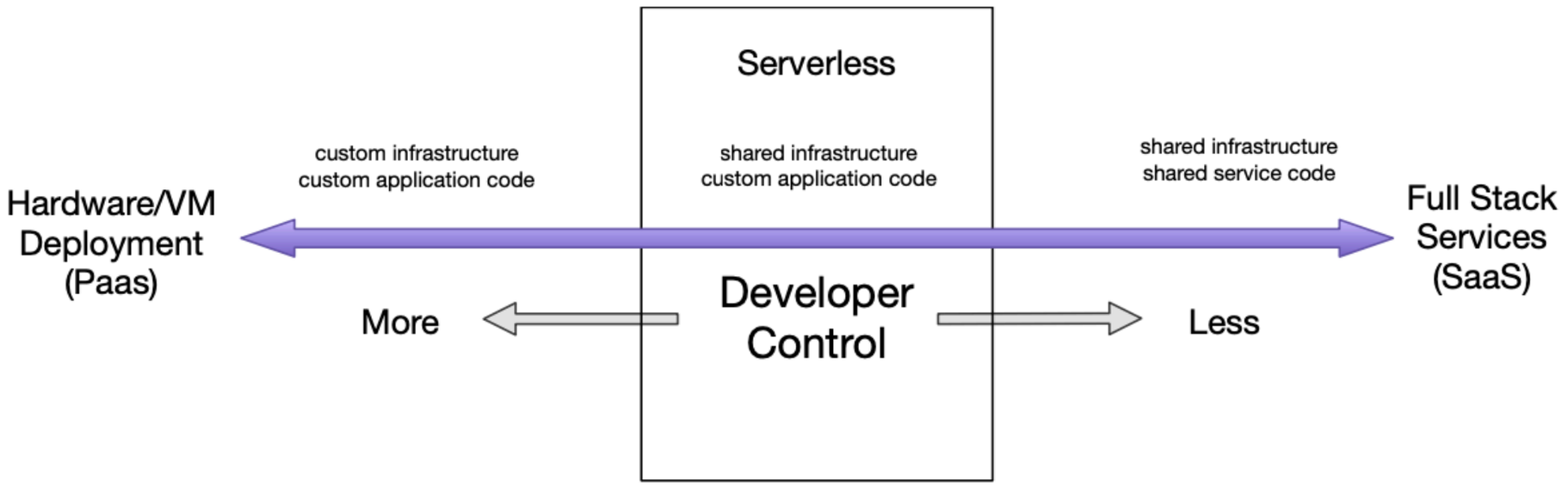
Worldwide ▾ Past 5 years ▾ All categories ▾ Web Search ▾

Interest over time

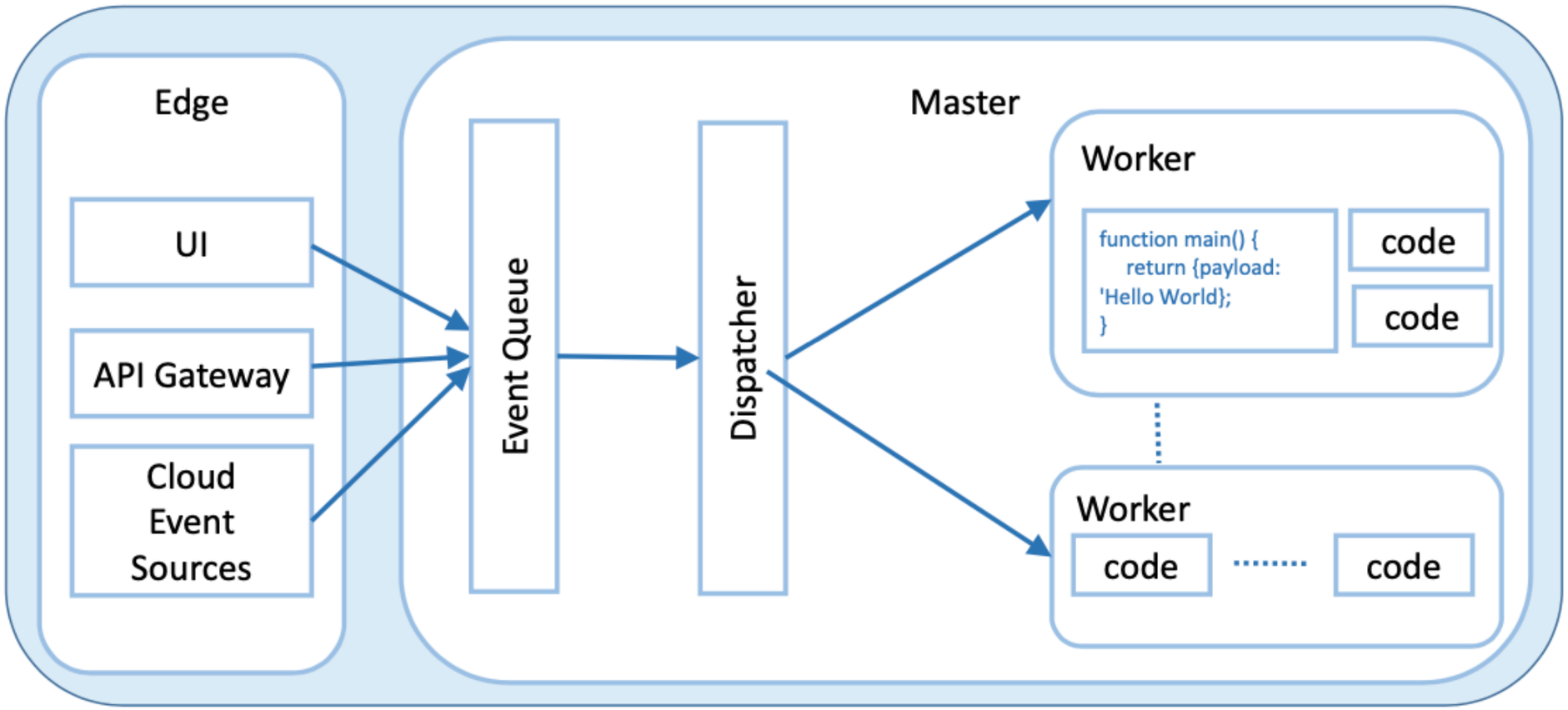


Enterprise Hardware	Enterprise Data Center	Co-location	Cloud Computing	Hybrid Cloud
Atomic Unit of Compute	Physical Server	Virtual Machines	Containers	Functions
Software Architecture	Monolith	n-Tier	Microservices	Serverless

Developer control and Serverless computing



Serverless platform architecture



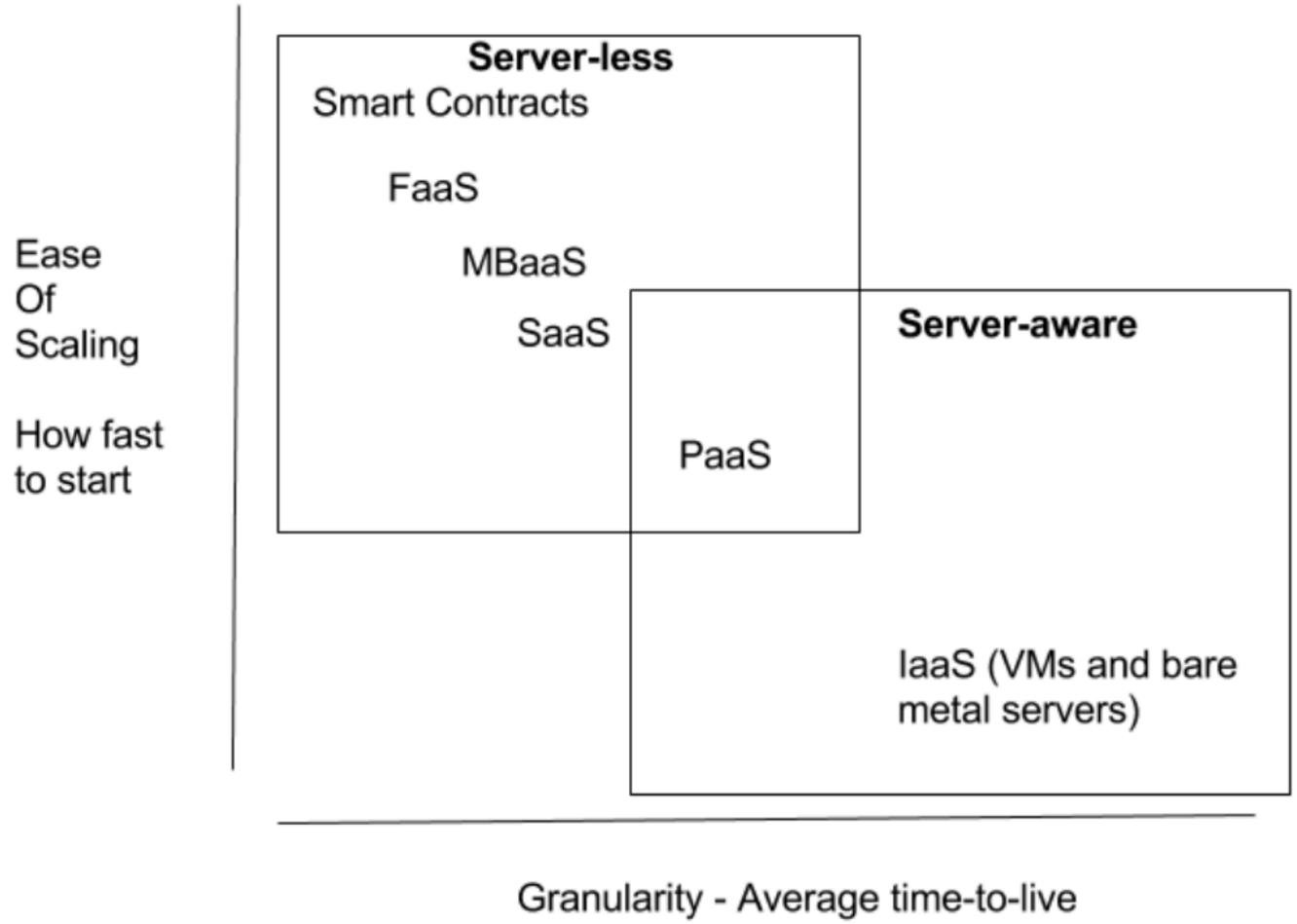
Characteristics

- Cost
- Performance and limits
- Programming languages
- Programming model
- Composability
- Deployment
- Security and accounting
- Monitoring and debugging

Programming model

```
function main(params, context) {
    return {payload:  'Hello, ' + params.name
                    + ' from ' + params.place};
}
```

Time-to-live vs Scaling



The figure is showing relation between time-to-live (x-axis) and ease-of-scaling (y-axis). Server-aware compute (bare metal, VMs, IaaS) has long time to live and take longer to scale (time to provision new resources); server-less compute (FaaS, MBaaS, PaaS, SaaS) is optimized to work on multiple servers and hide server details.

@Serverless Computing: Current Trends and Open Problems
Ioana Baldini, Paul Castro, Kerry Chang, Perry Cheng, Stephen Fink, Vatche Ishakian, Nick Mitchell, Vinod Muthusamy, Rodric Rabbah, Aleksander Slominski, Philippe Suter

01

API Gateway: The API Gateway acts as the communication layer between the frontend and the FaaS layer. It maps REST API endpoints with the respective functions that runs the business logic. With servers out of equation there is no need for deploying and manage load balancers also in this model.

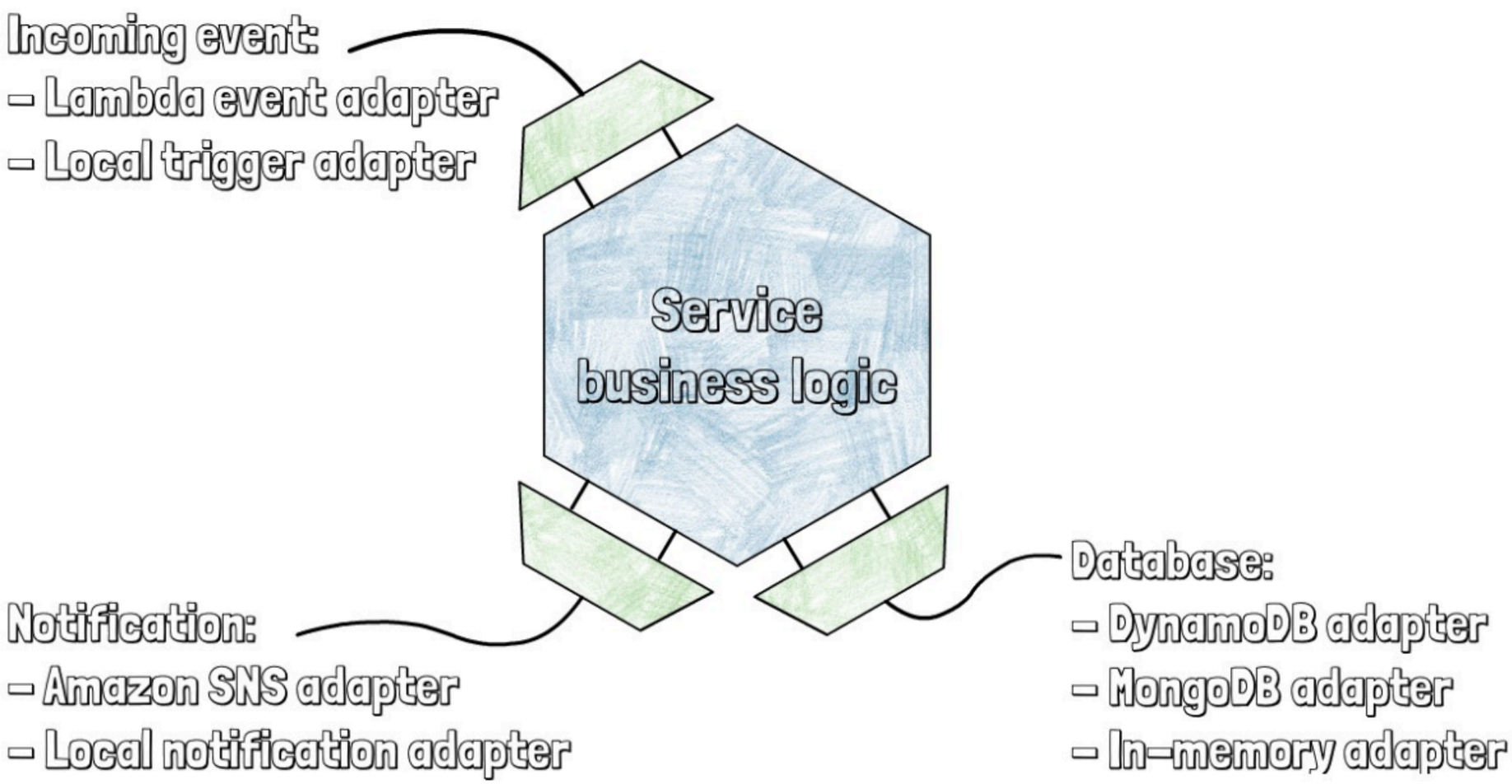
02

Functions or Function as a Service (FaaS): This is the layer that executes specific business logic (or code) with the cloud provider providing the level of abstraction in terms of executing the business logic.

03

Backend as a Service (BaaS): This is essentially a cloud based distributed NoSQL database which essentially removes database administration overheads.

Service Design



Serverless means:

Greater agility

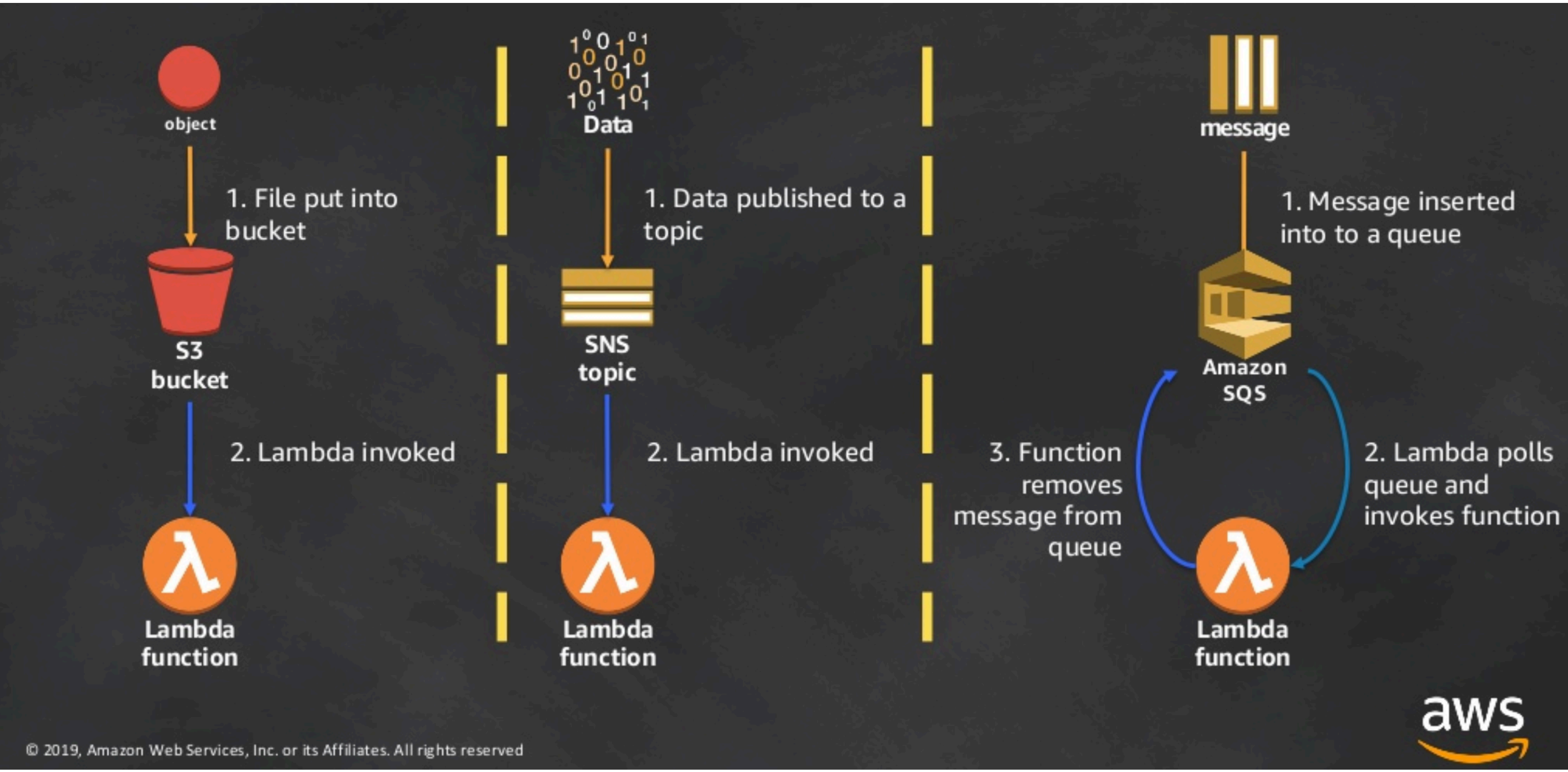
Less overhead

Better focus

Increased scale

More flexibility

Faster time to market


object
1. File put into bucket
S3 bucket
2. Lambda invoked
Lambda function
Data
1. Data published to a topic
SNS topic
2. Lambda invoked
Lambda function
message
1. Message inserted into to a queue
Amazon SQS
3. Function removes message from queue
2. Lambda polls queue and invokes function
Lambda function
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved